Ing. Miroslav Němeček

Průvodce

světem králíka Petra

Gemtree Software

Průvodce světem králíka Petra

Doprovodná příručka uživatele aplikace PETR

prosinec 2000, aktualizace září 2012

petr.hostuju.cz
www.gemtree.com

Text byl původně určen uživatelům licencované verze aplikace Petr, proto některé jeho části mohou být u současné verze již neplatné (např. instalace nyní nevyžaduje instalační CD a licenční disketu).

Obsah:

1 Instalace

Nainstalování do počítače

Práci s Petrem začneme jeho instalací do počítače. Potřebujeme k tomu především instalační CD-ROM, Licenční disketu, dále počítač typu Pentium (nebo alespoň 486) s disketovou mechanikou, CD mechanikou a operačním systémem Windows 95, 98, NT nebo 2000. Na disku počítače by mělo být volné místo 500 MB, v nouzi postačí i 10 MB.

Nejdříve vložte do CD mechaniky instalační CD-ROM. Po chvíli by se mělo objevit okno instalátoru. Pokud se tak nestane, je ve vašem počítači zřejmě vypnuto automatické spouštění CD a musíte instalátor spustit ručně - přes tlačítka **Start** / **Spustit** / **Procházet** nalistujte na CD-ROM program **SETUP.EXE** a ten spusťte.

Okno instalátoru obsahuje čtyři volby. První z nich je určena k instalaci aplikace Petr, druhá k přidání nebo odebrání nainstalovaných částí, třetí k odinstalování celé aplikace Petr a poslední k ukončení instalátoru bez provedení jakýchkoliv změn.

Klikněte na první volbu **Instalovat**. Instalátor vás vyzve k vložení Licenční diskety. Vložte vaši Licenční disketu do disketové mechaniky a klikněte na tlačítko **Další**. Okamžik musíte čekat, než instalátor z diskety načte všechna potřebná data. Po chvíli se objeví okno s Licenční smlouvou. Pozorně si ji přečtěte a potvrďte souhlas s licenčními podmínkami stiskem tlačítka **Souhlasím**.

Po potvrzení Licenční smlouvy se objeví okno voleb pro instalaci. V rámečku vpravo nahoře můžete vidět vaše licenční údaje. Zkontrolujte jejich správnost a úplnost. V levé horní části je okno přepínačů, určujících které části aplikace Petr se budou instalovat. Pravá strana přepínačů obsahuje informace o velikostech instalovaných částí. Zaškrtnutím přepínačů zajistíte nainstalování příslušných částí aplikace Petr.

Nejdůležitější je první přepínač, představující hlavní program aplikace Petr. Ten by měl zůstat vždy zapnutý. Druhým přepínačem nainstalujete ukázkové programy vytvořené v aplikaci Petr. Ukázkové programy jsou dobrým startem i pro vytváření vlastních programů, proto je vhodné tento přepínač také ponechat zapnutý. Další přepínače instalují knihovny obrázků, zvuků, sprajtů atd. Knihovny nejsou nutnými součástmi potřebnými k provozu aplikace Petr, příslušné přepínače je možné vypínat dle zvážení a dle volného místa na disku.

Požadované volné místo na cílovém disku vidíte v pravé části okna uprostřed. V dolní části okna je vypisována cílová složka, kam bude aplikace Petr nainstalována. Složku můžete změnit stiskem tlačítka **Procházet**. Instalaci zahájíte kliknutím na tlačítko **Dokončit**.

Víceuživatelské prostředí

Aplikaci Petr může používat více uživatelů nezávisle na sobě. Víceuživatelské prostředí je zajištěno oddělením složky aplikace Petr od datových složek jednotlivých uživatelů. Každý z uživatelů má svůj pracovní prostor, ve kterém může provádět libovolné změny vlastních i vzorových programů a knihoven aplikace Petr, aniž by tím ovlivnil programy a knihovny ostatních uživatelů.

Klikněte pravým tlačítkem myši v základní ploše Windows na ikonu Petra a z nabídky vyberte volbu **Vlastnosti**. Ve vlastnostech spouštěcí ikony jsou definovány dvě cesty. První cesta je označena **Cíl** a představuje přístupovou cestu k hlavnímu programu aplikace Petr. Typicky zde naleznete **"C:\Program Files\Petr\Peter.exe"**. Tato cesta bude pro všechny uživatele stejná. Druhá cesta je označena **Kde začít** a má význam přístupové cesty do pracovní složky uživatele. Typicky je zde text **C:\Dokumenty\Petr** (podle nastavení systému Windows). Každý uživatel má jinou pracovní složku.

Chcete-li vytvořit spouštěcí ikonu Petra pro nového uživatele, mělo by být vaším prvním krokem vytvoření pracovní složky uživatele. K tomu použijte například systémový program **Průzkumník**. Nalistujte společnou pracovní složku uživatelů (typicky **C:\Dokumenty**) a volbou **Soubor** / **Nový objekt** / **Složka** vytvořte novou složku. Poté pravým tlačítkem myši na ploše Windows popotáhněte spouštěcí ikonu aplikace Petr a volbou **Zkopírovat sem** vytvořte její kopii. Kliknutím pravým tlačítkem myši na novou ikonu vyvolejte nastavení vlastností a upravte cestu označenou **Kde začít** na cestu k pracovní složce nového uživatele. Ještě můžete, opět pravým tlačítkem myši, novou ikonu přejmenovat a tím je vše pro nového uživatele připraveno.

Síťová instalace

Při instalaci do síťového prostředí (například ve školní učebně) nainstalujte aplikaci Petr na **server** sítě stejným způsobem jako na samostatný počítač. Složce s nainstalovanou aplikací Petr můžete po instalaci přidělit přístupová práva **pouze pro čtení**, žádné další zápisy nebudou již do složky prováděny.

Po nainstalování aplikace Petr vytvořte na stanici správce sítě pracovní složku Petra pro ukládání souborů uživatele stanice (např. **H:\Dokumenty\Petr**). Připravte spouštěcí ikonu tak, aby cesta **Cíl** ukazovala na hlavní program aplikace Petr (soubor **Peter.exe**) a cesta **Kde začít** do pracovní složky Petra. Vytvořte pracovní složky i pro ostatní stanice sítě a překopírujte spouštěcí ikonu na pracovní plochy ostatních stanic.

Aplikace Petr, stejně jako jiné 32-bitové aplikace, podporuje dlouhá jména souborů s diakritikou. V síti Novell zajistíte podporu dlouhých jmen těmito úkony:

1. na serveru zadejte příkaz „**load os2**“,
2. na serveru zadejte příkaz „**add name space os2 to** *volume1*“, kde *volume1* je jméno svazku s instalovanou podporou dlouhých jmen,
3. do **STARTUP.NCF** přidejte příkaz „**load os2**“

V některých verzích sítí Novell nemusí jít spouštět programy s diakritikou ve jménech (nejen z Petra, ale ani z **Průzkumníka**), přestože všechny jiné souborové operace pracují správně. V tomto případě je nutné se vyvarovat používání znaků s diakritikou ve jménech programů nebo provést upgrade síťového software.

Přeinstalování

Při nedostatku místa na disku je možné instalovat aplikaci Petr neúplně. Jednotlivé instalované části lze dodatečně přidávat či odebírat volbou instalátoru **Přidat/ubrat**. Po vložení Licenční diskety a načtení licenčních dat se zobrazí výběrové okno stejně jako při instalaci s tím rozdílem, že nelze změnit cílovou složku a že přepínače instalovaných částí modrým zabarvením indikují, které části jsou již nainstalovány. Změnou přepínačů nastavíte nový požadovaný stav instalovaných částí aplikace. Přeinstalování zahájíte tlačítkem **Dokončit**.

Změny ve vzorové knihovně

Do vzorové knihovny mohou být přidávány nebo z ní rušeny soubory dle potřeby. Při odinstalování knihovny zruší instalační program pouze původní nezměněné soubory. Přiinstalováním budou existující soubory stejných jmen přepsány bez ohledu na datum, čas či velikost souborů.

Spuštěním aplikace Petr přes Windows nabídku **Start** můžete provádět změny přímo ve vzorové knihovně. Vzorové složky jsou v této chvíli současně pracovními složkami. Podobně můžete použít volbu **Petr s modifikací vzorové knihovny** z nabídky **Petr x.xx**.

Odinstalování

Aplikaci Petr odinstalujete spuštěním instalátoru z instalačního CD-ROM a kliknutím na volbu **Odinstalovat**. Jinou metodou je vyvolání odinstalátoru přes Windows službu **Přidat nebo odebrat programy** nebo přes nabídku **Start** / **Petr x.xx** / **Odinstalování**.

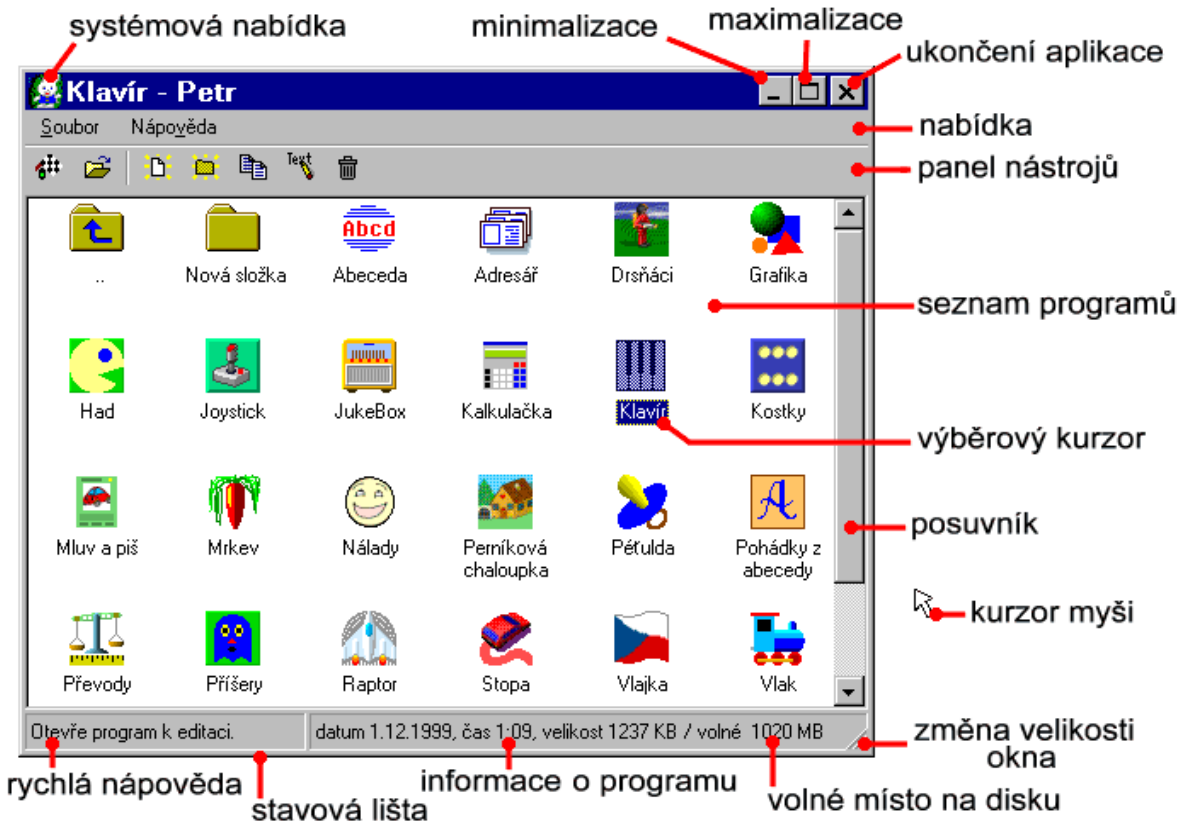
Odinstalováním budou zrušeny všechny nezměněné vzorové programy a knihovny a programové soubory aplikace Petr. Při odinstalování nejsou rušeny pracovní složky uživatelů. V případě potřeby je nutno zrušit složky ručně, například **Průzkumníkem**.

2 Okno programů

Po startu Petra upoutají zřejmě nejvíce vaši pozornost barevné ikony rozmístěné po ploše zobrazeného **okna programů**. Jsou to ukázkové programy vytvořené v Petrovi a je jich skutečně pěkná řádka. Ukázkové programy jsou dobrým startovacím bodem při vlastní tvorbě. Mohou být jak inspirací, tak i odpovědí na otázku „Jak na to?“. Není nic snazšího, než vzít hotový program a jeho úpravami vytvořit program nový.

Jak s Petrem začít? **Hrou.** Důkladně si prohlédněte připravené ukázkové programy, vyzkoušejte je a pohrejte si s nimi. Poznáte, co vše můžete od Petra očekávat, a seznámíte se s pravidly a zvyklostmi pro ovládání programů. Jistě vás přitom napadne nejedna možnost, co a jak udělat lépe.

Nebojte se hrát si, vždyť veškerá tvůrčí práce je vlastně hra. K vašemu dílu pociťujte nadšení a hravost, tak dosáhnete největších tvůrčích úspěchů.



Podívejme se blíže na okno programů. Vyzkoušejte si, jak se programy spouští. Dvakrát klikněte levým tlačítkem myši na ikonu některého z programů. Nebo můžete jedenkrát kliknout na ikonu a potom stisknout klávesu **Enter**.

Ukončení spuštěného programu je možné více způsoby. Většinu programů ukončíte stiskem klávesy **Esc**, mnohé z nich dokonce stiskem libovolné klávesy. Každý program ukončíte kliknutím levým tlačítkem myši na malé tlačítko s křížkem ⊠, nalézající se v pravém horním rohu okna. Pokud vám ani tento způsob nevyhovuje, stiskněte na klávesnici kombinaci kláves **Alt+F4** (držte **Alt** a k němu přidejte **F4**). Tak ukončíte i celoobrazovkové programy, které tlačítko s křížkem ⊠ nemají.

Zůstaňme ještě chvíli u okna spuštěného programu. U většiny programů najdete na horní liště okna kromě ukončovacího tlačítka s křížkem ještě další dvě tlačítka. Prostřední tlačítko s jedním nebo dvěma obdélníčky slouží k přepínání velikosti okna, a to buď přes celou obrazovku nebo na okno s měnitelnou velikostí. Levé tlačítko umožňuje okno programu „sklapnout“ do malé ikony na liště programů Windows. Opětovného rozbalení do původní velikosti dosáhnete kliknutím myší na tuto ikonu.

Při pohybu kurzoru myši přes okraj okna jste si možná již všimli změny vzhledu kurzoru na dvě šipky ↔. Šipky naznačují, že okrajem okna je možné posouvat. Pokud tak uděláte, uvidíte, že okno mění svou velikost a úměrně s ní se mění i zvětšení obsahu okna. Zvětšování obsahu okna je jev netypický pro standardní aplikace Windows, ale je běžnou vlastností grafických programů vytvořených v Petrovi. Tak si snadno roztáhnete svou oblíbenou hru přes celou obrazovku nebo ji naopak zmenšíte do miniaturního okénka.

Další zajímavou vlastností programů vytvořených v Petrovi je možnost pozastavit činnost programu stiskem klávesy **Pause**. Pokračování programu zajistíte stiskem libovolné klávesy, například opět **Pause**.

Poslední, neméně zajímavou, funkci programů Petra vyvoláte stiskem kláves **Alt+Enter** (držte **Alt** a poté přidejte **Enter**). Tato kombinace kláves způsobí přepnutí programu do celoobrazovkového módu nebo zpět. V celoobrazovkovém módu zmizí okraj okna a displej počítače se přepne do módu nejlépe vyhovujícího požadavkům programu. Podmínkou k přepnutí na celoobrazovkový mód je nainstalovaný ovladač DirectX. U systémů Windows 98 a Windows 2000 je ovladač DirectX již součástí systému.

Vraťme se zpět k oknu aplikace Petr. Podobně jako u programů Petra můžete i samotnou aplikaci Petr ukončit kliknutím na tlačítko s křížkem v pravém horním rohu okna nebo stiskem kombinace kláves **Alt+F4**. Stejně tak lze okno aplikace Petr minimalizovat, maximalizovat či taháním za okraj ↔ měnit jeho velikost.

Prohlédněte si znovu obrázek **okna programů** na předešlé stránce. Na ploše okna vidíte vlevo nahoře dvě poněkud zvláštní ikonky. Označují složky. Složky mohou obsahovat programy nebo další složky. Používáme je ke třídění programů do skupin, abychom nemuseli mít všechny programy „na jedné hromadě“.

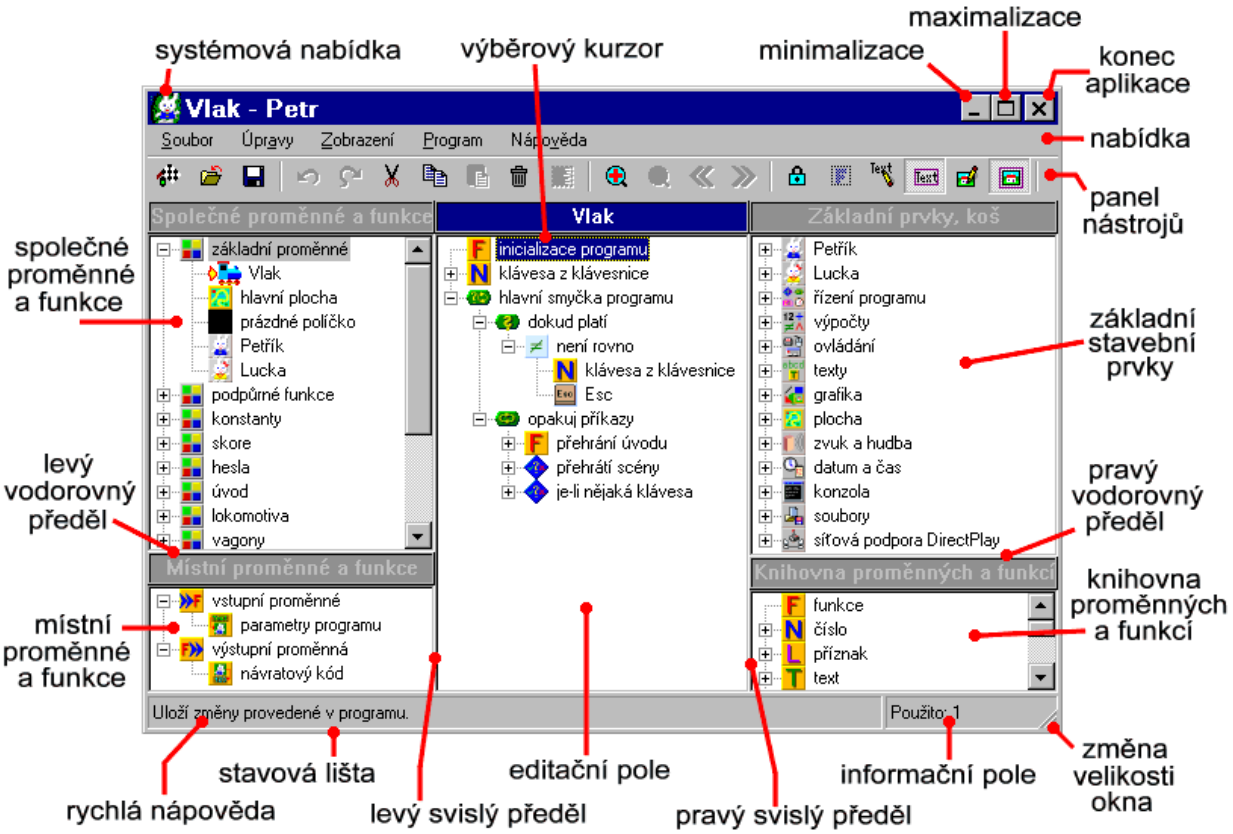
Druhá ikona zleva označuje složku se jménem **Nová složka**. Do složky se vnoříte dvojitým kliknutím levým tlačítkem myši na ikonu, stejně jako při spouštění programu. K návratu ze složky zpět slouží první ikona zleva, označená modrou šipkou a jménem **..** (dvě tečky).

3 Jak to funguje?

Předpokládejme, že jste si již dostatečně pohráli s připravenými programy. Jistě vás teď zajímá, jak to uvnitř takového Petrova programu vypadá.

Podíváme se spolu do některého z programů. Vyberme si například hru **Vlak**. Najdete ji ve složce **Hlavolamy**. Jednou klikněte levým tlačítkem myši na ikonu programu, ikona se orámuje a zvýrazní. Orámování ikony říkáme **výběrový kurzor**, zvýrazněním ikony je indikován **vybraný** program.

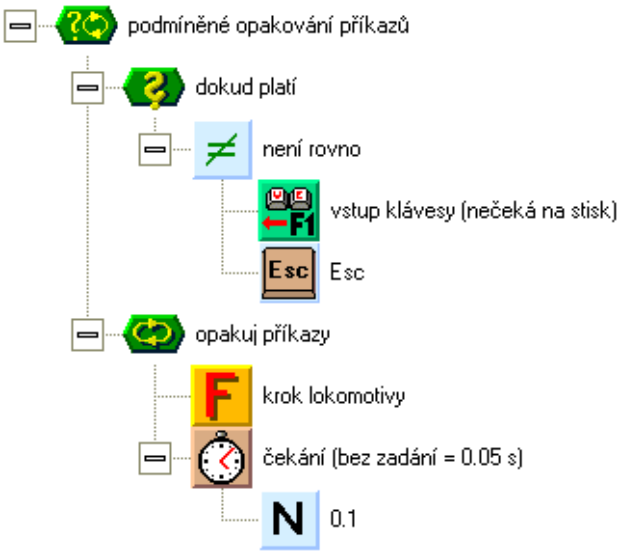
V horní části okna si všimněte pásu s barevnými tlačítky. Je to **panel nástrojů** a obsahuje tlačítka k rychlému vyvolávání akcí. Druhým tlačítkem zleva je tlačítko **Otevřít**  . Otevírá program k editaci (úpravám) obsahu programu. Klikněte na tlačítko levým tlačítkem myši.



Po kliknutí na tlačítko se program rozvine do mnoha barevných ikonek. Vstoupili jsme do editoru programu. Otevřel se před námi celý báječný svět králíka Petra. Že vám zatím ikonky nic neříkají? Žádné strachy, již brzy budete vše zvládat s profesionální lehkostí.

Jak si můžete všimnout, plocha editoru je rozdělena na pět částí - pět oken. Nejvíce nás zajímá prostřední okno. Při pohledu na obrázek vidíte, že je označeno **editační pole**. Toto okno je pro nás nejdůležitější, v něm probíhá skládání samotného programu. Záměrně používám slovo „skládání“, protože v Petrovi se skutečně program skládá jako skládačka z kostek. Každá barevná kostička - ikonka - představuje jeden

příkaz, jeden programový prvek. Všechny prvky dohromady tvoří fungující program, třeba právě hru **Vlak**.



Charakteristickým rysem Petrova programu je, že programové prvky, reprezentované graficky ikonkami, jsou sestaveny v jeden celek pomocí stromových struktur. Jsou to jakési listy stromu s větvemi. Nebo vám ikonky svou barevností a tvary připomínají spíše drahokamy? (Proč o tom mluvím? Hádejte, co bylo inspirací při vzniku názvu firmy **Gemtree**; gem = drahokam, tree = strom).

Jednou z hlavních výhod stromového vyjádření programu je možnost „sklapnutí“ jednotlivých větví programu do ikonek. Můžete ponechat rozevřenou pouze tu část programu, která vás zrovna teď zajímá. Tato vlastnost editoru Petra umožňuje radikální zvýšení přehlednosti programu.

Než se pustíme do přerovnávání programu, musíme se seznámit se základním ovládáním editoru. Při detailnějším pohledu si můžete nalevo od některých ikonek všimnout šedivých čtverečků se značkou „**+**“ nebo „**-**“:

Čtverečky nám napovídají, že daný programový prvek obsahuje uvnitř větev s dalšími prvky. Kliknete-li levým tlačítkem myši na čtvereček se značkou „**+**“, větev se rozevře. Kliknete-li na čtvereček se značkou „**-**“, větev se zavře.

Prohlédněme si okno vlevo nahoře. V titulku okna je nadepsáno **Společné proměnné a funkce**. Každý datový prvek v programu použitý, jako například číselnou proměnnou obsahující číslo či obrázkovou proměnnou obsahující obrázek, musíme nejdříve v okně **Společných proměnných a funkcí** vytvořit (nebo v okně **Místních proměnných a funkcí**, ale o tom až jindy). Úkonu vytvoření datového prvku říkáme v jiných programovacích jazycích **deklarace**, ale protože pracujeme v čistě grafickém prostředí, nemusíme se podobnými pojmy příliš zabývat.

Každý z datových prvků při vytváření programu něco obsahuje. Číselná proměnná obsahuje číslo, obrázková proměnná obrázek, předmětová proměnná předmět (což je v podstatě obrázek o velikosti 32x32 bodů). Obsahy datových prvků můžete prohlížet a měnit dvojitým kliknutím levým tlačítkem myši na ikonu prvku. V editačním okně se objeví obsah datového prvku a nabídka i panel nástrojů se změní podle typu prvku.

Při prohlížení obsahů prvků narazíte na dva zvláštní typy prvků. První z nich obsahuje příkazy programu a říkáme mu **funkce**  . Funkce není datový prvek, protože nemůžeme za běhu programu měnit nebo předávat její obsah. Používáme ji v programu jako běžný příkaz. Druhý prvek, **skupina**  , neobsahuje nic a slouží jen ke shromáždění jiných prvků do skupinky kvůli zpřehlednění programu.

Všimněte si na panelu nástrojů tlačítek **Zvětšit**  a **Zmenšit**  . Tlačítka mění velikost zobrazení pro aktivní (vybrané) okno. Nejčastější použití tlačítek je v grafickém editoru pro zvětšení či zmenšení zobrazení obrázku. Uplatňují se také u oken se stromovou strukturou, jako je například okno **Společné proměnné a funkce**. Nejprve okno vyberte kliknutím do plochy okna nebo raději na lištu okna. Vybrání okna je indikováno zvýrazněním lišty okna (obvykle tmavě modrá barva). Poté můžete tlačítky přepínat ikonky v okně na poloviční nebo normální velikost.

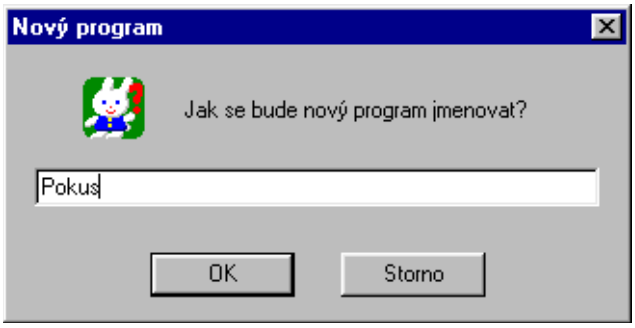
Jestli si chcete ještě pohrát s ukázkovými programy, můžete v nich překreslovat obrázky. Změněný program spustíte tlačítkem **Start**  (první tlačítko zleva na panelu nástrojů). Změny navrátíte tlačítkem **Zpět**  (čtvrté tlačítko zleva). Nebojte se poškození ukázkových programů. Když bude třeba, můžete program jednoduše zrušit. Zruší se pouze vaše upravená verze a na jejím místě se opět objeví původní ukázkový program. Až vás překreslování vzorových programů přestane bavit, zavřete editor programů tlačítkem **Zavřít**  (druhé tlačítko zleva) a začneme konečně tvořit.

Užitečná rada: Potřebujete-li vědět něco více o některém prvku, vyberte jej kliknutím myší na jeho ikonu a stiskněte klávesu **F1**. Zobrazí se podrobná nápověda k prvku.

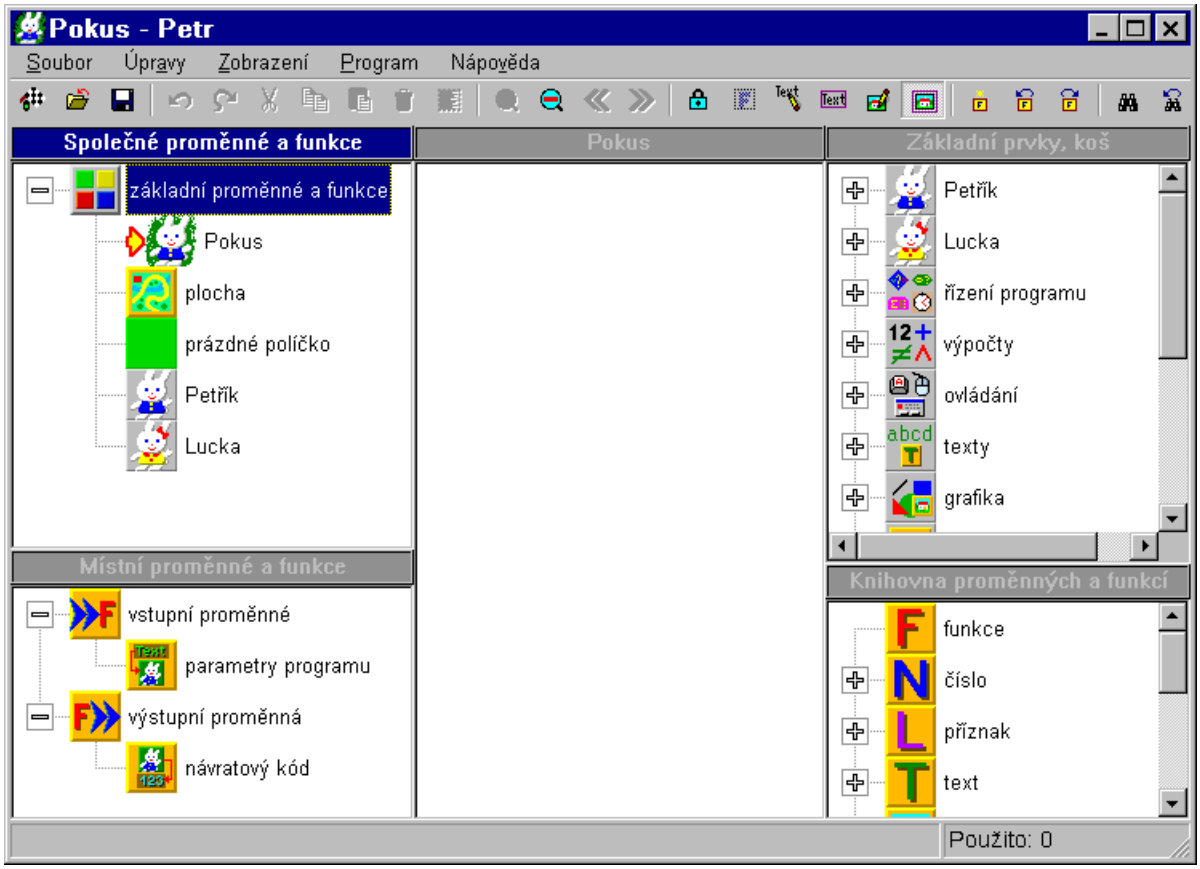
Ještě jedna rada: Někdy vám při ukládání či startu programu editor Petra může ohlásit chybu ukládání s podezřením, že program je spuštěn. Spuštěný program je nejčastější příčinou potíží. Pokud program neukončíte a přepnete se do editoru Petra, například kliknutím na okno Petra, ukryje se program pod Petrovo okno. U spuštěného programu nemůžete ukládat provedené změny, dokud program neukončíte. Proto vždy program nejdříve ukončete a až potom upravujte.

4 První kroky

Začněme s tvorbou vlastního programu. Ze všeho nejdříve musíme svůj nový program „založit“, to znamená vytvořit nový, prázdný, program. V **okně programů** (okno po startu Petra) klikněte levým tlačítkem myši na třetí tlačítko zleva . Když před kliknutím podržíte nad tlačítkem chvíli kurzor myši, objeví se malé okénko s textem **Nový** a v rychlé nápovědě na spodním okraji okna uvidíte text **Vytvoří nový program**. Po kliknutí na tlačítko budete vyzváni k zadání jména nového programu. Napište **Pokus** a stiskněte klávesu **Enter**.



Petr vytvoří nový prázdný program s názvem **Pokus** a otevře ho k editaci. Uvidíte plochu **editoru programu** tak, jak jsme si ji popsali v předešlé kapitole, jen editační okno bude zatím prázdné.



V okně **Společných proměnných a funkcí** (okno vlevo nahoře) je u nového programu připravena skupina **základní proměnné a funkce** s několika základními prvky. Prvky můžete měnit, ale nepodaří se vám je zrušit. Postupně si je všechny popíšeme.

Prvnímu z prvků  říkáme **hlavní funkce programu**. Obsahuje vlastně samotný program a u nového programu je prázdná. Text vedle ikony je **jméno hlavní funkce**.

Všimněte si šipky nalevo od ikonky hlavní funkce:  . Šipka označuje editovaný prvek, tj. prvek zobrazený v editačním okně. Editaci prvků vyvoláváme dvojitým kliknutím levým tlačítkem myši na ikonu prvku.

Hlavní funkce programu má dvě zajímavé vlastnosti: 1) Ikona hlavní funkce je současně **hlavní ikonou** programu. Je to ikona, kterou vidíte v okně programů Petra a také např. v Průzkumníkovi. 2) Jméno hlavní funkce se zobrazuje v titulku okna spuštěného programu. U nového programu je jméno hlavní funkce přednastaveno na jméno programu (v našem případě **Pokus**).

Ikonu hlavní funkce či jiného prvku změníte tlačítkem **Ikona**  . Jméno hlavní funkce i všech ostatních prvků můžete měnit tlačítkem **Popis**  . Prvek před úpravou vyberte jedním kliknutím myší na jeho ikonu. Modifikaci popisného textu můžete také zahájit stiskem kombinace kláves **Alt+Enter** nebo kliknutím levým tlačítkem myši na text vybraného prvku. Na místě textu prvku se objeví rámeček s blikajícím kurzorem. Po změně textu stiskněte klávesu **Enter**. Klávesa **Esc** navrátí text do původního stavu. Při změně textů u více prvků najednou můžete editační rámeček posouvat klávesovými šipkami nahoru a dolů a klávesu **Enter** stisknout až u posledního prvku.

Prvek pod hlavní funkcí programu je označen textem **plocha**  . Když na jeho ikonu dvakrát kliknete, objeví se v editačním okně zelená plocha rozdělená modrými čarami. Síť modrých čar slouží pouze k orientaci na ploše, a proto ji teď vypneme tlačítkem **Rastr**  . Zůstane plocha rozdělená na zelená políčka. Plocha ve skupině **základních proměnných a funkcí** představuje (oproti jiným plochám) současně plochu okna spuštěného vytvářeného programu, říkáme jí **hlavní plocha**.

Obsahy políček plochy nazýváme **předměty**, což jsou v podstatě obrázky o rozměru 32x32 bodů. Změnou předmětů v políčkách hlavní plochy můžeme měnit zobrazovaný obsah plochy, a tak provádět nejrůznější animace.

Pod prvkem hlavní plochy je prvek označený **prázdné políčko**  . Je to předmět, kterým je vyplněna každá nová plocha. Vyvolejte dvojitým kliknutím myši editaci prázdného políčka. Něco do políčka nakreslete a přejděte do editace hlavní plochy. Všimněte si, že se změnila všechna políčka na ploše.

Poslední dva prvky nesou označení **Petřík**  a **Lucka**  . Dvakrát klikněte na ikonku Petříka. Objeví se několik obrázků Petříka v nejrůznějších pozicích a směrech. Přesněji řečeno čtyři řady po pěti obrázcích. Takovému prvku říkáme **sprajt**. Sprajt je nějaký pohybující se animovaný objekt, třeba postava králíka, který můžeme přemisťovat po ploše okna. Později se naučíme měnit podobu Petříka a Lucky změnou sprajtu.

Teď se podívejme na okno vpravo nahoře. Nese označení **Základní prvky, koš** a obsahuje všechny příkazy a funkce vývojového prostředí Petra, které budeme potřebovat k vytváření programů. Doplňkovou funkcí okna je koš. Sem můžeme odhazovat všechny nepotřebné prvky.

Přikročme k „napsání“ našeho prvního příkazu programu. Dvojklikem na ikonku hlavní funkce zahajte editaci hlavní funkce programu (ikonka označená textem **Pokus**). Rozevřete v **Základních prvcích** první ikonku shora s obrázkem Petříka  . Uvidíte příkazy používané k ovládání postavy Petříka.

Uchopte levým tlačítkem myši prvek **krok**  a přetáhněte ho do editačního okna. Přesněji řečeno to znamená: Stiskněte na ikoně prvku levé tlačítko myši a - aniž tlačítko pustíte - přesuňte myš nad editační okno. Povšimněte si, že pod kurzorem myši držíte poloprůhledný obrázek přetahovaného prvku spolu s jeho textem. Na spodním okraji kurzoru myši se přitom objevily dva překrývající se bílé obdélníčky indikující kopírování prvku, tj. po přetažení zůstane původní prvek na svém místě a na novém místě se objeví kopie prvku. Zkuste myší zabrousit za okraj editačního okna. Spodní okraj myši se překryje černým přeškrtnutým kolečkem jako indikace, že na tomto místě není položení prvku povoleno. Vraťte se do editačního okna a pusťte tlačítko myši, prvek se objeví v levém horním rohu okna.

Tím jsme vytvořili první příkaz. To nás ale neuspokojí a vytvoříme si takové příkazy hned čtyři. Nebudeme je teď už tahat ze **Základních prvků**, ale „rozmnožíme“ si třikrát příkaz v okně již položený. Prvky budeme přetahovat podobně jako z okna **Základních prvků** s tím rozdílem, že k tomu použijeme pravé tlačítko myši, a budeme je klást postupně shora dolů pod sebe. Pravé tlačítko myši slouží vždy ke kopírování prvků. Levým tlačítkem myši prvky přesouváme na nové místo. Mezi okny editoru lze prvky pouze kopírovat nezávisle na použitém tlačítku myši.

Máme-li pod sebou naskládány čtyři prvky **krok**  , je náš první program hotov. Co nám ještě zbývá? Program spustit. K tomu slouží tlačítko **Start**  . Je v panelu nástrojů na první pozici zleva. Po stisku tlačítka se objeví okno spuštěného programu se zeleným povrchem. Vlevo dole je králík Petřík. Přecupitá několik kroků směrem doprava a program se ukončí. Skvělý program, že?

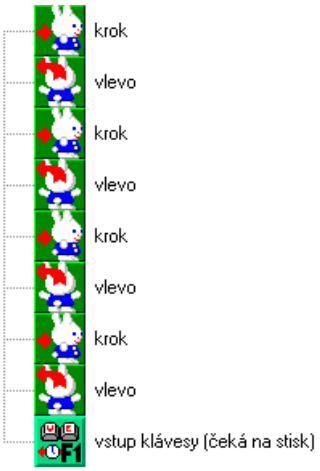
K dokonalosti mu chybí ještě maličkost. Nelíbí se nám, že program hned skončí. Doplníme proto příkaz pro čekání na stisk klávesy. Naleznete ho ve skupince **ovládání**, podskupince **klávesnice** a je označen **vstup klávesy (čeká na stisk)**. Prvek přidejte na konec za všechny příkazy. Celý program bude vypadat takto:

Program spusťte. Králík opět popojde doprava, tam se zastaví a čeká. Po stisku jakékoliv klávesy se program ukončí.

Náš první program je hotov. Moc toho zatím neumí, ale je to náš vlastní program. A co je na něm nejzajímavější - náš malý prográmek je plnohodnotnou **32-bitovou multitaskingovou aplikací Windows**. Co to znamená? Program můžete spustit kolikrát chcete a všechny spuštěné programy poběží současně. Ti zkušenější mohou pomocí Průzkumníka nahlédnout do pracovní složky programů Petra (zpravidla **C:\Dokumenty\Petr\Program**). Naleznou tam program se jménem **Pokus.exe**, který mohou dát svým přátelům, nebo pro něj vytvořit na základní ploše Windows spouštěcí ikonu. Vytvořený program **není** na prostředí Petra nijak závislý a můžete ho používat stejně, jako každý jiný program systému Windows.

Po našich prvních úspěšných krocích, vlastně Petříkových krocích („*Je to malý krůček pro králíka, ale velký skok pro lidstvo*“ - nepřipomíná vám to něco?), se pustíme s chutí do dalších pokusů.

Vedle příkazu pro krok jste si možná již všimli i dalších příkazů pro ovládání Petříka, jako je otáčení vlevo, vpravo a vzad. Zkuste náš prográmek upravit tak, aby Petřík obešel kolečko a vrátil se zpět na původní místo. Výsledek by měl vypadat takto:



Funguje? Blahopřeji vám pane, právě jste se stal programátorem.

5 Petříkova zahrádka

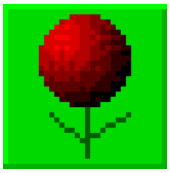
Po prvních krůčcích naučíme Petříka, jak si vysázet svou vlastní zahrádku. Budeme přitom pokračovat v úpravách našeho programu **Pokus**. Doplníme pouze obrázek květiny a řekneme Petříkovi, aby při své obchůzce květiny sázel.

Nejprve si připravíme obrázek květiny. Použijeme k tomu prázdné políčko, do kterého květinu dokreslíme. Podívejte se do okna s názvem **Společné proměnné a funkce**. Ve skupince **základní proměnné a funkce** vidíte ikonku **prázdné políčko**  . Uchopte ji pravým tlačítkem myši a přetáhněte dolů pod všechny prvky skupinky.

Po přetažení se vytvořila kopie prvku, jméno se změnilo na **prázdné políčko 2**. Jedním kliknutím levým tlačítkem myši na ikonku prvek vyberte (označí se obdélníkem). Klikněte na text u ikonky, opět levým tlačítkem myši. Pod textem se objeví rámeček a blikající kurzor. Napište jako nové jméno prvku text **Květina** a stiskněte klávesu **Enter**. Tím jsme vytvořili (nadeklarovali) nový předmět s názvem **Květina**.

Teď nakreslíme obrázek květiny. Dvakrát klikněte levým tlačítkem myši na ikonku předmětu **Květina**. V editačním okně se objeví zvětšený obrázek prázdného políčka. Můžete začít kreslit květinu, zájemcům poradím postup.

V panelu nástrojů vidíte rozbalovací seznam funkcí grafického editoru. Další rozbalovací seznam je pro volbu tloušťky čar. Zvolte funkci kreslení **koule**  . Dole v editačním okně naleznete okénka pro volbu barvy. Klikněte levým tlačítkem myši na červenou barvu (první shora). Na horním okraji obrázku, přibližně uprostřed, stiskněte levé tlačítko myši a táhněte myší do středu obrázku. Tam tlačítko myši pusťte. Vzniklá červená koule bude květem. Levým tlačítkem myši zvolte tmavě zelenou barvu (druhou odspodu) a v seznamu funkcí vyberte kreslení **čáry**  . Natáhněte jednu čáru od koule směrem dolů a potom ještě po stranách dvě čáry jako listy. Výsledek by mohl vypadat asi tak nějak (že by jiřina?):



Jste-li s kreslením hotovi, vraťte se k editaci hlavní funkce programu dvojklikem na ikonku hlavní funkce (stále ještě by měla mít jméno **Pokus**). Pro navracení k dříve editovaným prvkům můžete také používat tlačítka **Předešlá editace**  a **Následující editace**  , sloužící k listování v historii editovaných prvků.

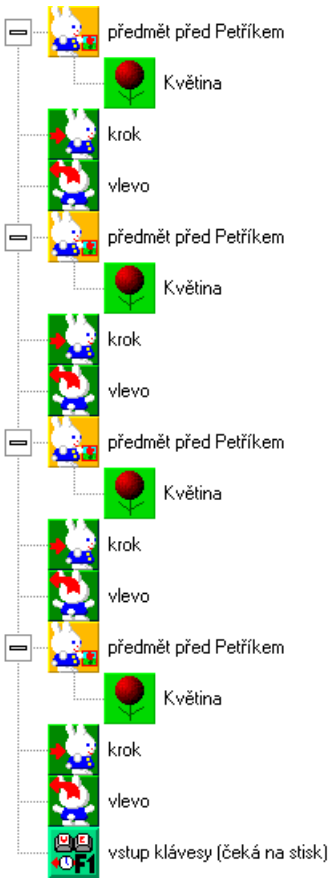
Začneme sestavovat nový program. V předešlé kapitole jsme program zanechali ve stavu, kdy Petřík popojde o krok, otočí se vlevo, oba příkazy zopakuje ještě třikrát, nakonec se zastaví a čeká na stisk klávesy.

Podívejte se znovu do skupiny příkazů pro Petříka. Připomínám, že ji najdete v pravém horním okně **Základní prvky, koš** a že je označena textem **Petřík**. Šestým prvkem shora je **předmět před Petříkem**  . Uchopte ho a přetáhněte úplně na začátek programu, před první příkaz. Po položení prvku se před ikonkou prvku objeví značka

rovnítka . To nás editor informuje, že prvek ještě potřebuje přidat nějaký parametr. Tímto parametrem bude námi vytvořená květina.

Uchopte myší v okně **Společné proměnné a funkce** předmět **Květina** a přetáhněte ho k prvku **předmět před Petříkem** - asi tak, aby levý horní roh přetahovaného prvku překrýval pravý dolní roh cílového prvku. Při přetahování si můžete všimnout, že jste-li poblíž prvku **předmět před Petříkem**, na textu prvku se objeví výběrový rámeček. Tím nám dává cílový prvek najevo ochotu přetahovaný prvek přijmout. Zásadní vlastností prvků je, že lze k sobě „přilepit“ pouze smysluplné kombinace prvků. Tak se nemůže stát, že sestavíte nesmysl (automatická syntaxe).

Po připojení přetahovaného předmětu květiny zmizela značka rovnítka před ikonkou. Prvek **předmět před Petříkem** dostal svůj parametr a teď je již spokojený. Vznikla nám malá větev o dvou prvcích, mající význam položení květiny před Petříka. Větev rozkopírujte ještě třikrát před ostatní příkazy kroků Petříka a to tak, že ji přetahujte pravým tlačítkem myši uchopením za prvek **předmět před Petříkem**. Výsledkem by měl být tento program:



Program spusťte. Petřík nasází 4 květiny a zůstane stát na výchozí pozici. Sice nám přitom šlape po květinách, ale to mu musíme odpustit. On se zahrádkaření teprve učí.

6 Zahrádka s opakováním

Představte si, že Petřík by si chtěl vysázet zahrádku o velikosti 15 políček na šířku i výšku. Mohli bychom sice vícekrát rozkopírovat příkazy z předešlé kapitoly, ale za něco takového by se musel snad každý programátor začervenat studem.

Naučíme se proto používat **cyklus**. Cyklus zajišťuje opakované provádění příkazů. Použijeme cyklus s pevným počtem opakování. Je označen **opakování příkazů s počtem** a najdete ho v okně **Základní prvky, koš** ve skupince **řízení programu**. Přetáhněte jej na úplný začátek programu z minulé kapitoly (před všechny příkazy).

Po položení prvku cyklu si všimněte, že si s sebou přinesl ještě další dva prvky. Tyto dva prvky jsou nedílnou součástí cyklu a nelze je zrušit ani nikam přesunout. První prvek je označen **pro počet opakování** a je určen k zadání počtu, kolikrát se mají příkazy cyklu provést. Druhý prvek nese označení **opakuj příkazy**, sem klademe příkazy cyklu určené k opakování. Říkáme mu také **tělo cyklu**.

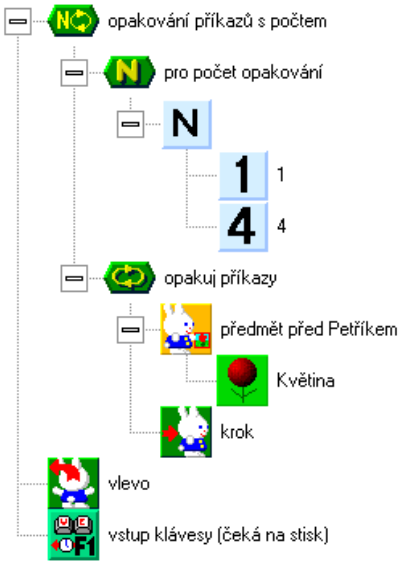
Nejdříve určíme příkazy, které bude cyklus provádět. Z předešlé kapitoly máme připraven příkaz pro krok a příkaz pro položení květiny před Petříka. Tyto dva příkazy přetáhněte do těla cyklu, tedy do prvku **opakuj příkazy**. Abych upřesnil, příkaz pro položení květiny před Petříka jsme si dříve vytvořili ze dvou prvků, z prvku **předmět před Petříkem** a k němu připojeného prvku **Květina**.

Nyní určíme počet opakování, kolikrát budou příkazy v cyklu provedeny. K tomu potřebujeme prvek **číselná konstanta**, který najdete ve skupince **výpočty**. Přetáhněte ho do prvku **pro počet opakování**. Pokud prvek číselné konstanty v okně **Základních prvků** rozvinete, najdete v něm prvky číslic **0** až **9**. Přetáhněte číslice do číselné konstanty v programu tak, aby vzniklo číslo počtu opakování příkazů. Sestavené číslo v číselné konstantě přitom čteme shora dolů. Jako počet opakování zvolíme číslo o jedno menší, než je šířka zahrádky. Nastavíme tedy číslo **14** - konstanta bude obsahovat číslici **1**, pod ní bude číslice **4**.

Prozradím vám ještě jeden způsob zadávání čísla. Neobsahuje-li číselná konstanta uvnitř žádný prvek číslice, uplatní se číslo napsané v popisném textu číselné konstanty. Za číslem může být uvedena jakákoliv poznámka. Takže v našem případě by stačilo napsat k prvku text **14 - počet kroků**. Předešlý způsob je však poněkud názornější.

Teď již vyhoďte do koše (okno vpravo nahoře) všechny ostatní prvky, které zůstaly mimo cyklus, kromě jednoho příkazu pro otočení vlevo a příkazu pro čekání na stisk klávesy. Vyhazovat můžete myší, ale existuje i rychlejší způsob pomocí klávesnice. Označte kliknutím levým tlačítkem myši na ikonu první z rušených prvků a potom tiskněte klávesu **Delete**, dokud nebudou všechny nepotřebné prvky odstraněny. Moc to ale nepřežeňte, ať nemusíte používat obnovovací tlačítko **Zpět**.

Po všech úpravách by vám měl vzniknout program vypadající takto:



Všimněte si, jak lze program díky popisům prvků číst celkem přirozeným způsobem: „*Pro počet opakování 14 opakuj příkazy: polož jako předmět před Petříkem Květinu, potom udělej krok.*“

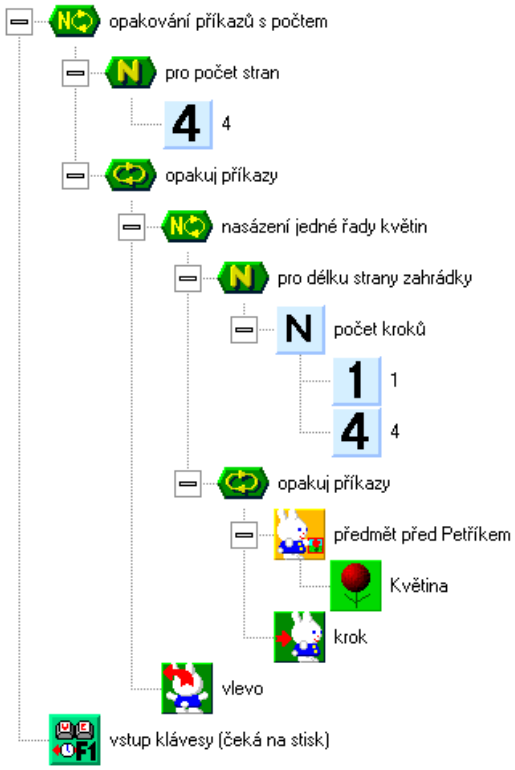
Program spusťte. Petřík se vydá k pravému okraji okna a cestou sází květiny. Když nasází květiny téměř po celém spodním okraji okna, otočí se nahoru a čeká. Zapomněl květinu na začátku řady vlevo, ale příště se určitě polepší.

Teď bychom mohli náš cyklus zkopírovat ještě třikrát; potřebujeme totiž celkem čtyři strany. Ale protože jsme správní programátoři, použijeme k tomu opět cyklus. Cyklus je totiž v podstatě běžný příkaz a tak ho můžeme vložit do jiného cyklu.

Přetáhneme si z okna **Základních prvků** další prvek **opakování příkazů s počtem** a vložíme ho úplně na začátek programu. Jako počet opakování uvedeme číslo **4**. Pro číselné konstanty s jednou číslicí můžeme používat přímo samotné číslice, které najdeme u prvku číselné konstanty. Takže do parametru cyklu **pro počet opakování** přetáhneme prvek číslice **4** 4.

Do těla cyklu **opakuj příkazy** přetáhneme dříve vytvořený cyklus pro vysázení jedné strany zahrádky a pod něj příkaz pro otočení vlevo. Příkaz pro čekání na stisk klávesy necháme úplně na konci celého programu. Dobrý programátor si ještě svůj výtvor okomentuje poznámkami, aby mu byla činnost programu jasná i na druhý den. A nejen jemu, ale i všem ostatním, kteří program někdy uvidí.

Zde vidíte výsledek naší usilovné práce:



Program spusťte. Petřík se rozběhne kolem své nové zahrádky a pilně sází květiny. Tentokrát již nasázel všechny a na žádnou nezapomněl. Líbí se vám jeho zahrádka?

Zvýšíte-li počet kroků ve vnitřním cyklu třeba na **44**, můžete pozorovat, že Petřík vysází květiny teď již skutečně po celém obvodu plochy, a ani mu nevadí, že na konci cesty vždy narazí hlavou do zdi. Pokud nemůže nějakou operaci provést, tak ji prostě neprovede a nic se nestane.

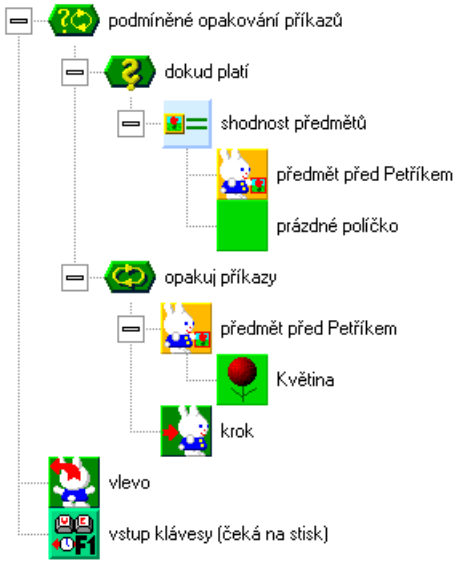
Chcete-li ještě experimentovat, zkuste zvětšit číslo v počtu opakování hlavního (vnějšího) cyklu. Petřík bude pobíhat kolem okraje okna neustále. Při velkém počtu opakování se možná ani konce nedočkáte a budete muset program přerušit kliknutím na tlačítko s křížkem ⊠ (v pravém horním rohu okna) nebo stiskem kláves **Alt+F4**.

7 Zahrádka s podmínkou

Prvek **předmět před Petříkem** můžeme použít nejen k položení předmětu před Petříka, ale také ke zjištění, jaký předmět před Petříkem leží. Testování předmětu využijeme v jiné metodě sázení zahrádky, založené na podmíněném cyklu.

Najděte v okně **Základní prvky, koš** ve skupince **řízení programu** prvek **podmíněné opakování příkazů**. Přetáhněte ho na začátek programu z minulé kapitoly (před všechny příkazy). Do těla cyklu **opakuj příkazy** přesuňte příkazy pro krok a položení květiny před Petříka. Za cyklem ponechejte prvky otočení vlevo a čekání na stisk klávesy. Ostatní prvky můžete vyhodit.

Prvek cyklu **dokud platí** slouží k testování podmínky, určující jak dlouho se mají příkazy v cyklu opakovat. My do podmínky cyklu uvedeme test, zjišťující zda je před Petříkem prázdné políčko. K sestavení podmínky použijeme prvek **shodnost předmětů**. Najdete ho v okně **Základní prvky, koš** ve skupince **plocha**. Prvek přetáhněte do prvku cyklu **dokud platí**. Do prvku **shodnost předmětů** vložíme dva prvky, které chceme porovnávat. Prvním z nich bude **předmět před Petříkem**, druhým **prázdné políčko** (z okna **Společné proměnné a funkce**). Zde je výsledek:



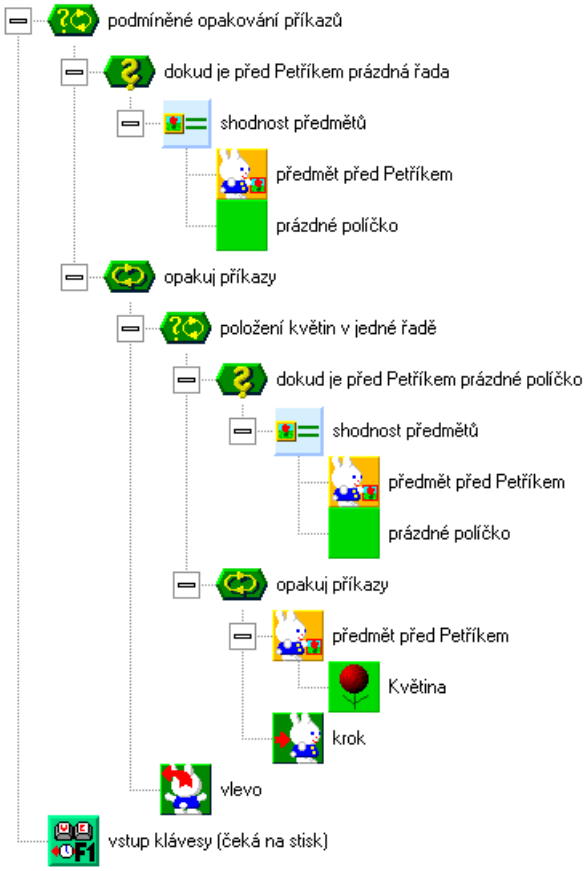
Zkuste program spustit. Petřík nasází květiny až k pravému okraji okna, otočí se nahoru a čeká na stisk klávesy. Možná vám činnost programu není zcela jasná, podívejme se proto na program podrobněji.

Jak funguje podmíněný cyklus? Popisy k prvkům cyklu nám říkají: „*dokud platí (něco) opakuj příkazy (něco)*“. Při detailnějším pohledu to znamená následující. Cyklus na svém začátku vyhodnotí podmínku, zda je splněna. Pokud ano, provede příkazy uvedené v těle cyklu. Potom vše opakuje od začátku. Opět vyhodnotí podmínku a při jejím splnění znovu provede příkazy v cyklu. Není-li podmínka splněna, nic se neprovede, cyklus se ukončí a program pokračuje dalšími příkazy za cyklem.

Náš prográmek bychom mohli popsat takto: Na začátku se cyklus ptá testovací funkce vyhodnocující podmínku: „*Je tvá podmínka splněna?*“ Testovací funkcí je zde funkce pro porovnání předmětů. Ta zjišťuje: „*Je před Petříkem prázdné políčko?*“ Pokud ano, odpoví cyklu: „*Ano, podmínka je splněna.*“ Cyklus v tom případě provede příkazy v těle cyklu - Petřík před sebe položí květinu a popojde o krok vpřed. To se opakuje až k okraji plochy. U okraje plochy testovací funkce zjistí, že před Petříkem již není prázdné políčko a sdělí to cyklu. Cyklus dále nepokračuje a ukončí se. Následuje otočení Petříka vlevo a zastavení programu s čekáním na klávesu.

Po položení květin v jedné řadě zůstane Petřík otočen vlevo, směrem na další řadu. Otestujeme, zda je před ním prázdné políčko, a pokud ano, přikážeme mu vysázet další řadu květin. Když se dostane opět na výchozí pozici, nebude již před ním prázdné políčko, ale vysázená květina, a tak se zastaví.

Takže teď vezmeme nový cyklus **podmíněné opakování příkazů**  . Dáme jej na začátek programu a do jeho těla přesuneme (levým tlačítkem myši) dříve vytvořený cyklus spolu s příkazem pro otočení vlevo. Na konci programu zůstane příkaz pro čekání na stisk klávesy. Do podmínky vnějšího cyklu zkopírujte (pravým tlačítkem myši) podmínku testující prázdné políčko před Petříkem (uchopením za prvek **shodnost předmětů** ). Zde je výsledek:



Program je hotov, zbývá jen ho vyzkoušet.

8 Petřík chodí po značkách

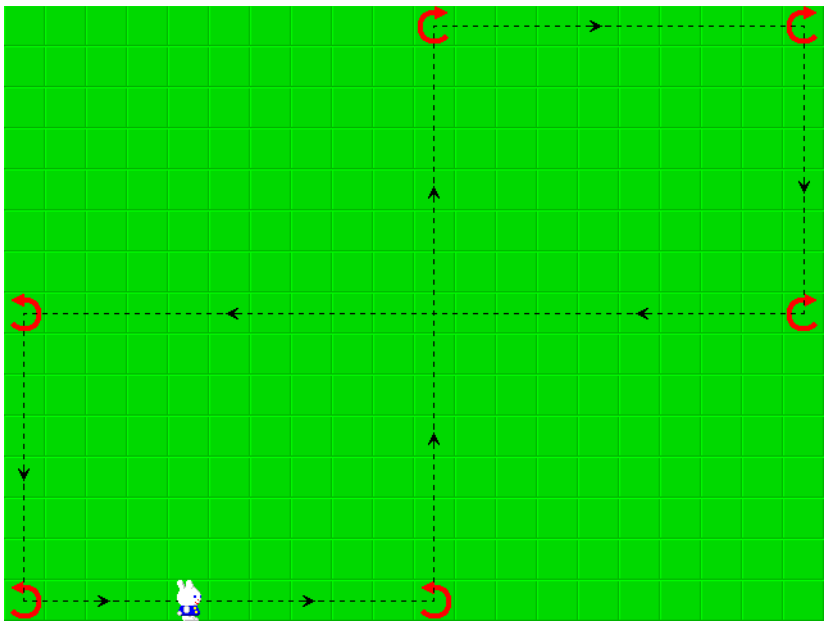
V dalším programu se naučíme používat podmíněné provádění příkazů. Petříka necháme procházet se po ploše okna, na které rozmístíme různé značky. Podle značek se bude rozhodovat, kterým směrem se vydá dále.

Ze všeho nejdříve založíme nový program. Jak si jistě vzpomínáte z kapitoly 4, nový program vytvoříme v okně programů pomocí tlačítka **Nový** . Pojmenujeme si ho třeba **Značky**.

V programu budeme potřebovat dva předměty, které použijeme jako značky. V okně **Společné proměnné a funkce** zkopírujte pravým tlačítkem myši prvek **prázdné políčko** na dva nové předměty. Pojmenujte je **Doleva** a **Doprava**. Pro připomenutí: přejmenování provedete například tak, že kliknutím myší na ikonu prvku prvek vyberete a stiskem kláves **Alt+Enter** zahájíte editaci jména prvku.

Vyvolejte editaci nově vytvořených předmětů dvojitým kliknutím levým tlačítkem myši na ikonky předmětů. Do prvku **Doleva** nakreslete zatočenou šipku vlevo, do prvku **Doprava** zatočenou šipku vpravo, například takto: a .

Připravíme si plochu okna. Dvakrát klikněte na prvek označený **plocha** . V editačním okně se objeví plocha programu. Vytvoříme cestu pro Petříka. Na plochu naskládáme předměty **Doleva** a **Doprava** tak, aby Petřík, až se bude podle nich řídit, mohl chodit po uzavřené dráze. Nezapomeňte, že Petřík vychází z levého dolního rohu směrem doprava. Předměty na ploše budeme rozmisťovat tak, že nejdříve přetáhneme oba prvky z okna **Společné proměnné a funkce** někam na plochu, kde je položíme. Potom levým tlačítkem myši předměty na ploše přemisťujeme, pravým tlačítkem je kopírujeme. Nepotřebný předmět zrušíte odhozením mimo plochu. Plocha může vypadat například takto (cesta Petříka je vyznačena čárkovanou čarou se šipkami):



Po přípravě plochy programu se pustíme do konstrukce programu. Tentokrát začneme z opačného konce - od hlavního cyklu. Petřík má za úkol neustále chodit dokola po značkách, takže je zřejmé, že základem programu bude nekonečná smyčka. Připravíme si proto do programu prvek **podmíněné opakování příkazů**  . Ve skutečnosti neuděláme smyčku nekonečnou, ale umožníme přerušit program stiskem klávesy **Esc**. K tomu si musíme nejprve povědět něco o klávesnici.

Každá klávesa na klávesnici při svém stisku vyšle do počítače svůj číselný kód, který je jakýmsi pořadovým číslem klávesy na klávesnici. Počítač tento číselný kód převede na znak a ten může předat programu například v podobě písmene či číslice. My máme v programu přístupné nejen znaky, ale také číselné kódy kláves. Funkci pro vstup znaků používáme k zadávání textů z klávesnice tak, jak to známe z běžného psaní textů (např. při držení **Shift** se generují velká písmena). Funkce pro vstup kláves slouží k řízení programů a her (kódy kláves generují i řídicí klávesy jako je **Ctrl**, zatímco znaky těmito klávesami generovány nejsou).

Znak nebo kód klávesy se při stisku klávesy uloží do zásobníku. V okamžiku stisku klávesy totiž program nemusí být připraven klávesu převzít. Až bude mít čas, převezme klávesu ze zásobníku, přitom dojde ke zrušení klávesy v zásobníku. Při používání funkcí vstupu z klávesnice je proto potřeba pamatovat na to, že načtením klávesy nebo znaku z klávesnice se klávesa či znak zruší a při příštím čtení budou funkce navracet již něco jiného.

Znaky a klávesy mají samostatné zásobníky, na sobě navzájem nezávislé. Jsou to jakoby oddělené datové proudy. Jedním kanálem proudí znaky, druhým kódy kláves.

Teď již můžeme připravit podmínku pro přerušení hlavní smyčky programu klávesou **Esc**. Použijeme funkci vstupu kódu klávesy z klávesnice. Víme, že vrací číselný kód, proto potřebujeme funkci k porovnání čísel. Porovnávací funkce pro čísla nalezneme ve skupince **výpočty**  , podskupince **porovnání**  . Použijeme funkci **není rovno**  . Přetáhněte ji do podmínky cyklu **dokud platí**  .

Prvním prvkem, který budeme porovnávat, je **vstup klávesy (nečeká na stisk)**  . Přetáhneme ho do porovnávací funkce ze skupiny **ovládání**  , podskupiny **klávesnice**  . S podobným prvkem, s čekáním na stisk klávesy, jsme se již seznámili v předešlých kapitolách. Rozdíl mezi prvky spočívá v tom, že tento prvek na stisk klávesy nečeká. Není-li žádný kód klávesy připraven, vrací kód nestisknuté klávesy (je to číslo **0**, ale my se konkrétní hodnotou nemusíme zatěžovat, protože máme k dispozici symbolický kód **žádná klávesa nestisknuta** ).

Druhým porovnávaným prvkem je klávesa **Esc**  . Najdeme ji ve stejné skupince jako funkci vstupu klávesy, ale ještě se musíme vnořit trochu hlouběji, a to do podskupinek **klávesy**  a **řídicí klávesy**  . Je to číselná konstanta s hodnotou číselného kódu klávesy **Esc**, abychom nemuseli kód klávesy znát. Položte prvek do porovnávací funkce pod prvek funkce vstupu klávesy.

Co náš cyklus teď dělá? Přečtěte si poznámky u prvků: „*Dokud platí, že není roven vstup klávesy klávese Esc, opakuj příkazy (něco).*“ Trochu krkolomně řečeno, ale snad je význam jasný. Cyklus se bude opakovat, dokud nepřijde z klávesnice klávesa **Esc**.

Naplníme tělo cyklu **opakuj příkazy**. Petřík se bude při své pochůzce rozhodovat o dalším kroku podle políčka, které leží před ním. Pro rozhodování použijeme nový prvek, **podmíněné provedení příkazů**. Najdete ho ve skupince **řízení programu**. Prvek přetáhněte do těla cyklu **opakuj příkazy**.

Po položení prvku podmíněného provedení příkazů si můžete všimnout, že prvek v sobě obsahuje další tři prvky. Prvním prvkem je **platí-li, že**. Je to test podmínky. S podobným prvkem jsme se již setkali u podmíněného cyklu. Prvek provádí test nějaké podmínky a je-li splněna, vykonají se příkazy v prvku **potom proveď**. Není-li podmínka splněna, provedou se příkazy v prvku **jinak proveď**.

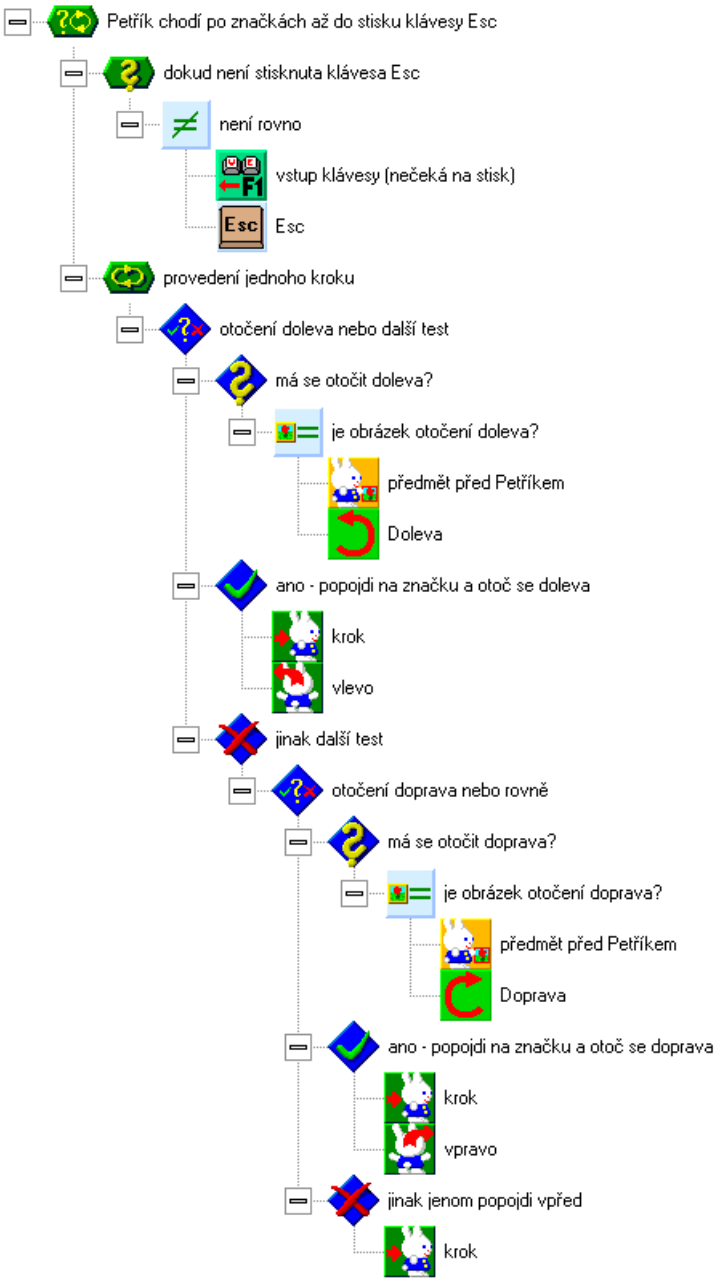
V testu podmínky budeme testovat, zda před Petříkem leží předmět pro otočení doleva. Podobný test známe z předešlé kapitoly, a tak je vám zřejmě již jasné, že použijeme prvky **shodnost předmětů**, **předmět před Petříkem** a **Doleva**.

Do první větve (**potom proveď**) doplníme dva příkazy, **krok** a **vlevo**. Naše konstrukce podmíněného příkazu má teď tento význam: „*Leží-li před Petříkem předmět pro otočení doleva, udělá krok a otočí se vlevo, jinak bude dělat něco jiného.*“

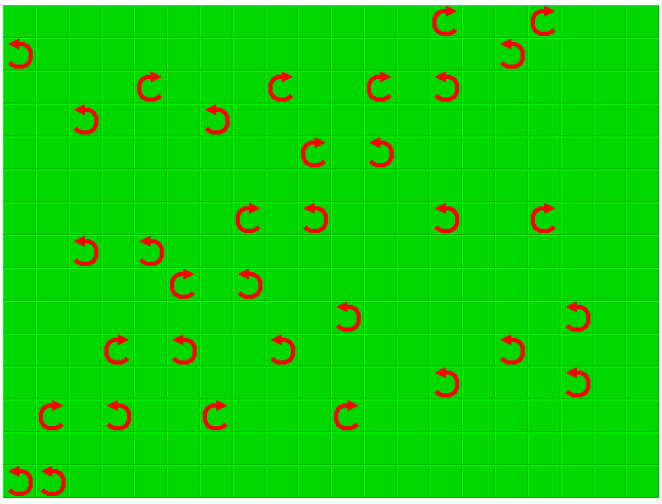
Nyní vytvoříme to „*něco jiného*“. Zkopírujte pravým tlačítkem myši celou konstrukci podmíněného příkazu ještě jednou o trochu níže. Nově vytvořený podmíněný příkaz znovu uchopte, tentokrát levým tlačítkem myši, a přetáhněte ho do druhé větve prvního podmíněného příkazu (větev **jinak proveď**). Tak jsme si zrychleně „předpřipravili“ část programu pro druhý případ, otočení doprava.

V novém podmíněném příkazu nahraďte prvek **Doleva** prvkem **Doprava**. Podobně nahraďte prvek **vlevo** prvkem **vpravo**. Teď už asi tušíte, co bude Petřík dělat, nebude-li před ním předmět pro otočení doleva. Zkusí, zda je před ním předmět přikazující otočení doprava. Pokud ano, udělá krok vpřed a otočí se vpravo. Jinak … co má vlastně dělat jinak? Nic jiného už od něj nechceme, takže stačí, když udělá **krok**. Prvek kroku zkopírujte pravým tlačítkem myši z vedlejší větve podmínky.

Vypadalo to složitě? Nebojte se, už jsme hotovi. Výsledný program vidíte na další stránce. Program spusťte. Je-li vše v pořádku, Petřík pobíhá mezi značkami a otáčí se na nich vlevo a vpravo. Chcete Petříka trochu zrychlit? Dvojklikem myší na prvku **Petřík** vyvolejte editaci sprajtu Petříka. Stiskněte tlačítko **Vlastnosti**. Objeví se okno pro nastavení vlastností sprajtu. Do pole **Fází na krok** napište číslo **4** a stiskněte klávesu **Enter**. Program opět spusťte. Petřík teď běhá jako blázen, ani se nezadýchá.



Zkuste Petříkovi připravit složitější dráhu. Tady je jeden příklad:



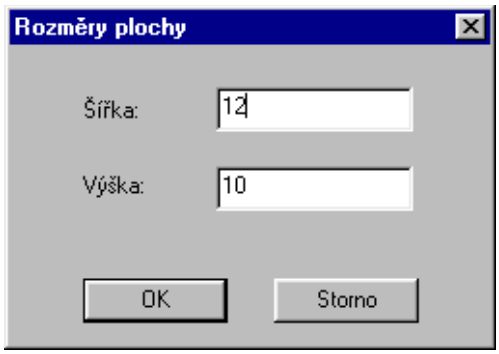
9 Petřík v bludišti

Petřík na svých toulkách přišel do velkého bludiště. Nemůže najít cestu ven, musíme mu pomoci.

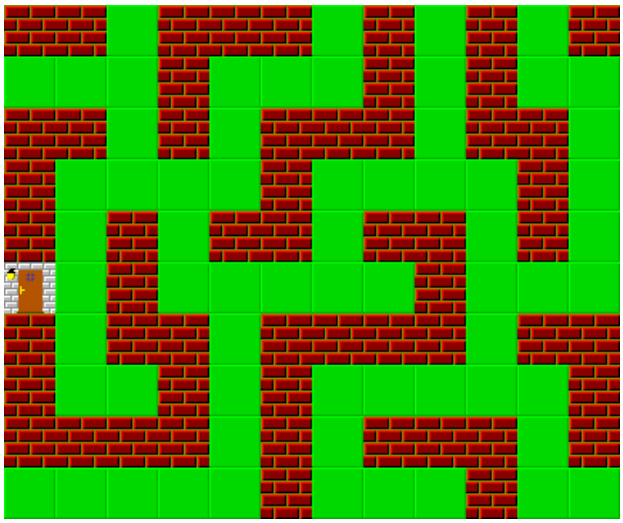
Začneme vytvořením nového programu. V okně programů stiskněte tlačítko **Nový**, jako jméno programu zadejte **Cesta**.

V okně **Společné proměnné a funkce** připravíme dva předměty pro bludiště - zeď a dveře. Mohli bychom si je namalovat, ale raději použijeme předměty již připravené v knihovně předmětů. Knihovna se nachází v okně vpravo dole, okno nese nadpis **Knihovna proměnných a funkcí**. Vidíme v něm datové prvky jako je číslo, předmět, obrázek. Rozviňte prvek **předmět** a ve skupince **[vzory]**, podskupince **Domek**, najděte prvky **Dveře** a **Zeď**. Přetáhněte je do okna **Společné proměnné a funkce**. Tak jsme snadno a rychle vytvořili dva nové předměty už pojmenované a nakreslené.

Teď vytvoříme plochu bludiště. Dvakrát klikněte na prvek **plocha**. Trochu si ji zmenšíme, ať nemusíme dělat příliš velké bludiště. Klikněte na tlačítko **Rozměry**. Objeví se okno pro zadání rozměrů plochy. Zadejte šířku plochy **12** a výšku **10** kroků.



Použijte předměty **Dveře** a **Zeď** a vytvořte bludiště podle obrázku:



Základem programu bude cyklus **podmíněné opakování příkazů**  . Chceme totiž, aby Petřík neustále opakoval své kroky po bludišti až do okamžiku, kdy najde dveře. V podmínce cyklu použijeme prvek **předmět na pozici s Petříkem**  . Je podobný prvku **předmět před Petříkem** s tím rozdílem, že se vztahuje k předmětu na políčku, na kterém stojí Petřík. Najdete ho v podskupince **Petřík - rozšíření**  .

Podmínku cyklu bychom mohli slovně vyjádřit: „*Dokud <u>není</u> na pozici s Petříkem předmět dveří, opakuj…*“. Slovo <u>není</u> nám dává najevo, že potřebujeme opak testu na shodnost předmětů. K tomuto účelu je určen prvek **neplatí-li, že**  . Prvku se říká **logická negace** a funguje tak, že převrátí výsledek porovnávací operace z **ano** na **ne** a z **ne** na **ano**. Najdeme ho ve skupince **výpočty**  , podskupince **logické operace**  .

Nejdříve tedy umístíme do podmínky cyklu **dokud platí**  prvek **neplatí-li, že**  . K němu připojíme prvek **shodnost předmětů**  a do něj vložíme prvky **předmět na pozici s Petříkem**  a **Dveře**  .

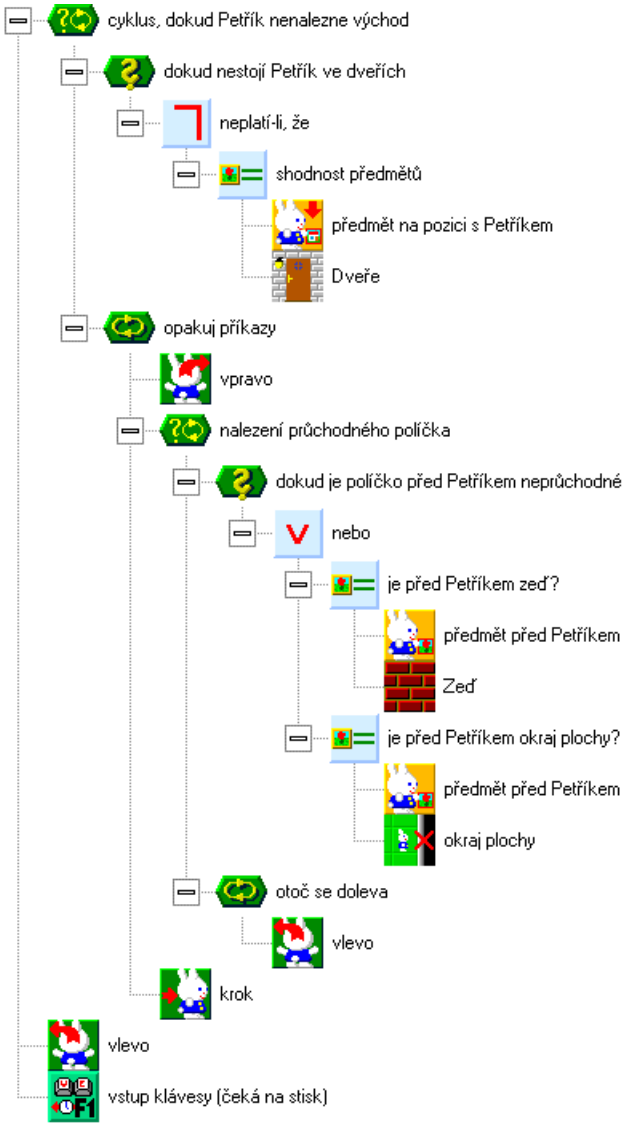
V těle cyklu chceme zajistit, aby Petřík procházel bludiště tak, že se bude neustále držet stěny po pravé straně. Na začátku cyklu necháme Petříka otočit vpravo příkazem **vpravo**  . Dalším příkazem bude cyklus **podmíněné opakování příkazů**  . Ten zajistí otáčení Petříka doleva, dokud před sebou nenajde průchodné políčko. Jako příkaz do cyklu proto umístíme prvek **vlevo**  .

V podmínce druhého (vnitřního) cyklu budeme testovat, zda je před Petříkem neprůchodné políčko a zda se tedy má Petřík ještě pootočit doleva. Prvním z neprůchodných políček bude **Zeď**  . Petřík nemůže projít ještě přes jednu překážku, a to přes okraj plochy okna. Okraj plochy okna můžeme testovat jedním zvláštním předmětem - **okraj plochy**  (je ve skupince **plocha** ). Zvláštní je v tom, že to není datová proměnná, ale konstanta, protože její obsah nemůžeme měnit. Její hodnotu lze uchovávat v proměnných a testovat, ale nelze ji zobrazit položením na plochu.

Máme tedy dva testy předmětů před Petříkem a potřebujeme vyjádřit formulaci „*Dokud je před Petříkem Zeď <u>nebo</u> je před Petříkem okraj plochy, opakuj vlevo*“. Slovo <u>nebo</u> je opět logický prvek - **nebo**  , říkáme mu **logický součet**. Najdeme ho ve skupince **výpočty**  , podskupince **logické operace**  . Vložíme prvek **nebo** do podmínky cyklu a přidáme dvě porovnání předmětů před Petříkem, se zdí a s okrajem.

Petřík je teď otočen směrem na volné políčko, takže nám zbývá doplnit poslední prvek do hlavního cyklu - **krok**  . Prvek zajistí popojítí Petříka vpřed na volné políčko.

Nakonec zajistíme, aby se Petřík po nalezení cíle na nás otočil a čekal na stisk klávesy. Za hlavní cyklus doplníme prvky **vlevo**  a **vstup klávesy (čeká na stisk)**  . Výsledný program vidíte na další stránce.



Program spusťte. Petřík probíhá celé bludiště, až nakonec dorazí k východu. Schválně jsme mu dali východ hned za roh po levé straně, ale protože to neví, snaží se najít východ vpravo a tak mu to dlouho trvá.

Prohoďme mezi sebou prvky **vlevo** a **vpravo**. Tím jsme změnili metodu hledání východu tak, že se Petřík bude držet levé stěny namísto pravé. V našem bludišti najde východ mnohem dříve.

10 Ovládání klávesnicí

Zůstaneme ještě u programu **Cesta** z minulé kapitoly. Program upravíme tak, abychom Petříka v bludišti ovládali klávesnicí, namísto aby hledal cestu sám. Budeme zkoušet více metod ovládání, necháme si je všechny v programu uschované pro možnost porovnání. Uschováme si i metodu, ve které Petřík hledá východ sám.

Přetáhněte do hlavního cyklu programu ze skupinky **řízení programu** prvek **skupina**. Přemístěte do něj všechny příkazy, které jsou nyní v hlavním cyklu: prvek otočení vpravo, cyklus pro nalezení průchodného políčka a příkaz pro provedení kroku. Skupinu sklapněte a připište k ní poznámku **Petřík jde sám**. Tím jsme si metodu automatického vyhledání východu uschovali do jedné skupiny, aby se nám „nepletla dohromady“ s ostatními metodami, které budeme vytvářet. Zkuste program pro kontrolu spustit. Měl by fungovat stejně jako před úpravou.

Novou skupinu vyberte jedním kliknutím myší na její ikonku. Na panelu nástrojů zapněte tlačítko **Vypnout**. Skupina i všechny prvky v ní zešednou. Tato funkce nám umožňuje vypínat části programu, pokud je zrovna v programu nepotřebujeme. Když program spustíte, uvidíte, že Petřík pouze stojí v levém dolním rohu a nic se neděje. Program se chová tak, jakoby tam skupina s příkazy vůbec nebyla.

Naši novou část programu budeme konstruovat uvnitř hlavního cyklu programu, hned pod předcházející skupinou. Využijeme nový konstrukční prvek - **vícestupňové větvení příkazů** (mezi programátory někdy označován jako „hrábě“). Je ve skupince **řízení programu**. Položte ho do hlavní smyčky pod skupinu **Petřík jde sám**.

Vícestupňové větvení příkazů slouží k větvení programu na více částí v závislosti na hodnotě proměnné či výrazu. Obvyklým použitím je větvení podle číselné hodnoty. Později budeme větvení používat i pro jiné typy dat, jako například texty.

Prvek větvení obsahuje další čtyři prvky. Do prvního prvku, **pro hodnotu výrazu**, vkládáme proměnnou či výraz, podle kterého se má větvení provést. My sem umístíme prvek **vstup klávesy (čeká na stisk)**, který bude větvící konstrukci předávat číselný kód stisknuté klávesy.

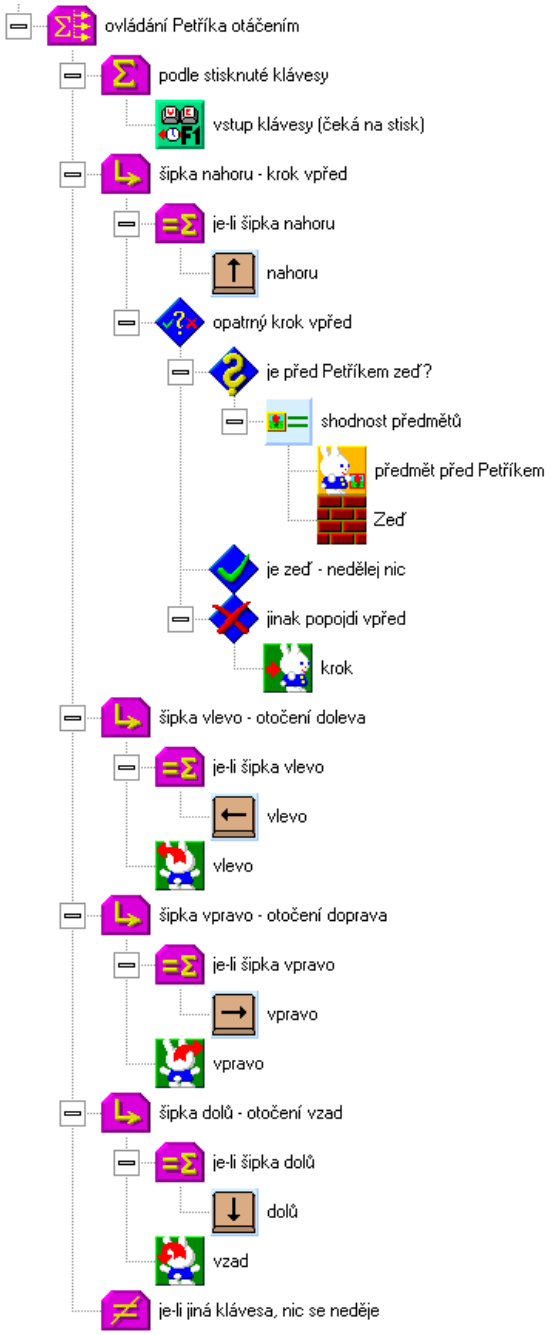
Následují dva spojené prvky - **proveď příkazy v případě, že** a **výraz je roven hodnotě**. Jedná se o jednu větev větvící konstrukce. Do prvku **výraz je roven hodnotě** umisťujeme hodnotu, pro kterou budou příkazy ve větvi provedeny. Větví může být libovolné množství, stejně tak testovaných hodnot ve větvi může být libovolně. Celou větev **proveď příkazy v případě, že** zkopírujte ještě třikrát.

Máme teď čtyři větve větvící konstrukce. Do testované hodnoty v první větvi dáme ze skupinky **ovládání**, **klávesnice**, **klávesy**, **řídicí klávesy** kód klávesy **nahoru**. Tato větev se provede při stisku klávesy nahoru. Do příkazů větve dáme příkaz pro krok. Budeme hlídat, aby Petřík neprocházel přes zeď. Okraj plochy

hlídat nemusíme, Petřík ven z plochy neuteče. Namísto jednoduchého kroku použijeme podmíněný příkaz. Do testu podmínky dáme porovnání předmětů, zda je před Petříkem zeď. Pokud ano, nebude se dít nic. Pokud ne, Petřík udělá krok.

Do dalších tří větví uvedeme postupně jako testované hodnoty klávesy **vlevo**, **vpravo** a **dolů** a jako příkazy prvky **vlevo**, **vpravo** a **vzad**.

Posledním prvkem větvící konstrukce je prvek **jinak proveď příkazy**. Jak napovídá název, příkazy v této části konstrukce se provedou tehdy, když testovaná hodnota nebude nalezena ani v jedné z větví. V našem případě zde nebude nic.



Spusťte program. Petřík stojí v levém dolním rohu. Vyzkoušíte-li kurzorové klávesy (šipky), zjistíte, že se Petřík otáčí vlevo, vpravo, vzad a dělá krok vpřed. Můžete Petříka vodit bludištěm. Až dorazíte k východu, Petřík se otočí na vás a po stisku další klávesy program skončí.

Použitý způsob ovládání je ovládání typické pro auta. Používáme ho v případech, kdy potřebujeme pohybovaným objektem přesně otáčet do požadovaného směru, ale není tak důležitá rychlost pohybu po ploše (manipulovatelnost).

V případech, kdy je potřeba se rychle a snadno přemisťovat, použijeme následující způsob ovládání. Řídíme v něm ne otáčení postavy, ale popocházení ve směru šipky.

Zkopírujte pravým tlačítkem myši předešlou větvící konstrukci ještě jednou níže. Původní konstrukci vypněte tlačítkem **Vypnout**. Novou konstrukci nazvěte **ovládání Petříka směry**. Z větví vyhoďte příkazy pro otáčení vlevo, vpravo a vzad.

První větev větvící konstrukce (pro šipku nahoru) upravíme tak, aby se Petřík otočil nahoru a udělal krok. Krok máme již připraven - kontroluje se v něm, zda před Petříkem není zeď. Směr otočení Petříka nastavíme pomocí prvku **směr**. Najdete ho v rozšířených funkcích Petříka. Prvek vložte před podmínku k provedení kroku Petříka. Jako parametr připojte prvek **nahoru (1/2 pi, tj. 90 stupňů)**, je ve skupince **výpočty**, podskupince **úhel, směr**.

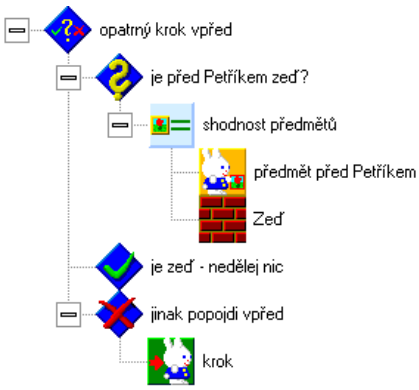
Vytvořte příkazy pro nastavení směru i v ostatních větvích větvící konstrukce, směry změňte odpovídajícím způsobem podle směru šipek. Příkaz pro opatrné provedení kroku bychom mohli zkopírovat z první větve, ale raději použijeme **funkci**.

Co je funkce? Funkce je prvek, který obsahuje část programu. Namísto vícenásobného výskytu shodné části programu použijeme vícekrát funkci, která potřebnou část programu vykoná. Program se zpřehlední a budou se nám snáze provádět změny.

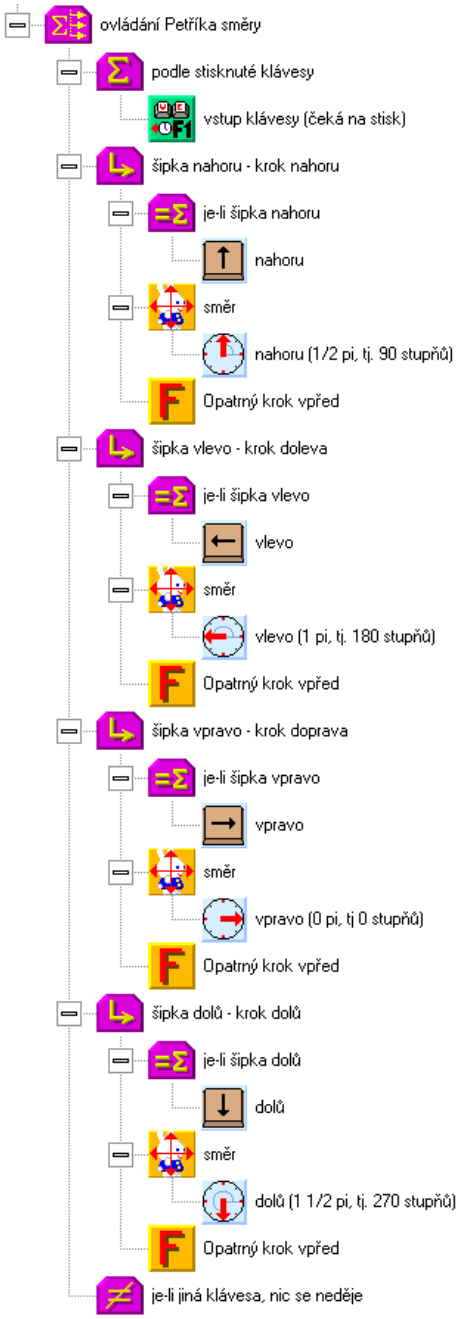
Novou funkci vytvoříme přetažením prvku **funkce** z okna **Knihovna proměnných a funkcí** do okna **Společné proměnné a funkce**. Funkci pojmenujte **Opatrný krok vpřed**. Když dvakrát kliknete na ikonku funkce, objeví se prázdné editační okno, funkce zatím neobsahuje žádné příkazy.

Mohli bychom teď ve funkci vytvořit příkaz pro krok, ale my si to usnadníme, protože máme tento příkaz již v programu vytvořený. Přepněte se zpět do hlavní funkce programu (tlačítkem **Předešlá editace**) a najděte podmíněný příkaz s testem zdi a krokem vpřed (je ve větvi pro šipku nahoru). Kliknutím na ikonku podmíněný příkaz označte. Klikněte na tlačítko **Kopie** (nebo na klávesnici stiskněte **Ctrl+C**). Označená větev programu se uschová do schránky k přenášení dat.

Vraťte se do funkce **Opatrný krok vpřed** (tlačítkem **Následující editace**). Klikněte na tlačítko **Vložit** (nebo stiskněte **Ctrl+V**). V okně se objeví přenášená část programu.



Teď již můžeme z větve pro šipku nahoru podmíněný příkaz vyhodit a nahradit ho funkcí pro opatrný krok. Doplníme ji i do ostatních větví za příkazy nastavení směru.



Program můžete spustit a vyzkoušet. Jak vidíte, Petřík teď popojde o krok ve směru stisknuté klávesy šipky.

Do třetice všeho dobrého si vyzkoušíme ještě jednu metodu ovládání, kterou budeme používat u složitějších a dokonalejších her. Možná jste si všimli, že držíte-li u předešlých dvou metod chvíli klávesu, Petřík se pohybuje i potom, co jste již klávesu dávno pustili. Příčinou je rychlejší generování kódů držených kláves, než je schopen Petřík popocházet, a tak se klávesy uchovávají v zásobníku kláves. Jistým řešením je příkaz **vyprázdnění zásobníku kláves**, který zruší klávesy uchované v zásobníku kláves. Použijeme ho vždy na začátku každé obsluhy platného znaku z klávesnice před provedením kroku Petříka.

Další nectnost, která nám může u předešlých dvou metod ovládání vadit, se projevuje při ovládání rychlých dějů nebo při jemném grafickém ovládání. Zkuste si v nastavení sprajtu Petříka změnit **počet fází na krok** na **1** (po zkoušce vraťte zpět na **8**). Petřík bude rychle přeskakovat po políčkách. Když na delším volném úseku klávesu chvíli podržíte, Petřík poskočí nejdříve o políčko, chvíli čeká a až pak začne rychle přeskakovat na další políčka. Příčinou tohoto jevu je způsob generování znaků klávesnicí - prodleva po první klávese a potom opakování. Při psaní textů je tato vlastnost užitečná, ale při hraní her nám může trochu vadit.

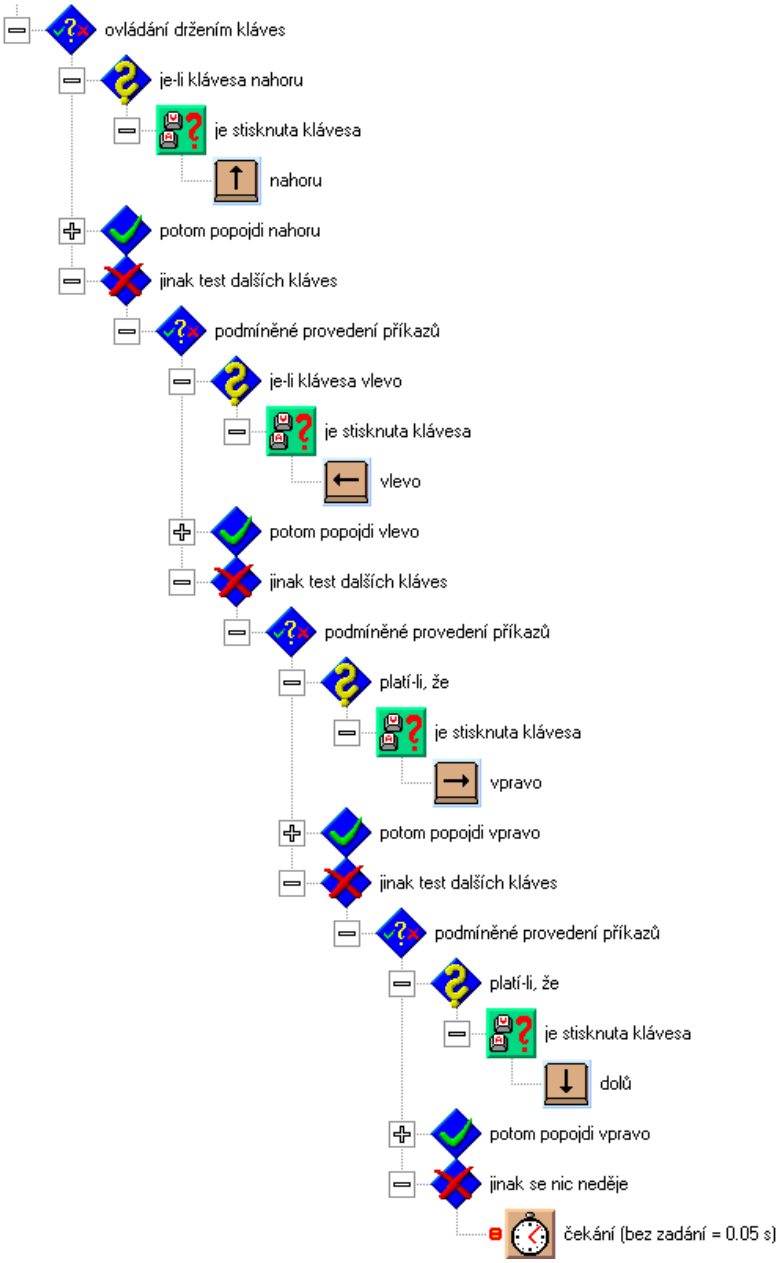
Náš požadavek na ovládání klávesnicí je, aby se Petřík pohyboval vždy, když držíme klávesu. Bez úvodních prodlev a bez setrvačnosti po uvolnění klávesy. K tomuto účelu slouží funkce **je stisknuta klávesa** ze skupinky **klávesnice**. Funkce otestuje požadovanou klávesu, zda ji uživatel drží, a navrátí příznak **ano/ne**.

Nejdříve opět vypněte předešlou metodu tlačítkem **Vypnout**. Pod ni připravte prvek **podmíněné provedení příkazů**, můžete ho nazvat třeba **ovládání Petříka držením kláves**. Do testu podmínky **platí-li, že** vložte prvek **je stisknuta klávesa**, jako parametr k němu připojte klávesu **nahoru**. Do větve platnosti podmínky **potom proveď** vložte příkazy pro popojítí nahoru, tak jako u předešlého typu ovládání. Konstrukci teď čteme: „*Je-li držena klávesa nahoru, potom se otoč nahoru a opatrně popojdi o krok, jinak pokračuj testem dalších kláves.*“

Ve větvi neplatnosti podmínky **jinak proveď** vytvořte stejnou konstrukci znovu, tentokrát ale pro směr vlevo. V její větvi neplatnosti podmínky vytvořte obsluhu směru vpravo a nakonec vytvořte obsluhu směru dolů. Do větve neplatnosti podmínky v obsluze směru dolů vložte příkaz **čekání (bez zadání = 0.05 s)**, najdete ho ve skupince **řízení programu**. Značka malého rovnítka před ikonkou čekání indikuje nepovinnost parametru příkazu, žádný parametr nemusíme doplňovat.

Proč příkaz čekání? Operační systém Windows je víceúlohový. To znamená, že současně může být v chodu více programů. Každý program by měl zajistit, aby zbytečně nespotřebovával výkon počítače, když to nutně nepotřebuje. Známkou dobrého programátora je slušné chování jeho programů v prostředí, ve kterém poběží. Zkušení uživatelé si mohou spustit systémový program **Sledování systému** a sledovat zatížení procesoru. Program neberoucí ohled na své okolí zabírá téměř celý výkon počítače a počítač se tak stává těžce ovladatelný.

Na obrázku vidíte hotovou obsluhu ovládání Petříka. Části pro obsluhu kroku v daném směru jsou sklapnuty, známe je již z předešlého ovládání.



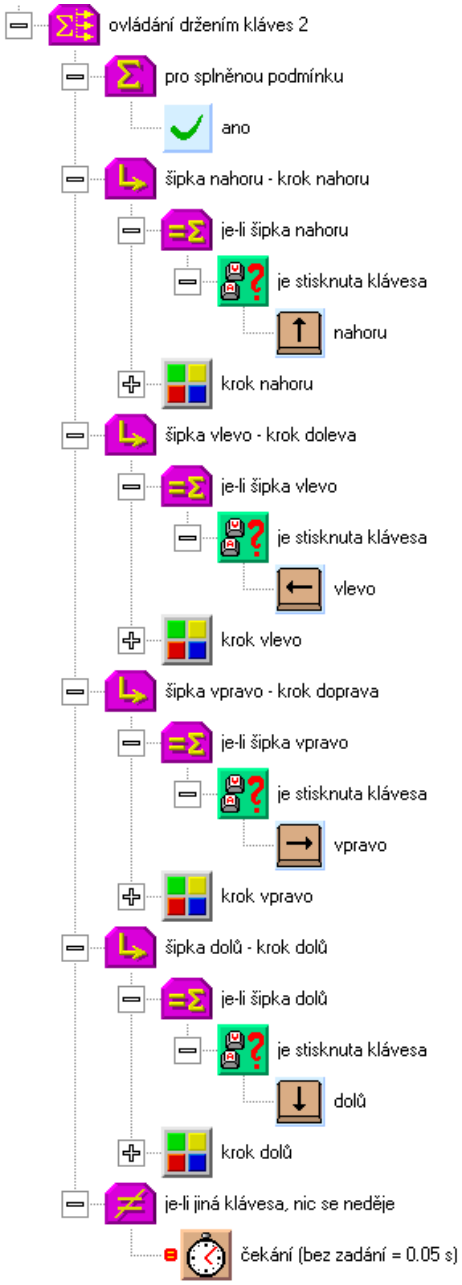
Nevýhodou takového dlouhého řetězce podmínek je, že může být na první pohled nepřehledný. Ukážeme si ještě jinou možnost stavby podobné konstrukce.

Vypněte naši poslední variantu tlačítkem **Vypnout**. Zkopírujte si znovu variantu **ovládání Petříka směry** a přejmenujte ji na **ovládání držením kláves 2**. Podmínková konstrukce pracovala tak, že se provedla právě jedna z větví v závislosti na stisknuté klávese. Z prvku **pro hodnotu výrazu** zrušte funkci vstupu z klávesnice a nahraďte ji prvkem **ano** (ze skupinky **výpočty**, **logické operace**).

V testovaných hodnotách jednotlivých větví postupně nahraďte kódy kláves kombinacemi prvku **je stisknuta klávesa** a kódu klávesy. Postupujte následovně: Kliknutím myší na ikonu kódu klávesy prvek kódu klávesy vyberte. Stiskem tlačítka **Vyjmout** (nebo z klávesnice **Ctrl+X**) přesuňte prvek kódu klávesy do schránky.

Na jeho původní místo vložte prvek **je stisknuta klávesa** . Vyberte ho myší a stiskněte tlačítko **Vložit** (nebo z klávesnice **Ctrl+V**). Kód klávesy se připojí k testu stisku klávesy. Závěrem doplňte do prvku **jinak proveď příkazy** prvek **čekání (bez zadání = 0.05 s)** , jinak by program ve volných chvílích přetěžoval chod počítače.

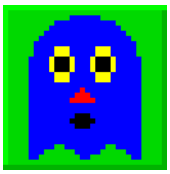
Hotovou konstrukci vidíte na dalším obrázku. Jak funguje? Program při svém chodu na začátku zjistí, že ve větvích bude testována logická hodnota platnosti. Prochází jednotlivé větve a hledá stejnou hodnotu, tedy platnou podmínku. Příkazy v nalezené větvi provede a pokračuje až za celou konstrukcí vícestupňového větvení příkazů, dalších případných splněných podmínek a větví si již nevšímá. Jinak řečeno, vykoná se první (a jediná) větev se splněnou podmínkou (stisknutou klávesou), jinak se vykoná až poslední z příkazů - čekání.



11 Samá příšera

Zkusíme si udělat malou hru. Ani byste nevěřili, co se stalo. Petřík v bludišti narazil na příšery. Ale nebojte se, až ho ozbrojíme, hravě je zvládne. Budeme pokračovat v našem programu **Cesta** z předešlé kapitoly, ve kterém jsme zanechali zapnutý poslední způsob ovládání - **ovládání držením kláves 2**. Nebo si můžete otevřít program **Příšery**, který je již připraven v ukázkových programech Petra.

Nejdříve zajistíme, aby se v bludišti objevovaly příšery. Zkopírujte pravým tlačítkem myši v okně **Společné proměnné a funkce** prvek **prázdné políčko** . Přejmenujte ho na **Příšera** a přikreslete do něj nějakou příšerku. Může vypadat třeba takto:



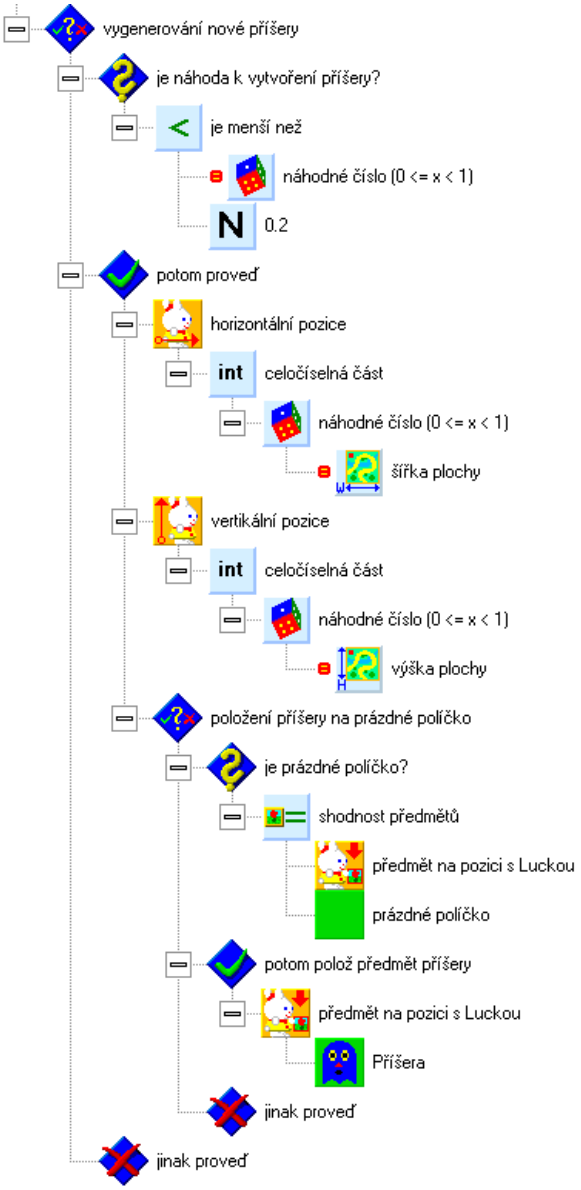
Příšerky se nám budou generovat náhodně. Do hlavní smyčky programu přidejte prvek **podmíněné provedení příkazů** . Do testu podmínky **platí-li, že** doplňte porovnávací funkci **je menší než** (ze skupiny **výpočty**, podskupiny **porovnání**). Jako první parametr pro porovnání použijeme funkci **náhodné číslo (0 <= x < 1)** (ze skupiny **výpočty**, podskupiny **funkce**). Druhým parametrem bude **číselná konstanta** s nastavenou hodnotou **0.2** (buď jako text prvku nebo pomocí prvků číslic a desetinné tečky). Za chvíli doplníme obsluhu vytvoření příšery.

Co taková funkce s náhodou bude dělat? Z textů k prvkům čteme: „*Platí-li, že náhodné číslo je menší než 0.2, vytvoř příšeru.*“ Náhodné číslo je desetinné číslo s náhodnou hodnotou od nuly do jedničky. Číslo **0.2** je jedna pětina z jedničky. Náhodné číslo má hodnotu menší než **0.2** v pětině případů. To znamená, že při opakovaném průchodu se příšera vytvoří v každém pátém případě.

K vytvoření příšerky použijeme Petříkovu kamarádku - Lucku. Nejdříve určíme náhodné místo, kde bude příšerka vytvořena. Do větve pro splněnou podmínku přetáhněte prvky **horizontální pozice** a **vertikální pozice** (ze skupiny **Lucka**, podskupiny **Lucka - rozšíření**). K oběma prvkům připojte funkci **celočíselná část** (ze skupiny **výpočty**, podskupiny **funkce**), protože souřadnice políček udáváme v celých číslech. K funkcím celočíselné části připojte funkci **náhodné číslo (0 <= x < 1)** . K náhodnému číslu lze připojit jako parametr číslo, určující rozsah náhodného čísla. Například přidáním čísla 10 by náhodné číslo generovalo čísla v rozsahu 0 až 10. My doplníme k náhodnému číslu pro horizontální pozici prvek **šířka plochy** a pro vertikální pozici prvek **výška plochy** (oba ve skupince **plocha**).

Rozeberme si podrobněji způsob výpočtu **horizontální** náhodné pozice. Výchozím prvkem v horizontálním směru je **šířka plochy**. Tento prvek nám navrací číslo udávající šířku plochy v krocích (v políčkách). V našem programu to bude číslo 12, což je šířka plochy, jak jsme si ji kdysi nastavili. Údaj o šířce plochy prvek předá funkci **náhodné číslo (0 <= x < 1)**, která vygeneruje náhodné číslo v rozsahu 0 až šířka plochy (bez koncové hodnoty), nebo-li číslo v rozsahu 0 až 11.99999999. Z funkce náhodného čísla je číslo předáno funkci **celočíselná část**. Ta ořízne část čísla za desetinnou tečkou a navrátí pouze celou část čísla. Vznikne číslo 0 až 11, což jsou souřadnice prvního až posledního políčka ve vodorovném směru. Možná jste si již všimli, že se políčka počítají z levého dolního rohu, počínaje nulou.

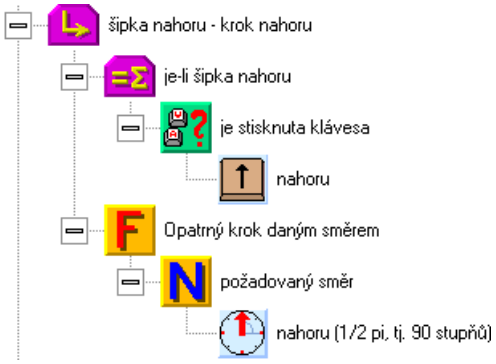
Máme tedy Lucku na náhodné pozici v ploše. Teď bychom mohli položit do plochy předmět příšerky, ale ještě musíme otestovat, zda je na tomto místě volné políčko, zda tam není třeba zeď. Pokud políčko nebude volné, nic se neprovede a příšerka se nevytvoří. Výsledek je zde, zkuste program spustit:



Nyní trochu vylepšíme obsluhu provádění kroků Petříka. Zkopírujte pravým tlačítkem myši v okně **Společné proměnné a funkce** funkci **Opatrný krok vpřed**. Vytvoří se nová funkce s názvem **Opatrný krok vpřed 2**. Přejmenujte ji na **Opatrný krok daným směrem**.

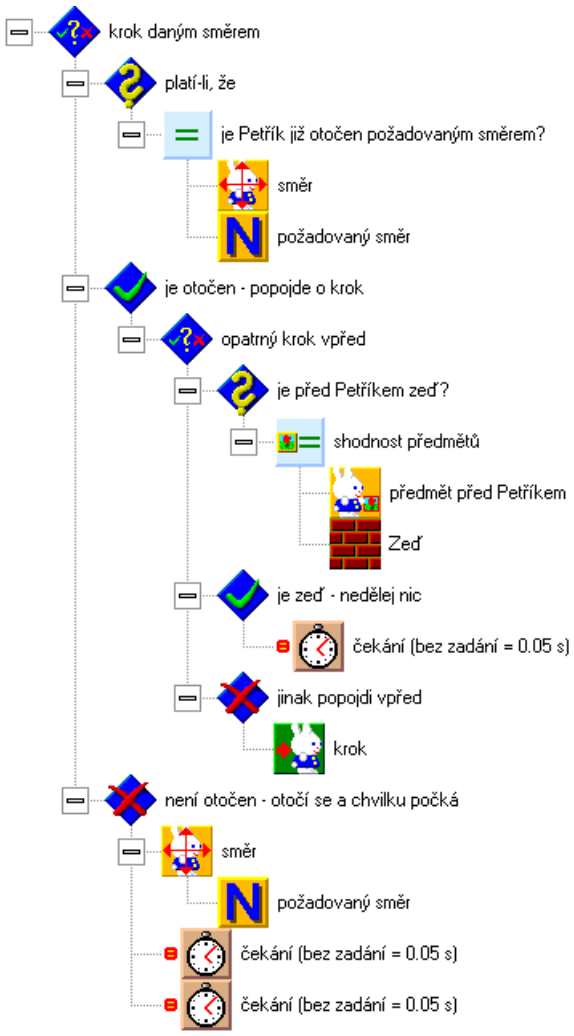
Přepněte se dvojklikem do nově vytvořené funkce a podívejte se na okno vlevo dole. Má titulek **Místní proměnné a funkce**. Význam okna nás zatím ještě nebude zajímat, pouze si všimneme prvku **vstupní proměnné**. Z okna **Knihovna proměnných a funkcí** přetáhněte do prvku novou číselnou proměnnou **číslo** a nazvěte ji **požadovaný směr**.

Přepněte se zpět do hlavní funkce programu. V konstrukci **ovládání držením kláves 2** najděte první větev ovládání Petříka, obsluhu pro šipku nahoru. Položte sem nově vytvořenou funkci **Opatrný krok daným směrem**. Po položení funkce můžete vidět, že k funkci je připojena číselná proměnná s názvem **požadovaný směr**. Je to samozřejmě ta, kterou jsme si právě sami vytvořili. S její pomocí budeme funkci předávat parametr, kterým bude požadovaný směr kroku. Přetáhněte z příkazu pro nastavení směru do parametru funkce prvek **nahoru**. Zbývající příkazy - nastavení směru a funkci **Opatrný krok vpřed** - můžete z větve vyhodit. Stejně upravte i větve pro ostatní směry. Takto bude vypadat obsluha kroku nahoru:



Teď připravíme obsah funkce **Opatrný krok daným směrem**. Přepněte se do ní. Nejdříve zkušebně doplňte na začátek funkce příkaz pro nastavení směru Petříka a jako parametr doplňte vstupní proměnnou **požadovaný směr**. Tím jsme vlastně navrátili původní funkčnost programu, abychom ověřili, že jsme při úpravách neudělali chybu. Program spusťte a vyzkoušejte ho, měl by fungovat stejně jako dřív. Je-li vše v pořádku, můžete příkaz pro nastavení směru opět zrušit.

Pokud bychom původní ovládání použili pro střílení, tak by nám možná mohlo vadit, že se nemůžeme otočit na cíl, aniž bychom přitom nepopošli o krok k cíli. Uděláme si proto malé vylepšení ovládání. Jestliže nebude Petřík otočen v požadovaném směru, nejdříve se tam otočí a až potom se rozejde. Bude tak možné Petříka krátkým ťuknutím na klávesu otočit a až držením klávesy ho rozejít vpřed. Upravte obsah funkce podle následujícího obrázku.



Na začátku funkce se nejdříve provede porovnání současného směru otočení Petříka s požadovaným směrem, předaným jako parametr funkce. Zjistíme tak, zda je Petřík v požadovaném směru již otočen.

Je-li Petřík otočen správně, může popojít o krok vpřed. Udělá to ovšem opatrným způsobem. Nejdříve otestuje, zda před ním není zeď. Není-li zeď, může popojít. Je-li zeď, zůstane stát a proběhne obsluha čekání. Hlavní smyčka programu totiž potřebuje, aby každý její průchod trval alespoň jedno čekání kvůli rovnoměrnému vytváření nových příšerek. Jedno čekání je v hlavní smyčce dosazeno v případě, že není stisknuta žádná klávesa pro pohyb. Čekání při stisku klávesy pro pohyb je obslouženo v této funkci. Nečeká se po příkazu kroku, příkaz kroku již čekání zajišťuje vnitřně.

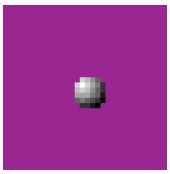
Není-li Petřík správně otočen, otočí se do požadovaného směru a program chvíli čeká. Čekání zajišťuje, aby se Petřík při krátkém ťuknutí do klávesnice nerozešel.

Spusťte program a vyzkoušejte ovládání. Především zkuste otáčení Petříka na místě a zatáčení během chůze.

Pustíme se do obsluhy střelby po příšerkách. Trochu si přitom „vystřelíme z Lucky“. Použijeme totiž ke střelbě opět Petříkovu kamarádku - Lucku. Lucka bude střelou, kterou bude Petřík střílet. Ale to je v pořádku, udělá to pro něj ráda.

Dvojklikem na sprajt **Lucka** v okně **Společné proměnné a funkce** vyvolejte editaci sprajtu Lucky (poznámka - sprajtem označujeme pohyblivý animovaný objekt). Uvidíte plochu s 4 x 5 obrázky Lucky. Stiskněte tlačítko **Vlastnosti**, objeví se okno pro nastavení vlastností sprajtu. Změňte řádek **Fází na krok (0 = ihned)** z **8** na **2** (střela poletí rychle) a řádek **Fází pro pohyb** změňte ze **4** na **0** (střela při letu nepotřebuje měnit vzhled). Stiskněte klávesu **Enter** (nebo klikněte na tlačítko **OK**). Obrázky sprajtu se změnily tak, že Lucka je teď pouze v jednom sloupci.

Chytněte levým tlačítkem myši první z obrázků Lucky a odsuňte ho mimo plochu obrázků. Po uvolnění tlačítka myši obrázek zmizí, vyhodili jsme ho ze sprajtu pryč. Dvakrát klikněte levým tlačítkem myši na uvolněné prázdné okénko. Vyvolá se editor obrázku sprajtu. Namalujte obrázek střely - například malou šedou kuličku (nástrojem **koule**, použijte bílou barvu):



Jako podklad obrázku ponechejte původní našedle fialovou barvu. Je to „průhledná“ barva, zajišťující viditelnost původního obsahu okna v okolí střely. Ve výběru barev grafického editoru je to barva vlevo nahoře.

Tlačítkem **Předešlá editace** se vraťte zpět do editoru sprajtu Lucky. Uchopte pravým tlačítkem myši upravený první obrázek a přetahováním jej rozkopírujte do všech ostatních políček sprajtu. Později můžete obrázky ještě posunout tak, aby střely vycházely přesně z místa hlavně pistole.

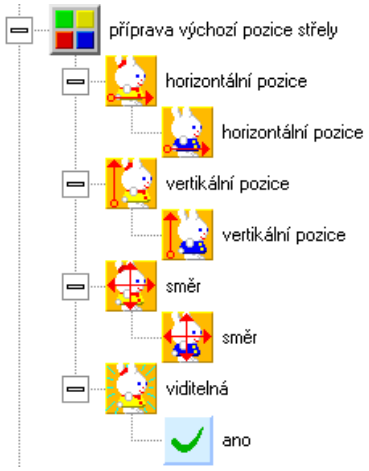
Teď upravený sprajt vyzkoušejte. Klikněte na tlačítko **Test**. Objeví se okno se zelenou plochou, uprostřed je sprajt střely. Klikněte někam do plochy okna, střela se na určené místo přesune. Test sprajtu ukončíte tlačítkem **Storno**.

Střelu máme připravenou, nyní musíme vytvořit ovládání střely. Vraťte se do hlavní funkce programu. Do hlavní smyčky, ihned za obsluhu vygenerování nové příšery, doplňte nový prvek **podmíněné provedení příkazů** a nazvěte ho **výstřel ze zbraně**. Střelbu ze zbraně budeme aktivovat klávesou mezerníku, proto do testu podmínky **platí-li, že** dejte funkci **je stisknuta klávesa** s prvkem klávesy **mezerník** (ve skupině **klávesy**, podskupina **znakové klávesy**).

Celkovou sestavu obsluhy střelby vidíte na následujícím obrázku. V obsluze nastavíme střelu (Lucku) na výchozí pozici, potom střela letí, po doletu střelu vypneme.

Při zahájení letu střely nastavíme pozici a směr střely podle pozice a směru Petříka, potom zapneme viditelnost střely. Prvky pozice a směru již známe, co se ale skrývá za prvkem viditelnosti? Všechny sprajty (tj. pohyblivé objekty, kterými jsou i Petřík a Lucka) mají dva základní stavy - viditelný a neviditelný. Ve viditelném stavu (kromě toho, že je sprajt vidět) se provádí animace sprajtu a sprajt se pohybuje pomalu. Je-li sprajt neviditelný, přemisťuje se na novou pozici okamžitě. Z Petříka se stává „Super Petřík“, který se pohybuje jako blesk.

Viditelnost Lucky nastavíme příkazem **viditelná**. Jako parametr viditelnosti sprajtů obvykle uvádíme logickou konstantu **ano** nebo **ne**, tak přepínáme sprajt mezi režimy „Louda“ a „Super“. Celou obsluhu aktivace střely vidíte na následujícím obrázku. Pro zpřehlednění je obsluha v samostatné skupině.

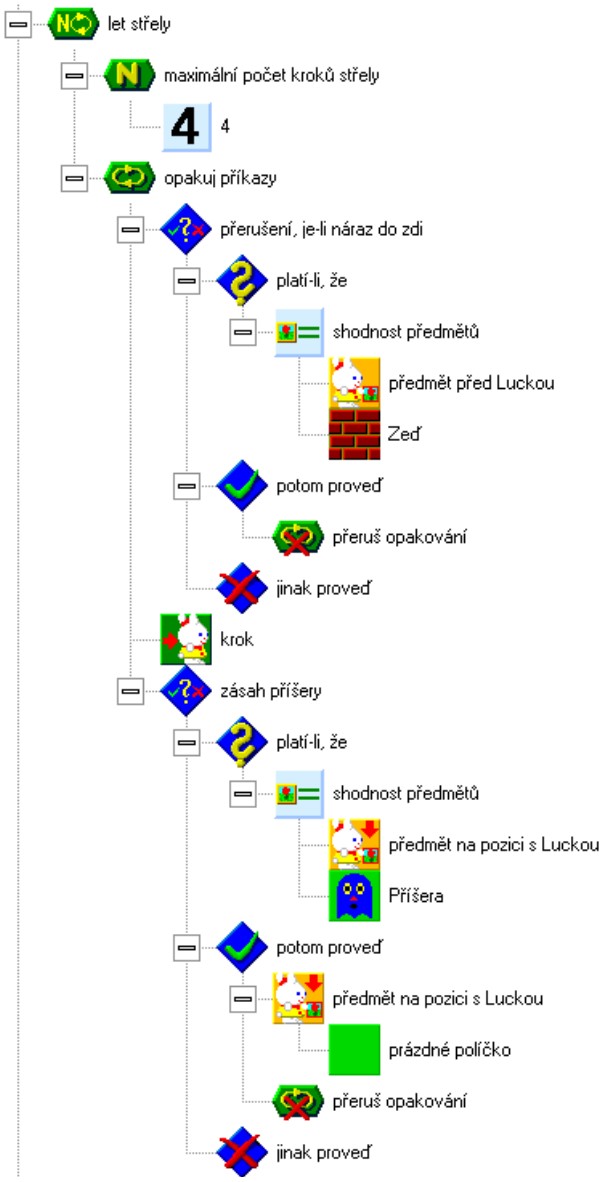


Vytvoříme obsluhu pohybu střely. Za skupinu přípravy výchozí pozice střely přidejte prvek cyklu **opakování příkazů s počtem**, pojmenujte ho **let střely**. Do počtu opakování cyklu dejte číslo **4**. To je maximální vzdálenost, kam střela poletí.

Let střely přerušíme, narazí-li střela do stěny. Proto dejte na začátek těla cyklu podmínku s testem, zda je před Luckou předmět **Zeď**. Pokud ano, přeruší se provádění cyklu příkazem **přeruš opakování** (je ve skupince **řízení programu**). Po kontrole nárazu do zdi již můžeme dát příkaz pro pohyb střely **krok**.

Za posunem střely o krok otestujeme, zda střela zasáhla cíl. Použijeme podmíněný příkaz s testem, zda předmět na pozici s Luckou je předmět **Příšera**. Pokud ano, vymažeme příšerku položením předmětu **prázdné políčko** a přerušíme cyklus letu střely. Někdy v budoucnu sem můžete doplnit další obsluhy zásahu, jako třeba čítač zásahů nebo zvuk zaúpění příšery.

Celou obsluhu letu střely vidíte na dalším obrázku. Za obsluhou letu střely uveďte ještě vypnutí viditelnosti Lucky a program můžete vyzkoušet.



Chcete-li dále hru vylepšovat, můžete přikreslit Petříkovi zbraň. Nebo můžete na začátek podmíněného příkazu pro obsluhu střelby doplnit zvuk výstřelu. Použijte prvek **přehrátí zvuku**  . Do okna **Společných proměnných a funkcí** přetáhněte z banky zvuků třeba zvuk **[vzory]\Zbraně\Puška a pistole\Pistole** a ten použijte ve funkci přehrátí zvuku. Podobně můžete do obsluhy zásahu příšery doplnit zaúpění příšery - například zvuk **[vzory]\Lidské\Výkřiky\Au 2**.

12 Jak nakrmit hada

Naším dalším programem bude hra **Had**. S podobnou hrou jste se zřejmě již setkali. Po ploše se pohybuje had, požírá jídlo a s každým polknutím se prodlouží o článek. My si hru ještě trochu vylepšíme. Had bude požírat různé typy jídla a články hada budou také různé podle jídla, které had spolkne.

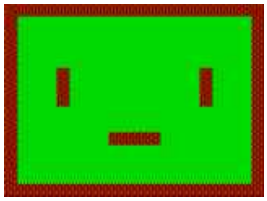
Založte nový program s názvem **Had** (popř. **Had 2**, pokud program tohoto jména již existuje). Otevře se program zející prázdnotou a bez jediného příkazu, ale to nám nevadí, protože za chvíli okna programu zaplníme hezkou novou hrou. Jestli máte zájem, můžete otevřít program **Had** již vytvořený v ukázkových programech Petra.

Připravíme hrací plochu programu a to tak, že ji po obvodě obestavíme zdí. Z knihovny předmětů přetáhněte do okna **Společné proměnné a funkce** předmět **Zeď**  (z podskupiny **[vzory]\Domek**). Dvojklikem na prvek **plocha**  vyvolejte editor hlavní plochy programu. Položte předmět **Zeď** do levého horního rohu plochy.

Podíváme se na novou vlastnost plochy. Každé políčko na ploše obsahuje kromě obrázku předmětu ještě pět logických přepínačů a tři číselné hodnoty. Logický přepínač umožňuje uchovat stav **ano**/**ne** a tento stav později testovat. V našem programu použijeme první přepínač k signalizaci, že na políčku je postavena zeď. Nebudeme tak muset vyhodnocovat předmět na políčku, jenom se vždy zeptáme přepínače: „*Je na políčku zeď?*“ Můžeme dokonce používat více variant zdí.

V panelu nástrojů editoru plochy je úplně vpravo rozbalovací seznam typu editace. Rozbalte jej a zvolte funkci **Přepínač 1**  . Zobrazená plocha se změní, v levém horním rohu každého políčka se objeví tmavě červený čtvereček  . Čtvereček je indikátor stavu přepínače 1 v políčku. Nyní je vypnutý (stav „**ne**“). Zapněte v panelu nástrojů přepínací tlačítko **Editace**  (kliknutím na tlačítko). Tím aktivujete režim úpravy obsahu políček. Klikněte na políčko v levém horním rohu s předmětem **Zeď**. Čtvereček se stane světle červený se značkou zaškrtnutí  . Přepínač 1 na políčku je nyní zapnutý (stav „**ano**“). Vypněte editaci políček novým kliknutím na tlačítko **Editace**  .

Teď bychom rádi položili zeď kolem celé plochy. Máme k dispozici trochu snadnější postup, než neustálé kopírování předmětu pravým tlačítkem myši. V panelu nástrojů zapněte přepínací tlačítko **Výplň**  . Stiskněte levé tlačítko myši na předmětu zdi v levém horním rohu plochy a táhněte myší do pravého horního rohu, tam tlačítko myši pusťte. Plocha, kterou jste takto označili, se vyplní výchozím políčkem. Podobně vyplňte i ostatní strany plochy. Nakonec vypněte tlačítko **Výplň**  a ještě můžete pravým tlačítkem myši doplnit zeď někam do plochy, například podle vzoru:



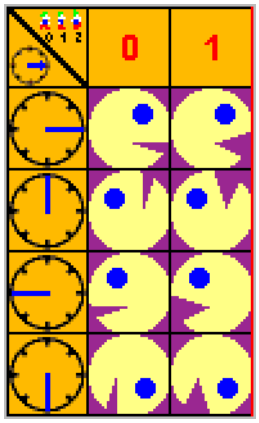
Připravíme jídlo, které bude had požírat. Vytvořte pět nových předmětů a nakreslete do nich obrázky hadího jídla. Například:

Vytvoříme články hada. Můžete zkopírovat předměty jídla a články dokreslit:

Podobně vytvoříme tři předměty s obrázky jedu a jeden článek hada s obrázkem lebky:

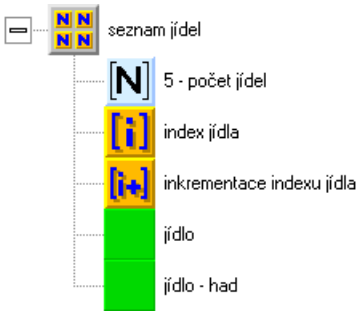
Posledním grafickým úkonem bude vytvoření hlavy hada. Jako hlavu hada použijeme Petříka. Vyvolejte editaci sprajtu Petříka dvojklikem myší na prvek **Petřík** v okně **Společných proměnných a funkcí**. Otevřete tlačítkem **Vlastnosti** okno pro nastavení vlastností sprajtu. Změňte řádek **Prodleva mezi fázemi** z **55** na **165** (doba mezi jednotlivými fázemi animace), řádek **Fází na krok (0 = ihned)** z **8** na **0** (posouvá se ihned), řádek **Fází pro klid** změňte z **1** na **2** a řádek **Fází pro pohyb** ze **4** na **0**. Uzavřete okno stiskem klávesy **Enter**. Sprajt Petříka má teď čtyři směry a dvě fáze pro klid.

Vyhoďte první obrázek vlevo nahoře s Petříkem. Dvojklikem do obrázku vyvolejte editaci obrázku a nakreslete hlavu hada dívající se směrem doprava. Vraťte se tlačítkem **Předešlá editace** do editoru sprajtu. Pravým tlačítkem myši zkopírujte první obrázek i do druhého obrázku na stejném řádku. V editoru upravte hlavu tak, aby had otvíral pusu. Zkopírujte oba obrázky i do ostatních řádků pro ostatní směry. Vyvoláním editací obrázků otočte hlavy podle směrů na řádcích sprajtu, použijte k tomu funkci **Úpravy**. Nakonec můžete sprajt vyzkoušet tlačítkem **Test**.



Předmětů s obrázky jídla máme v našem programu již větší množství a možná bychom je rádi v budoucnu ještě rozšířili. S našimi současnými znalostmi bychom museli každý typ předmětu obsluhovat zvlášť, a to by již bylo trochu těžkopádné. Proto se naučíme používat nový prvek - **seznam**.

Nejdříve si seznam připravíme, přestože zatím nevíme, k čemu vlastně je. Ze skupiny **řízení programu** přetáhněte do okna **Společné proměnné a funkce** prvek **seznam**. Pojmenujte ho **seznam jídel**. Po jeho rozevření uvidíte tři prvky. První prvek, **počet**, přejmenujte na **5 - počet jídel**. Druhý prvek, **index**, přejmenujte na **index jídla**. Třetí prvek, **inkrementace indexu**, přejmenujte na **inkrementace indexu jídla**. Do seznamu doplňte ještě dva nové předměty, pojmenujte je **jídlo** a **jídlo - had**.

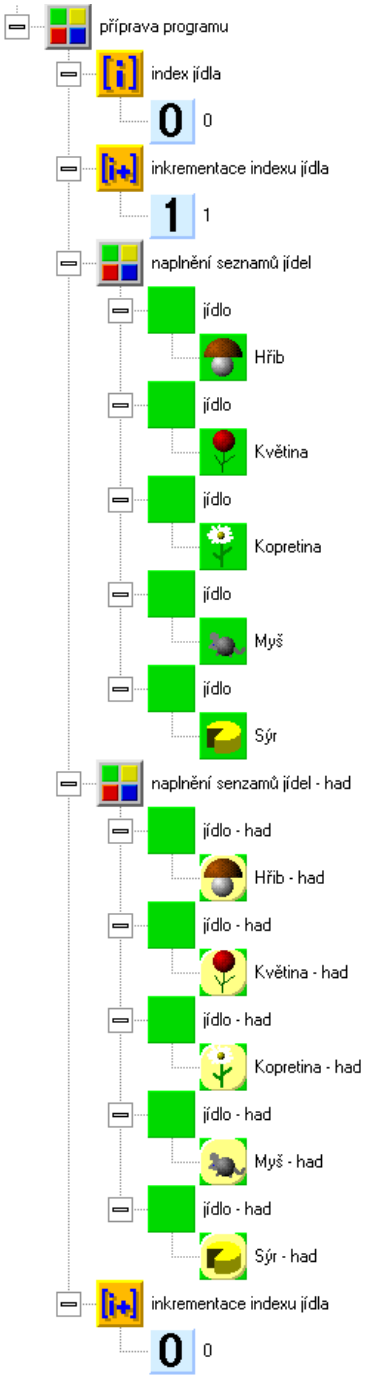


Nyní si vysvětlíme, co to seznam je. Seznam je něco jako kniha. Všechny stránky knihy vypadají podobně, například na každé stránce bude jeden obrázek a jeden text, ale přitom je v každém obrázku či textu něco jiného. V našem seznamu uchováme předměty jídla a článků hada, abychom si usnadnili jejich použití. Na každé stránce seznamu bude jiný obrázek jídla a jiný obrázek článku hada. První položka v seznamu udává počet stránek seznamu, my máme v seznamu 5 typů jídel. Druhá položka je číslo nastavené stránky seznamu - ukazatel seznamu (čísluje se od stránky 0). Třetí položka určuje, o kolik stránek se ukazatel automaticky posune při každém přístupu k datům v seznamu (*poznámka:* slovo inkrementace znamená zvýšení).

Seznam jídel musíme při startu programu naplnit, to znamená určit, jaký předmět na které stránce seznamu bude. Příkazy pro naplnění seznamu vidíte na obrázku na následující stránce. Umístěte je na začátek hlavní funkce programu.

Před zahájením plnění seznamu určíme počáteční stránku, od které budeme seznam plnit. Nastavíme prvek **index jídla** na hodnotu **0**. Použijeme automatické zvyšování čísla stránky, proto nastavíme prvek **inkrementace indexu jídla** na hodnotu **1**.

Nejdříve naplníme předmět **jídlo** na všech stránkách seznamu. Postupně nastavujeme prvek **jídlo** na jednotlivé předměty jídla. Předměty se ukládají do seznamu a po každém nastavení předmětu se stránka automaticky zvýší o jedničku. Po nastavení posledního předmětu se ukazatel stránek automaticky převine na začátek. Začneme plnit předměty **jídlo - had**, opět se stránky samy zvyšují nahoru. Na závěr nastavíme automatické zvyšování stránek na hodnotu **0**, protože v programu již automatické zvyšování nebudeme používat.



Bude-li během hry hráč neúspěšný, to znamená narazí-li had na překážku nebo se nají jedu, začne hra od začátku. K tomu budeme potřebovat znovu vykreslit plochu programu tak, jako při startu, tj. bez jedů a jídel. Mohli bychom plochu vyčistit programově, ale je jeden snadnější způsob. Hlavní plochu si můžeme uschovat do proměnné typu plocha a při nové hře ji pomocí této proměnné jednoduše obnovit.

Vytvořte novou proměnnou plochy (kopií prvku **plocha** z okna **Knihovny proměnných a funkcí** do okna **Společných proměnných a funkcí**) a pojmenujte ji **uschovaná hrací plocha**. Na začátek programu dejte příkaz pro uschování hlavní plochy:

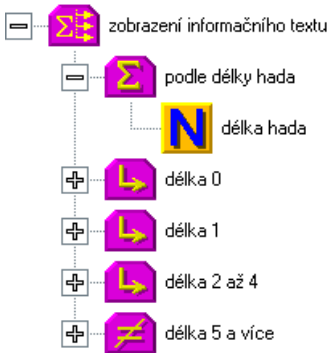
Při hře budeme hráči zobrazovat jeho nejvyšší dosažené skóre. Vytvořte číselnou proměnnou s názvem **dosažené maximum**, její hodnotu nastavte na nulu.

Vytvořte novou funkci s názvem **nová hra**. Funkci necháme vykonat při startu programu, a proto ji doplňte na začátek programu za skupinu pro naplnění seznamu jídel. Její obsah budeme sestavovat podle obrázku uvedeného níže.

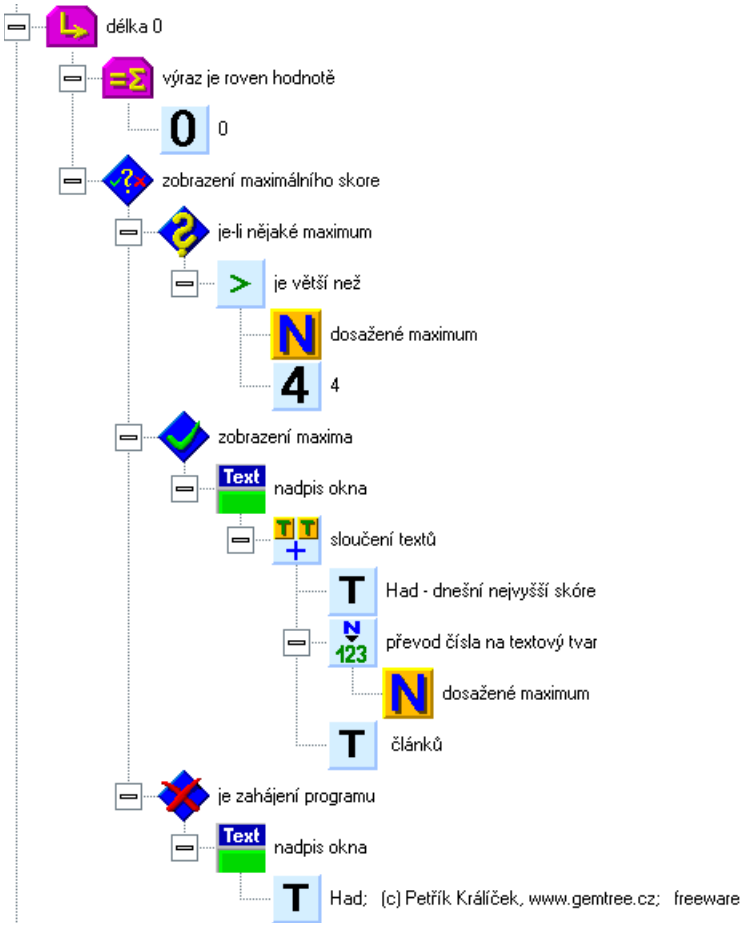
Vytvořte logickou proměnnou **příznak pohybu hada**, číselnou proměnnou **délka hada**, číselnou proměnnou **čítač pro další krok** a funkci s názvem **zobrazení počtu článků hada**. Logickou proměnnou **příznak pohybu hada** použijeme k indikaci zahájení pohybu hada (po startu hry bude had stát na místě a čekat na stisk první klávesy). Hodnotu proměnné nastavte ve funkci **nová hra** na vypnuto. Proměnná **délka hada** udává počet článků hada a během hry se zvyšuje. Při zahájení nové hry ji nastavte na **0**. **Čítač pro další krok** čítá časové impulsy **0.1** sekundy, než se provede posun hada o políčko (snižuje se od hodnoty **5** do **0**). Nastavte čítač při zahájení hry na hodnotu **1**, aby se první krok provedl ihned po stisku první klávesy.

Dalším příkazem při zahájení nové hry je návrat uschované hrací plochy. Použijte stejný příkaz jako při úschově plochy, ale s obráceným pořadím prvků. Následujícími příkazy se provede nastavení výchozí pozice hada. Vypněte viditelnost Petříka, nastavte jeho souřadnice na **X=9** a **Y=8**, otočte ho vpravo a opět viditelnost zapněte. Na závěr funkce **nová hra** uveďte funkci **zobrazení počtu článků hada**.

Teď se zaměříme na funkci zobrazení počtu článků hada. Při mnoha hrách je třeba uživateli sdělovat různé informace, jako je např. dosažené skóre. Nechceme-li se zobrazováním informací příliš zabývat, je dobrým a rychlým řešením vypisovat takové informace v nadpisu okna. My budeme sdělovat hráči dosaženou délku hada a případně maximální dosažené skóre. Při zobrazení počtu článků se budeme snažit alespoň částečně vyjadřovat lidsky, proto rozlišíme stavy „1 článek“, „2…3…4 články“ a „5…6… článků“. Většina hráčů nám snad odpustí češtinářský prohřešek, když namísto „23 články“ vypíšeme „23 článků“. Zde je obsah funkce:



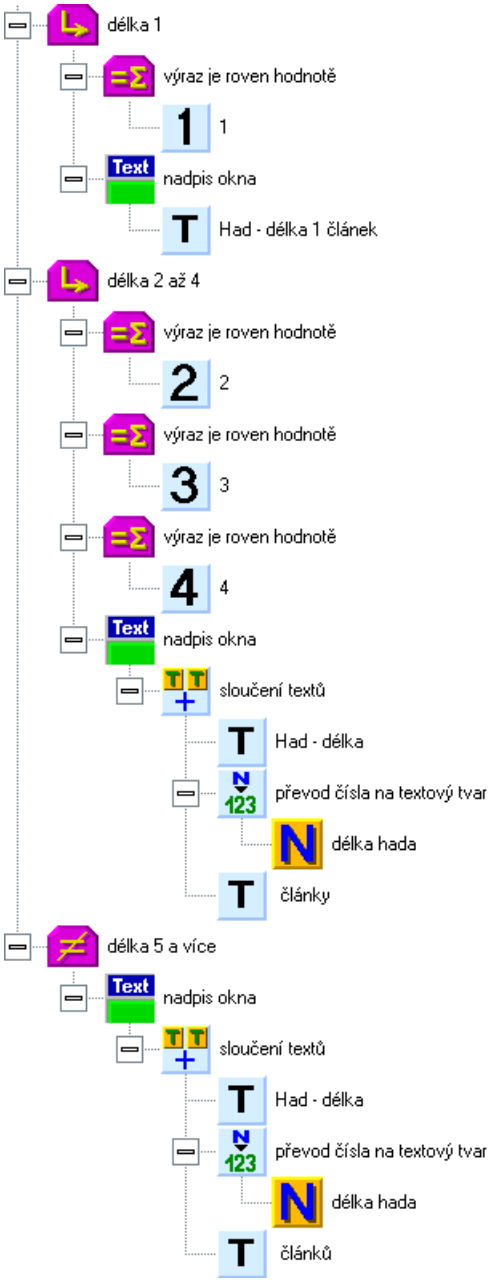
Délka hada **0** se vyskytne vždy při zahájení nové hry. V této chvíli můžeme zobrazit informace o autorovi programu nebo později maximální skóre.



Na obrázku vidíte použitý nový prvek - **nadpis okna** (naleznete jej ve skupině **ovládání**, podskupina **dialogy**). Prvek má charakter textové proměnné (lze jej nastavovat i číst), představující text na horní liště okna (titulek okna). Prvek **sloučení textů** slouží ke sloučení více textů do jednoho textu. Sloučení probíhá shora dolů.

Běžný text je možné zadat zápisem textu u prvku **textová konstanta** T. Prvek **převod čísla na textový tvar** 123 převádí číslo na textové vyjádření čísla.

Podobně jsou sestaveny i větve pro ostatní délky hada. Všimněte si, že můžeme jednu větev větvící konstrukce použít pro více hodnot délek.

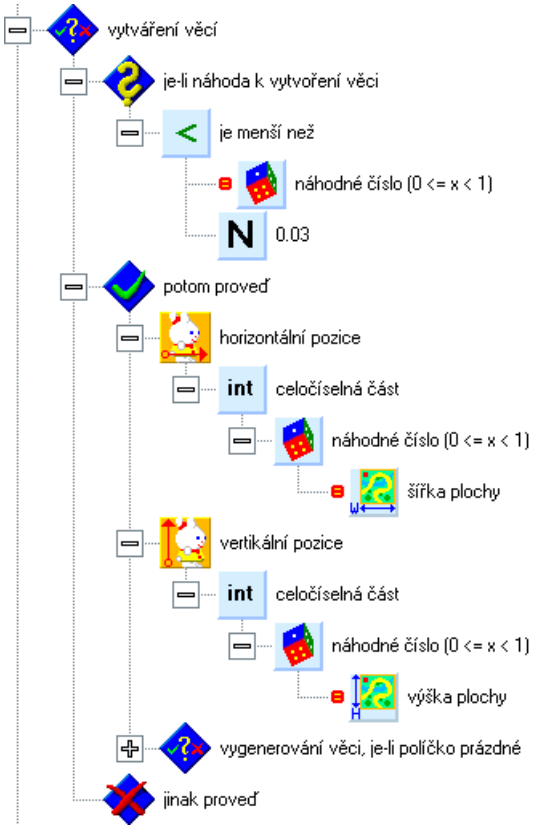


Jedna poznámka k používání nadpisu okna. Při startu programu se v nadpisu okna objeví jméno hlavní funkce programu. Není-li potřeba během chodu programu nadpis okna měnit, stačí patřičně upravit jméno hlavní funkce. U nového programu je jméno hlavní funkce přednastaveno na jméno programu.

Dále se budeme zabývat hlavní smyčkou programu. Na konec hlavní funkce umístěte **podmíněné opakování příkazů**. Do testu podmínky dejte prvek **ano**, čímž změníme cyklus na nekonečný cyklus. Ukončení programu bude uskutečněno uvnitř obsluhy kláves jako reakce na klávesu **Esc**. Na začátku hlavní smyčky vytvoříme nejdříve obsluhu kláves, následuje obsluha pohybu hada a na závěr obsluha vytváření

nových věcí (jídlo a jed). Posledním příkazem ve smyčce bude obsluha čekání zajišťující časování programu. Příkaz čekání můžete doplnit do smyčky již teď, ostatní obsluhy budeme doplňovat postupně.

Začneme obsluhou vytváření nových věcí. Před obsluhu čekání vložte **podmíněné provedení příkazů**, nazvěte ho **vytváření věcí**. Obsah vidíte na dalším obrázku.

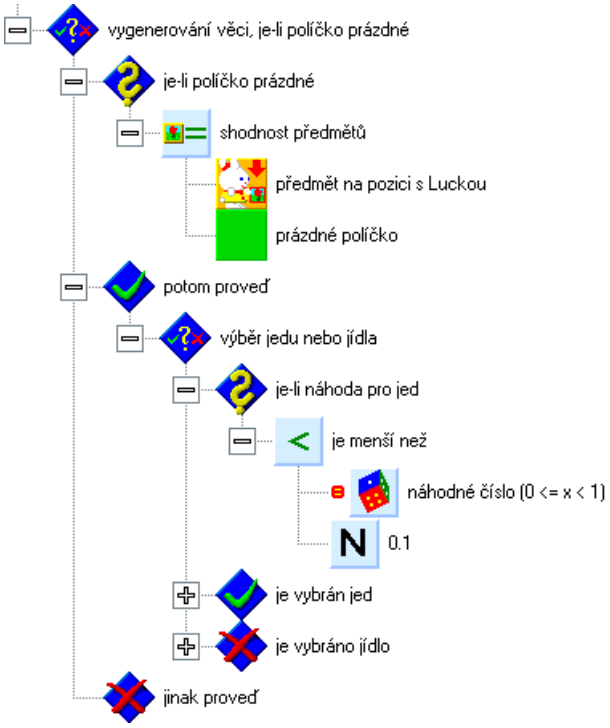


Novou věc budeme vytvářet náhodně s určitou pravděpodobností, proto do testu podmínky dáme porovnání náhodného čísla s konstantou. Nastavením hodnoty na **0.03** zajistíme, že se nová věc vytvoří v každých **3** ze **100** průchodů programu hlavní smyčkou, pravděpodobnost vzniku věci je tedy **3%**.

K vytvoření věci na ploše použijeme opět Petříkovu kamarádku - Lucku. Nejdříve ji nastavíme na náhodné políčko plochy. Pro horizontální pozici Lucky použijeme prvek **šířka plochy**, který nám předá číselnou hodnotu **20**, což je šířka plochy programu. Je možné číslo zadat přímo, ale výhodnější je pracovat s obecnými čísly. Ušetříme si tak problémy, až někdy v budoucnu budeme něco v programu měnit. Šířku plochy předáme funkci **náhodné číslo**, která vytvoří náhodné číslo v rozsahu od 0 do

19.999999999. Následující funkce **celočíselná část** int odstraní desetinnou část čísla. Vznikne tak náhodné celé číslo v rozsahu 0 až 19, což jsou horizontální souřadnice prvního až posledního políčka plochy. Připomeňme si, že se políčka číslují od nuly z levého dolního rohu směrem doprava a nahoru. Podobně vytvoříme i náhodnou pozici ve vertikálním směru.

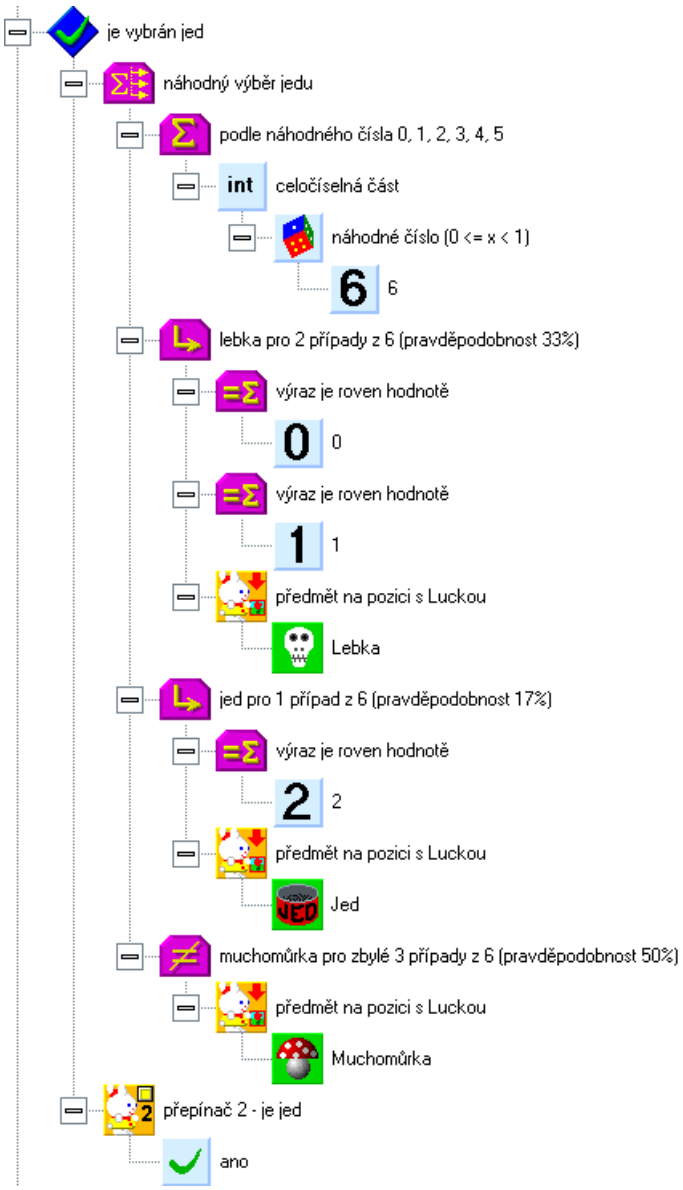
Za nastavením náhodné pozice následuje podmíněný příkaz, testující zda je nastavené políčko prázdné. Pokud ano, rozhodne se program dalším podmíněným příkazem, jestli vytvoří jed nebo jídlo. Náhodné číslo porovnáváme s hodnotou **0.1**, která zajistí, že se v **1** z **10** případů vytvoří jed, v ostatních případech jídlo. Vše vidíte na obrázku:



Na dalším obrázku vidíte obsluhu vytvoření nového jedu. Jedy budeme vytvářet s různou četností. Nejčastěji chceme vytvářet muchomůrku a nejméně často krabičku s jedem. Použijeme k tomu **vícestupňové větvení příkazů**. Větvícím výrazem bude náhodné celé číslo v rozsahu **0** až **5** vytvořené pomocí prvků **celočíselná část** int, **náhodné číslo** a číslice **6** 6.

V první větvi vytvoříme lebku pro 2 případy z 6 (hodnoty **0** a **1**), pravděpodobnost vzniku lebky bude tedy 33%. Ve druhé větvi vytvoříme krabičku s jedem s menší pravděpodobností - pro 1 případ z 6 (hodnota **2**), pravděpodobnost vzniku je 17%. V ostatních případech vytvoříme muchomůrku - pro zbylé 3 případy z 6 (hodnoty **3**, **4** a **5**), pravděpodobnost 50%.

Obsluhu vytvoření jedu zakončíme příkazem pro zapnutí **přepínače 2**. Jak si vzpomínáte, **přepínač 1** jsme použili k indikaci, že na políčku je zeď. Přepínač 2 budeme používat k indikaci jedu na políčku. To nám později usnadní práci, nebudeme muset testovat všechny možné jedy, které v programu použijeme.

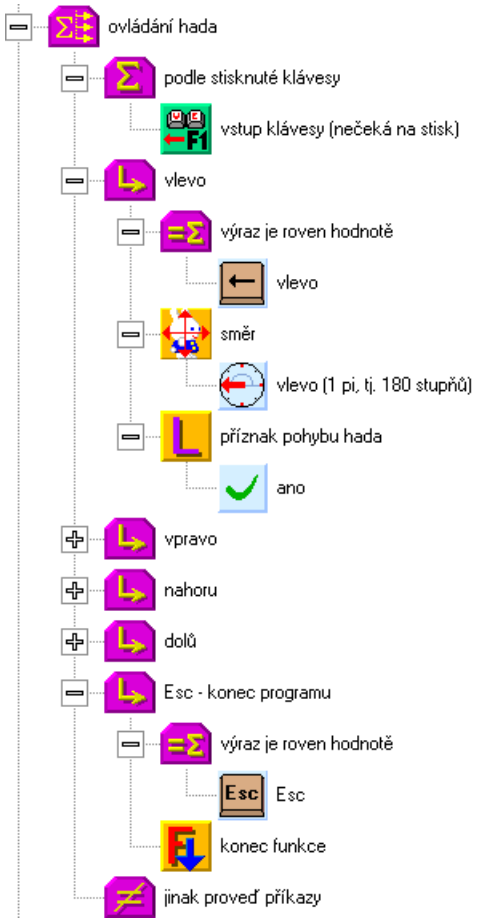


Obsluhu vytváření jídla zajistíme jinou metodou. Jak vidíte na obrázku na následující stránce, obsluha je díky použití seznamu celkem jednoduchá.

Na začátku obsluhy náhodně vybereme stránku seznamu, ze které převezmeme předmět jídla. Index jídla nastavíme na náhodné celé číslo v rozsahu **0** až **4**, což jsou indexy první až poslední stránky seznamu. Namísto přímého zadání čísla rozsahu náhodných hodnot použijeme prvek udávající počet stránek v seznamu [N]. Díky tomu můžeme v budoucnu přidávat do seznamu další jídla, aniž bychom museli měnit obsluhu jídel v programu. Číslo jídla si pomocí prvku **číselná hodnota 1 (rozsah 0 až 1023)** uchováme v políčku pro příští použití. Dalším příkazem položíme předmět jídla na pozici Lucky. Posledním příkazem bude zapnutí **přepínače 3**. Přepínač 3 použijeme k indikaci, že na políčku je jídlo.

Tím je obsluha vytváření nových věcí hotova a můžete ji vyzkoušet (nezapomeňte na obsluhu čekání na konci hlavní smyčky). Uprostřed plochy by se měla objevit hlava hada s otevírající se pusou a kolem vznikající nové věci. Program ukončíte kliknutím na křížek ⊠ na horní liště okna vpravo.

Pokračujeme obsluhou ovládání hada. Doplňte na začátek hlavní smyčky programu (před obsluhu vytváření věcí) prvek **vícestupňové větvení příkazů** s názvem **ovládání hada**. Obsluhu vidíte na následujícím obrázku.

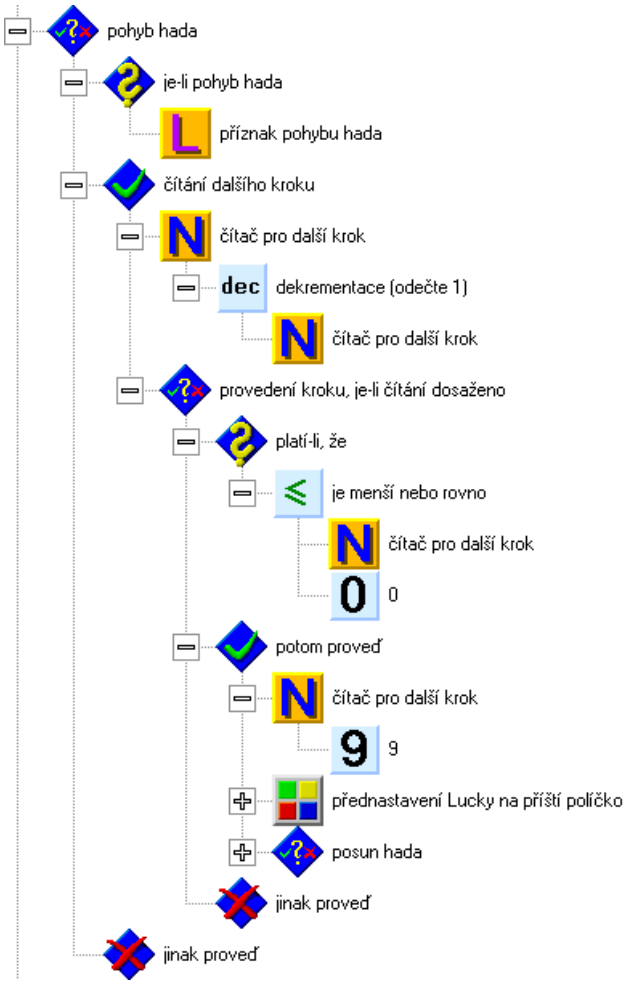


Větvící hodnotou bude funkce **vstup klávesy (nečeká na stisk)**. Klávesou **vlevo** otočíme Petříka (tedy hlavu hada) doleva a zapneme příznak, že had je v pohybu. Podobně obsloužíme i klávesy ostatních směrů. Jako poslední obsloužíme klávesu **Esc** a to tak, že příkazem **konec funkce** (ze skupiny **řízení programu**) ukončíme funkci. Protože se jedná o hlavní funkci programu, ukončíme tím i celý program.

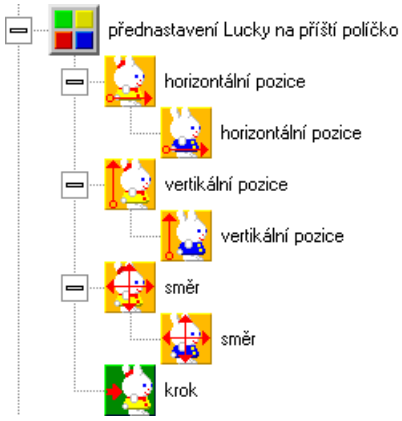
Program vyzkoušejte. Kurzorovými klávesami (šipkami) můžete otáčet hlavou hada do stran, klávesou **Esc** program ukončíte.

Naším dalším úkolem bude hada rozpohybovat. Rychlost pohybu hada zvolíme asi **2** kroky za sekundu. Abychom mohli lépe obsluhovat klávesnici a generování věcí, cykluje hlavní smyčka programu trochu rychleji (**18** průchodů za sekundu). Proto použijeme čítač pro další krok, který zajistí posun hada při každém devátém průchodu cyklem.

Obsluhu pohybu hada vidíte na dalším obrázku. Umístěte ji za obsluhu ovládání hada, ale před obsluhu generování nových věcí. Na začátku obsluhy je prováděn test, zda je had v pohybu (při zahájení hry totiž stojí na místě až do stisku první klávesy směru). Je-li tomu tak, dekrementuje se (tj. sníží o 1) čítač pro další krok. Dosáhne-li nuly, může had popojít o krok vpřed. Nejdříve znovu nastavíme čítač pro další krok. Zvolili jsme číslo **9**, které představuje **2** kroky za sekundu (základní časovou jednotkou počítače je **55** milisekund, za jednu sekundu tedy uplyne **18** časových jednotek).



Před obsluhou posunu hada otestujeme, zda není před hadem neprůchodná překážka. Za neprůchodnou překážku budeme považovat zeď, ale také samotného hada, tedy některý z jeho článků. K provedení testu použijeme Lucku. Nastavíme ji na pozici Petříka, otočíme v jeho směru a provedeme krok vpřed.



Jak si jistě vzpomínáte, zeď jsme označili **přepínačem 1**  . Abychom si to usnadnili, budeme tak označovat i políčko s článkem hada. Je-li přepínač zapnut, je před hadem zeď nebo článek hada a had umírá. Přehrajeme zvuk nárazu a provedeme animaci smrti hada. Budeme se jí zabývat až později. Jinak se had může posunout vpřed.

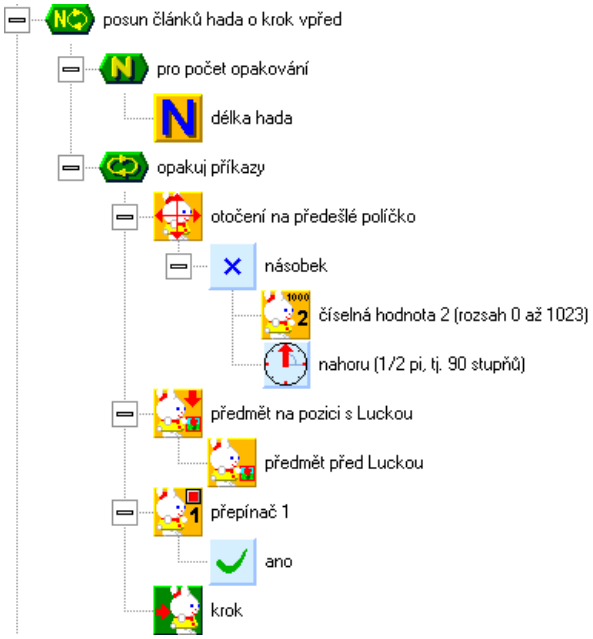
V případě posunu hada se nejdříve budeme zabývat Petříkem - hlavou hada. Zhruba uprostřed obsluhy posunu hada vidíte, že Petřík provede krok vpřed. Následující větvící konstrukce uschová na novém políčku číslo, udávající kterým směrem leželo minulé políčko. Toto číslo nám pomůže později při hledání cesty ke konci hada.

Číslo představující směr na minulé políčko uchováme v **číselné proměnné 2**  . Pro minulé políčko směrem vpravo použijeme číslo **0**, pro směr nahoru **1**, pro směr vlevo **2** a pro směr dolů **3**. Případ pro minulý směr vlevo vidíte na obrázku, ostatní směry jsou obdobné.

Vraťme se na začátek obsluhy posunu hada o krok vpřed. Lucku jsme naposledy zanechali na příštím políčku, kde testovala stav přepínače 1. Teď ji otočíme zpět a posuneme o krok vpřed, tím ji navrátíme na stejnou pozici jako je hlava hada. Ještě můžeme doplnit přehrátí zvuku kroku hada.

Následuje cyklus zajišťující posun článků hada, jak vidíte na dalším obrázku. Počet průchodů cyklu je dán délkou hada. Na začátku je Lucka nastavena na pozici hlavy hada. Na každém políčku s článkem nebo hlavou hada je uchováno číslo udávající směr na minulé políčko. Číslo představuje počet kvadrantů (tj. čtvrtin otáčky), o kolik se musí Lucka otočit proti směru hodinových ručiček od nulového úhlu (nulový úhel odpovídá směru doprava).

Nejdříve se tedy Lucka otočí na minulé políčko. Následujícím příkazem přenese předmět z minulého políčka na políčko, na kterém stojí. Tím posune článek hada o krok vpřed. Dalším příkazem zapne **přepínač 1**  . Řekli jsme si, že přepínačem 1 budeme indikovat nejen zeď, ale i článek hada. Posledním příkazem popojde Lucka o krok vpřed, to znamená na další políčko hada směrem k jeho konci.



Po vykonání všech průchodů cyklem zůstane Lucka stát na políčku za novým koncem hada. To platí i pro délku **0**, kdy se neprovede ani jeden průchod cyklem. Následuje posun hlavy hada o krok vpřed a úschova směru na minulé políčko, jak jsme si obsluhu popsali dříve. Vymažeme obsah políčka za koncem hada prázdným políčkem (provede Lucka) a vypneme příznak, že je na políčku článek hada. V případě nalezení jídla nebo jedu na políčko později doplníme nový článek hada.

Had je posunut. Teď si povšimneme obsahu políčka na nové pozici hlavy hada. Nejdříve vyzkoušíme, je-li na políčku jídlo. Jídlo jsme si označili přepínačem 3, budeme tedy testovat **přepínač 3**  . Obsluhu jídla vidíte na obrázku na předcházející stránce.

Je-li na políčku jídlo, vymažeme jídlo položením prázdného políčka a vypneme **přepínač 3**. V číselné proměnné políčka 1 máme uschováno číslo jídla ze seznamu jídel. Převezmeme tedy **číselnou hodnotu 1** políčka a použijeme ji jako index jídla, abychom získali článek hada, který k nalezenému jídlu patří. Vezmeme ze seznamu článek hada a položíme ho na pozici s Luckou. Jak víme, Lucka zůstala stát na políčku za koncem hada, před chvílí jsme tam prozatímně položili prázdné políčko. Tím jsme vytvořili nový článek hada se spolknutým jídlem. Ještě zapneme **přepínač 1**  jako příznak článku hada a zvýšíme délku hada. Na závěr můžeme přehrát zvuk polknutí a zobrazit novou délku hada.

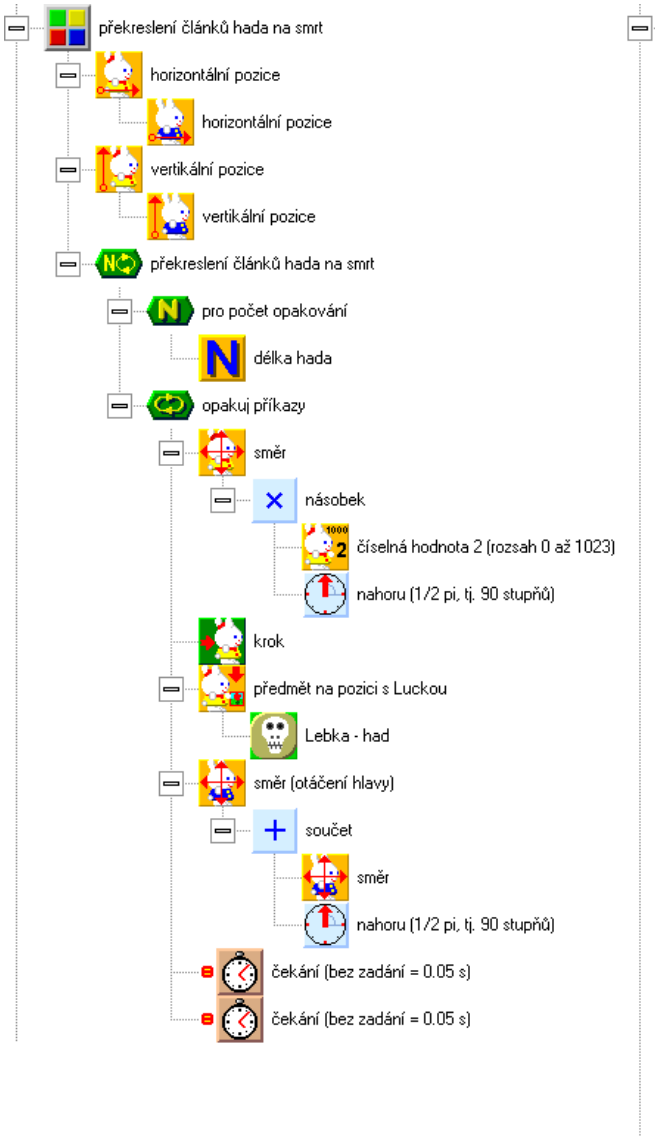
Obsluha nalezení jedu je podobná. Vidíte ji na obrázku níže. Jak víte, jed označujeme **přepínačem 2**. Zjistíme-li tedy testem přepínače 2 nalezení jedu, opět nejdříve vymažeme políčko položením prázdného předmětu a vypneme přepínač 2. Provedeme obsluhu podobně jako při běžném jídle. Nový článek hada použijeme vždy stejný, nezávislý na nalezeném jedu. Opět zapneme na pozici Lucky přepínač 1 jako indikátor článku hada, zvýšíme délku hada, přehrajeme zvuk polknutí a zobrazíme novou délku. Vše zakončíme smrtí hada.

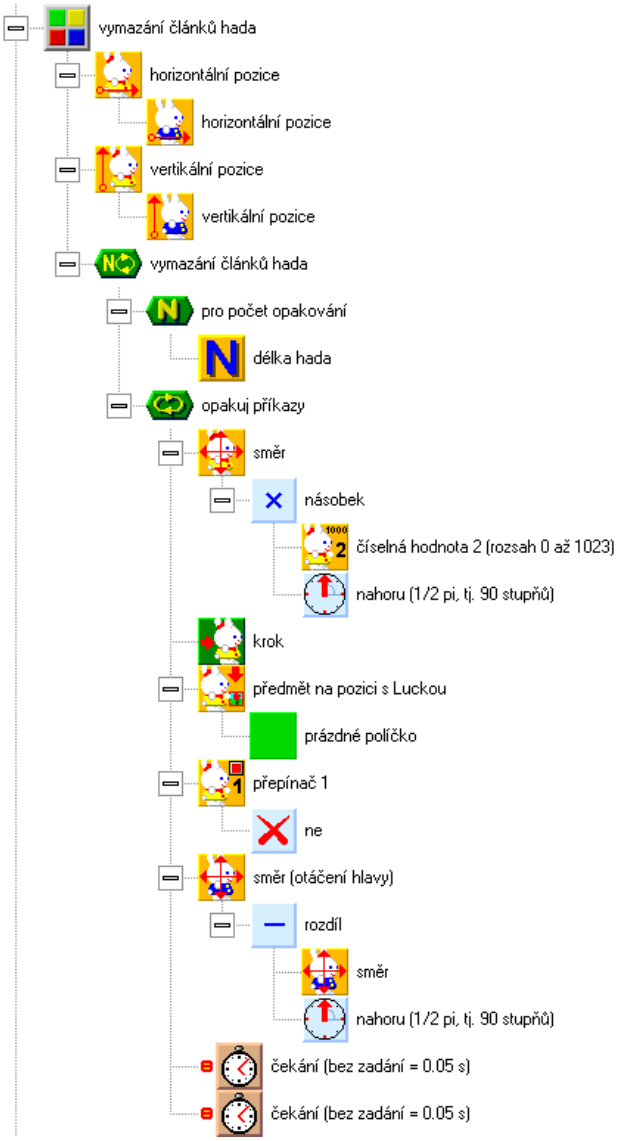
Naším posledním úkolem bude vytvoření funkce pro obsluhu smrti hada. Obsah funkce vidíte na dalším obrázku.

Nejdříve je uchována dosažená délka hada jako nové maximální skóre. Po kratičké pauze se ozve zvuk zasípění. Smrt budeme indikovat postupným překreslením všech článků hada na články s lebkou. Po překreslení a krátké pauze články hada vymažeme (znázorníme tak odumírání hada). Opět následuje krátká pauza a zahájí se nová hra.

Po zahájení hry je důležité doplnit příkaz vyprázdnění zásobníku kláves. Často se stává, že hráč v zápalu hry mačká klávesy i během ukončování hry. Tak by si mohl nechtěně odstartovat další hru.

Následující obrázky ukazují obsluhu překreslení článků hada na smrt a vymazání článků hada. Opět využíváme uschované směry na předchozí políčko v číselné hodnotě 2 políčka. Provádíme přitom animaci otáčení hlavy hada (hadovi se motá hlava), v prvním případě proti směru hodinových ručiček, v druhém případě obráceně.





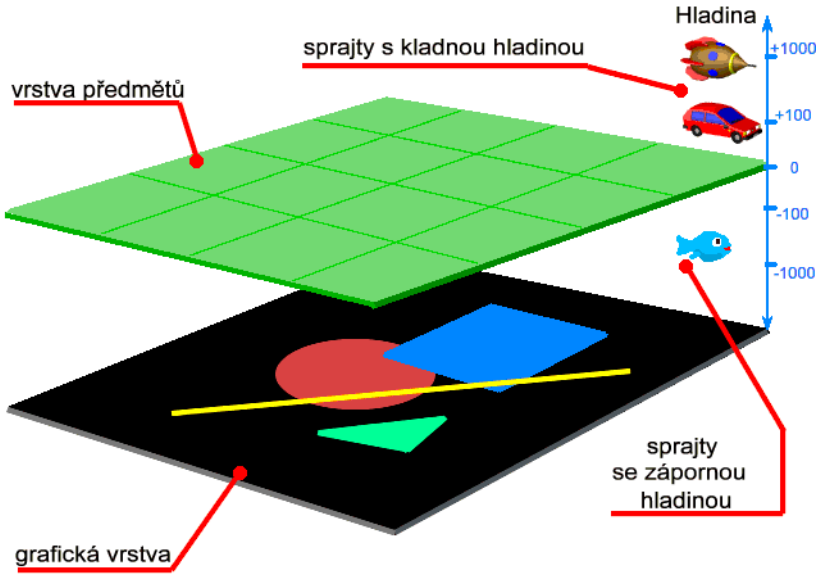
Teď již můžete program spustit a vyzkoušet. Přeji mnoho zábavy u vaší první opravdové hry.

13 Začínáme s grafikou

Dosud jsme programy vytvářeli pouze s využitím předmětů a sprajtů. Nyní se seznámíme s možná ještě zajímavější oblastí - s grafikou.

Grafika přináší neomezené možnosti tam, kde již předmětová animace nestačí. Umožní nám kreslení grafických prvků jako jsou body, čáry, kružnice, ale především vykreslování obrázků. Pomocí obrázků můžeme zobrazovat fotografie, pohybovat objekty, psát texty, kreslit tlačítka speciálních ovládacích prvků.

To, co pro práci s grafikou potřebujeme znát nejdříve, je kam a jak probíhá vykreslování grafiky. Podívejte se na následující obrázek.



Na obrázku vidíte znázorněné zobrazovací vrstvy okna programu v Petrovi. Základem je **vrstva předmětů**. S tou jste se již setkali v předešlých kapitolách. Je to vrstva, kam pokládáme předměty, nazýváme ji **hlavní plochou** programu. Nad vrstvou předmětů se pohybují sprajty s **kladnou** výškovou hladinou. Mezi ně patří Petřík a Lucka (pokud jejich hladinu nezměníme na zápornou). Pod vrstvou předmětů se pohybují sprajty se **zápornou** hladinou. Úplně nejnižší vrstvou je **grafická vrstva**. Do ní probíhá veškeré kreslení grafických prvků.

Vrstvy si můžete představit jako moře. Vrstva předmětů je hladinou moře. Po hladině se pohybují lodě (sprajty s výškou 0), nad nimi létají ptáci (sprajty s kladnou výškou, například 100). Pod hladinou se pohybují ryby (sprajty se zápornou výškou, například -100). Úplně dole je dno moře - grafická vrstva ke kreslení.

Při pohledu na okno programu se díváme na pomyslné moře shora. Běžně vidíme předměty na ploše a sprajty pohybující se nad předměty (Petřík, Lucka), ale nevidíme sprajty se zápornou výškovou hladinou a nevidíme ani grafickou vrstvu. Pro jejich zviditelnění musíme předměty zprůhlednit. Jak, to si ukážeme dále.

14 Kreslíme „mišmaš“

Nelíbí-li se vám slovo „mišmaš“ v nadpisu kapitoly, dosaďte si třeba „změť“. Ve slovníku ho možná nenajdete, ale zřejmě nejlépe vystihuje to, co teď budeme tvořit.

Vyzkoušíme si základní grafické příkazy pro kreslení. Vytvořte nový program s názvem **Grafika** nebo otevřete již hotový program v ukázkových programech Petra.

Petříka nebudeme v našem programu potřebovat, proto na začátku programu vypneme jeho viditelnost příkazem **viditelný** s parametrem **ne**. Základem programu bude cyklus **podmíněné opakování příkazů** s přerušením klávesou **Esc**. Přidejte ho za vypnutí viditelnosti Petříka a vložte do něj příkaz čekání.

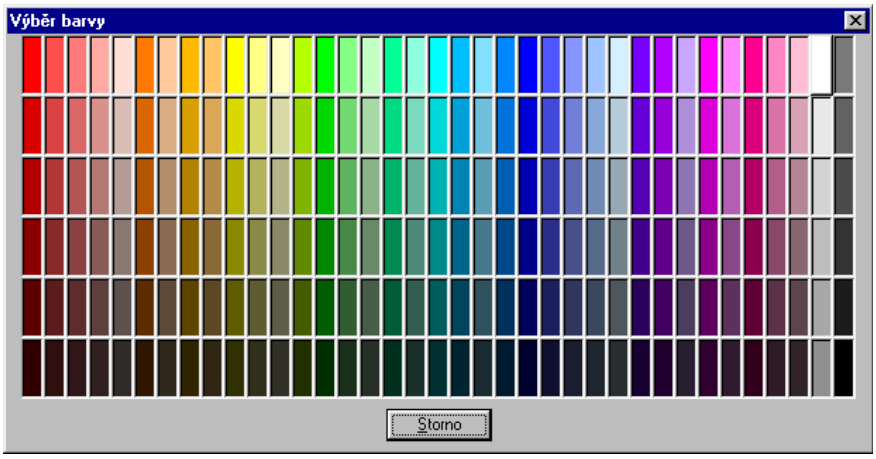
Dalším, velmi důležitým, úkonem je zprůhlednění políček ve vrstvě předmětů. Vyvolejte editaci předmětu **prázdné políčko**. V okénku volby barev vyberte barvu vlevo nahoře (našedle fialová). Je to **průhledná barva**, která zajistí zprůhlednění políčka s předmětem. V nástrojích pro kreslení vyberte **vyplněný obdélník** a překreslete celou plochu předmětu průhlednou barvou.

Když teď program spustíte, neuvidíte nic jiného než černou plochu. Vidíte grafickou vrstvu, která se stane základem našich dalších programů. V budoucnu budeme používat předměty i grafiku současně. Po vyzkoušení prvního grafického příkazu si můžete zkusit vkreslit něco do předmětu prázdného políčka. Uvidíte vykreslovanou grafiku a přes ni síť obrázků prázdných políček.

V okně **Základních prvků** si nalistujte skupinu **grafika**, podskupinu **kreslení**. Najdete v ní příkazy pro kreslení grafických prvků. Vezměte příkaz **bod** a vložte ho do hlavní smyčky programu. Příkaz obsahuje další čtyři prvky, sloužící k zadání parametrů vykreslovaného bodu.

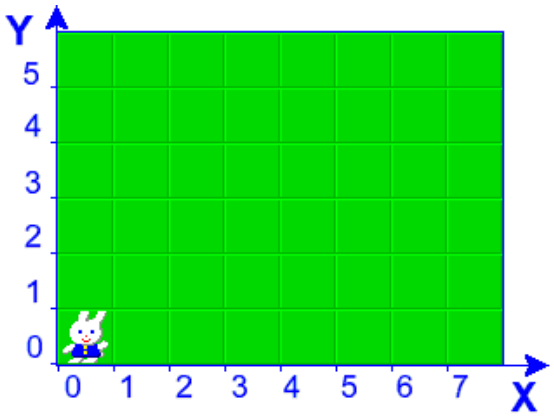
První parametr, **barva pera**, určuje barvu bodu. Barva je číslo, které můžeme uchovávat v číselné proměnné. Jeho hodnota nás zatím nezajímá, protože na určování barvy můžeme používat speciální funkce. U nově vytvořeného příkazu prvek **barva pera** obsahuje prvek **barva**. Je to **konstanta barvy**, předávající příkazu zvolenou barvu. Novou barvu vybereme dvojitým kliknutím levým tlačítkem myši na

prvek konstanty barvy. Objeví se okno volby barev, vybraná barva je označena vyvýšením políčka. Kliknutím na některou z barev vybereme jinou barvu. Uvidíme ji také v ikonce prvku konstanty barvy.



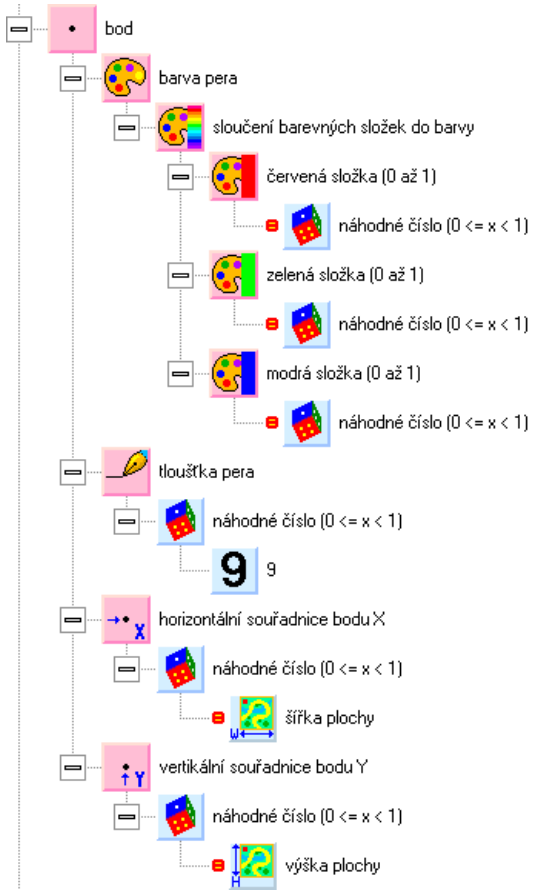
V našem programu budeme chtít vykreslovat body s náhodnou barvou. Vyhodíme proto pryč konstantu barvy a na její místo dosadíme prvek **sloučení barevných složek do barvy** (ze skupiny **grafika**). Prvek obsahuje další tři prvky, **červená složka (0 až 1)**, **zelená složka (0 až 1)** a **modrá složka (0 až 1)**. Prvky určují úroveň jednotlivých barevných složek ve výsledné barvě. Čím víc se hodnota složky blíží číslu 1, tím je složka světlejší. Například žlutá barva vznikne hodnotami 1/1/0. Náhodnou barvu vytvoříme tak, že do jednotlivých složek umístíme prvky **náhodné číslo (0 <= x < 1)** (rozsah náhodného čísla je také 0 až 1).

Dalším parametrem příkazu pro kreslení bodu je prvek **tloušťka pera**. Je to číselná hodnota, udávající kolik grafických bodů bude kreslený bod široký. Velikost bodu budeme generovat náhodně, proto sem vložíme náhodné číslo s parametrem **9**, čímž zajistíme náhodné body o velikosti 1 až 9 grafických bodů (nulou se nemusíme zabývat, příkaz pro kreslení bodu ji upraví na hodnotu 1).



Poslední dva prvky příkazu, **horizontální souřadnice bodu X** a **vertikální souřadnice bodu Y**, určují místo, kam se bod vykreslí. Na obrázku výše vidíte určování grafických souřadnic. Ničím se neliší od číslování políček, jak jsme ho poznali v předešlých kapitolách. Základní jednotkou souřadného systému Petra je **jednotkový krok**. Délka jednotkového kroku odpovídá šířce políčka, počátek souřadnic je vlevo dole. Díky vyjádření souřadnic desetinným číslem můžeme udávat souřadnice políček, grafiky i sprajtů jednotně, bez nutnosti vzájemných převodů.

Souřadnice k vykreslení bodu určíme náhodně, jak to již známe z dřívějších kapitol, pomocí náhodného čísla s parametrem šířky popřípadě výšky plochy. Výsledný příkaz pro náhodné nakreslení bodu bude vypadat takto:



Program spusťte. Na ploše se začnou náhodně objevovat barevné puntíky různých velikostí. Vykreslování není příliš rychlé. Po každém bodu program ve smyčce chvíli čeká (55 ms), rychlost vykreslování je proto pouze 18 bodů za sekundu.

Teď je vhodná chvíle pozastavit se u časování programu. Jak jsme si již dříve řekli, příkaz **čekání** má, kromě vlastní funkce čekání, také funkci spolupráce s ostatními programy (a také s jádrem systému Windows). Znalejší uživatel si může systémovým programem **Sledování systému** ověřit, že program **Grafika** zatěžuje počítač minimálně, a navíc je rychlost chodu programu nezávislá na rychlosti počítače. Pro upřesnění - příkaz čekání ve skutečnosti neznamená opravdovou prodlevu o zadané délce, ale synchronizaci na vnitřní hodiny počítače. Proto nezávisí rychlost programu na tom, jak rychle se provedou příkazy mezi dvěma příkazy čekání.

Doplňte na zkoušku do příkazu čekání hodnotu **0**. Po spuštění můžete pozorovat, že program teď chrlí velké množství bodů. Program již nečeká na časový krok, ale jede maximální rychlostí. Přitom po každém průchodu smyčkou ještě zajistí zobrazení plochy okna na displeji. To je další funkce příkazu čekání - **zajištění zobrazování**. Program totiž neví, ve které chvíli byly dokončeny všechny grafické operace a kdy je tedy vhodné okno zobrazit na displeji. Pokud by okno zobrazoval sám, mohl by je zobrazit ve chvíli, kdy je vykresleno pozadí hry, ale ještě ne postavy ve hře. To by se projevilo problikáváním pozadí pod postavami. Proto probíhá zobrazování na obrazovku ve chvíli čekání, kdy jsou zřejmě již všechny grafické operace dokončeny.

Tip pro optimalizaci programu: Program překresluje na obrazovku obdélníkový úsek části okna, kde byly provedeny změny. Chcete-li zvýšit rychlost programu, vykreslujte pouze změněnou část grafického výjevu. Opačným extrémem ovšem může být vykreslování velkého počtu malých částí výjevu, náročnější než vykreslení celého okna najednou.

Časování s hodnotou **0** je vhodné používat v situacích, kdy je buď potřeba vyšší výkon programu, nebo je časování po 0.055 sekundách příliš hrubé. Časování s **0** zajišťuje plynulé překreslování okna, maximální rychlost programu a ještě dostatečnou spolupráci se systémem Windows. Rychlost chodu programu teď již závisí na rychlosti počítače. Systémovým programem **Sledování systému** bychom zjistili, že program spotřebovává téměř celý výkon počítače a to i ve zdánlivě klidovém stavu.

Při další zkoušce časování programu můžete zkusit příkaz časování úplně zablokovat tlačítkem **Vypnout**  . Když teď program spustíte, uvidíte namísto teček barevný šum. V této chvíli již program pracuje skutečně na plný výkon bez jakýchkoliv prodlev. Možná si všimnete, že namísto plynulého barevného šumu je obraz trochu trhavý. Program totiž zajišťuje alespoň minimální překreslování okna na displej. Není-li po dobu asi 0.2 sekundy použit příkaz čekání, překreslí okno sám.

Vraťme se zpět ke grafickým příkazům. Přidejte do hlavní smyčky programu i další grafické příkazy - **čára**  , **obdélník**  , **vyplněný obdélník**  , **kružnice**  , **kruh**  , **koule**  , **trojúhelník**  - a zkuste jejich náhodné vykreslování. Pro kružnici, kruh a kouli použijte náhodný poloměr **2**. **Výplň**  vynechejte, zde by se moc neprojevila.

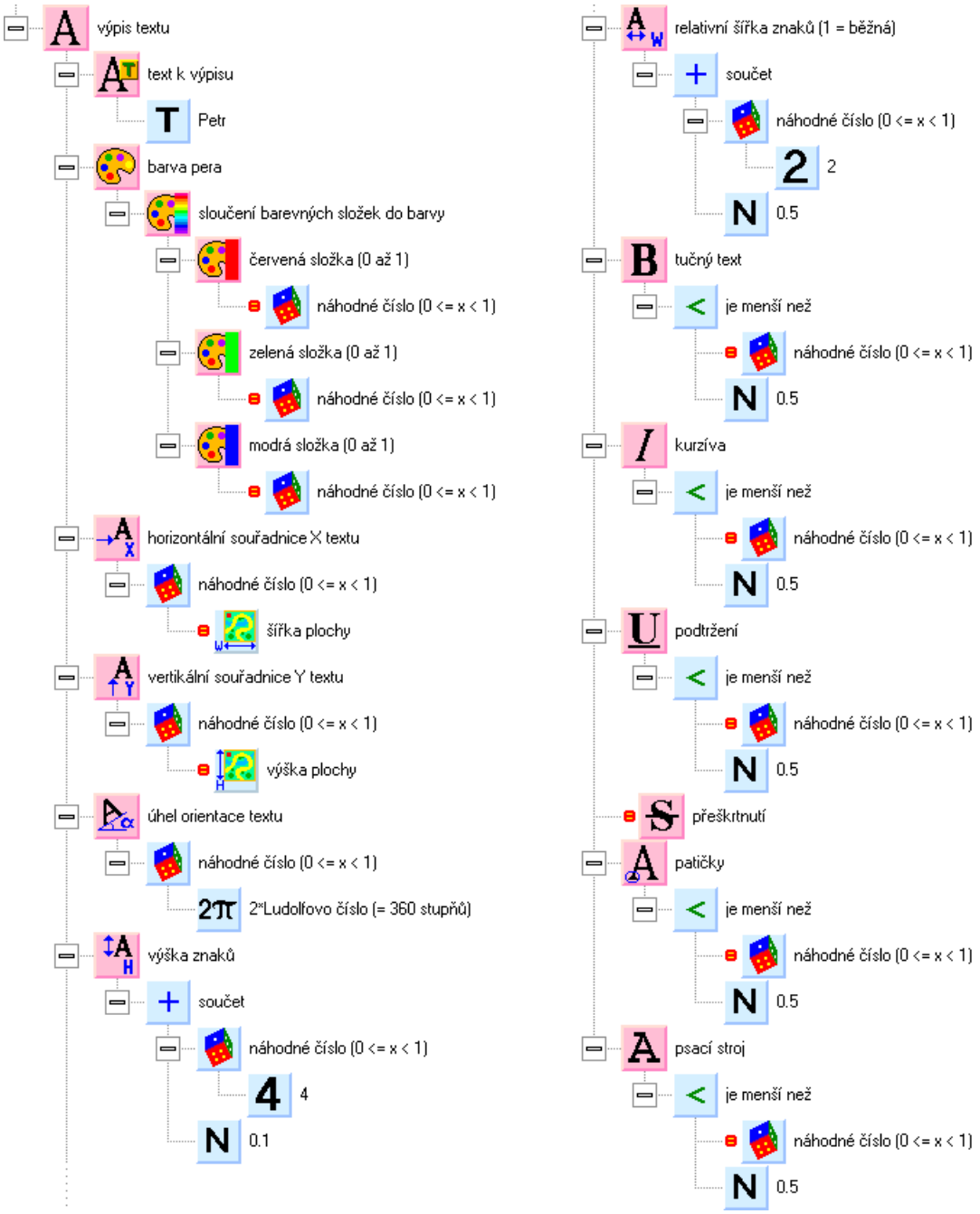
U příkazu **výpis textu**  se pozastavíme. Příkaz má více parametrů, než jste možná zvyklí, ale nemusíte se jich bát. Parametry, které vás nezajímají nebo které neznáte, nechejte prostě nenastavené, použijí se jejich implicitní hodnoty. Význam parametrů **text k výpisu**  , **barva pera**  , **horizontální souřadnice X textu**  a **vertikální souřadnice Y textu**  je zřejmě poměrně jasný.

Novým prvkem je **úhel orientace textu**  . Je to úhel, o který je text otočen kolem levého dolního rohu. Úhel se udává v radiánech, stejně jako všechny ostatní úhly v Petrovi. Můžeme využít konstanty směru známé z nastavování směru Petříka a Lucky. Bez zadání směru je použit směr **0** (vodorovně zleva doprava).

Prvek **výška znaků**  určuje výšku znaků textu. Je udán v jednotkových krocích. Bez zadání výšky bude výška znaků **0.5** (polovina výšky políčka).

Prvek **relativní šířka znaků (1 = běžná)**  představuje poměrné číslo, udávající kolikrát jsou znaky širší než jejich běžná šířka. Hodnota **1** nastavuje běžnou šířku znaků. Větší číslo než **1** nastaví znaky širší, při menším čísle budou použity znaky užší. Zvláštní hodnotou je číslo **0**. Číslo **0** nastaví doporučenou šířku použitého typu znaků. Přibližně tato šířka odpovídá běžné šířce, ale mírně se může lišit. Bez zadání šířky znaků je použita hodnota **0**, tedy doporučená šířka znaků.

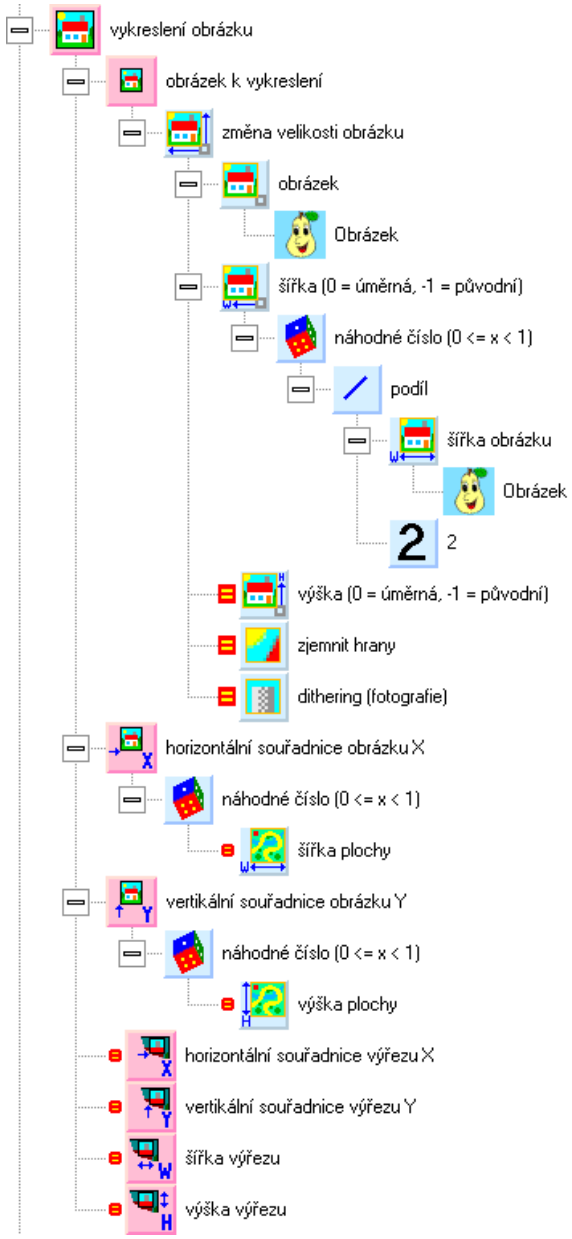
Prvky **tučný text** B, **kurzíva** I, **podtržení** U a **přeškrtnutí** S jsou logické přepínače a zapínají zvláštní efekty písma. Bez zadání hodnoty jsou všechny přepínače vypnuté. Přepínač **patičky** A zapíná patičky u znaků. Bez zadání je vypnutý. Přepínač **psací stroj** A zapíná znaky s pevnou roztečí a pevnou šířkou. Při vypnutém přepínači mají znaky proměnlivou šířku (**I** je užší než **M**). Bez zadání je přepínač vypnutý.



Celý příkaz pro náhodný výpis textu vidíte na předcházejícím obrázku. Příkaz je dost dlouhý, proto je obrázek rozdělen na dvě části. Po vytvoření příkazu program zkuste.

Jednou z nejsilnějších stránek grafiky jsou zřejmě obrázky. Pomocí obrázků můžete zajistit vše, na co nestačí základní vybavení Petra. Vytvoříte pohyblivé postavičky, okno s animovanými tlačítky, pohyblivé pozadí hry a samozřejmě fotografie.

Příkazy a funkce pro práci s obrázky najdete ve skupině **grafika**, podskupina **obrázky**. Základním příkazem je příkaz **vykreslení obrázku**. Kromě samozřejmých parametrů, jako je obrázek k vykreslení a souřadnice obrázku, má příkaz ještě další parametry, umožňující vykreslit pouze určitou část obrázku.



V našem zkušebním programku budeme náhodně vykreslovat nějaký obrázek. Než se pustíte do jeho kreslení, zkuste se podívat do **Knihovny proměnných a funkcí**. S Petrem dostáváte i velké množství obrázků. Otevřete skupinu **[vzory]\Kresby**. Zde jistě najdete obrázek, který se vám bude líbit. Při prohlížení obrázků vyberte kliknutím myši první obrázek ve skupině a potom popojíždějte pomocí kurzorových kláves výběrovým kurzorem dolů. V editačním okně uvidíte náhledy na jednotlivé obrázky.

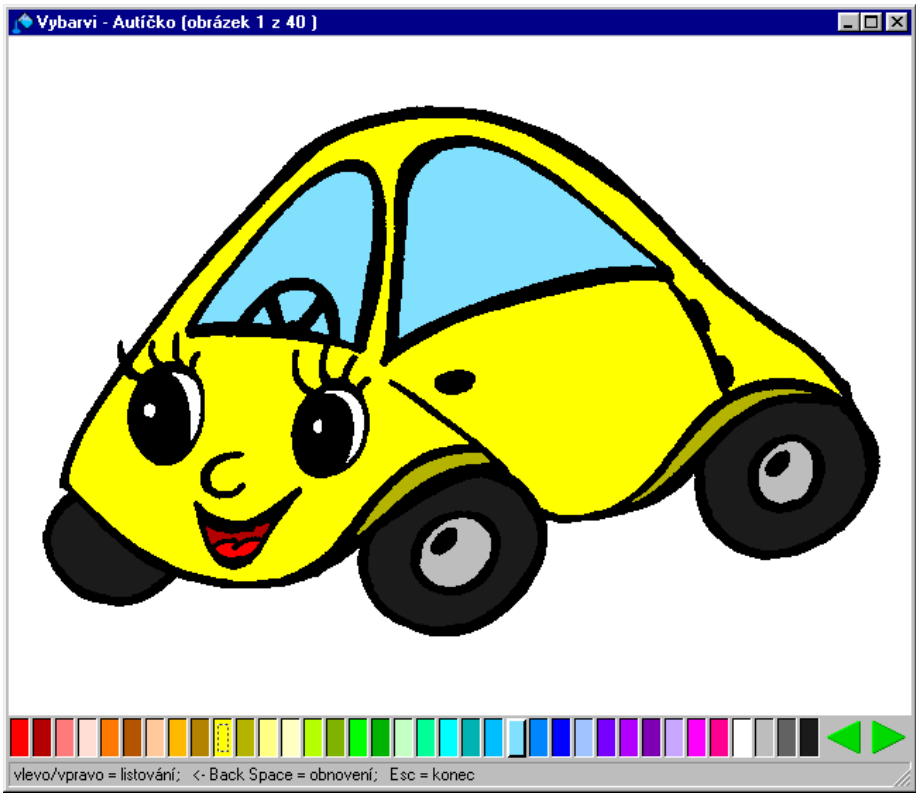
Vybraný obrázek přetáhněte do **Společných proměnných a funkcí**. Je-li to obrázek s jednotnou barvou pozadí, můžete vylít pozadí obrázku průhlednou barvou (použijte funkci **Výplň** ), tím zajistíte zobrazení obrázku bez pozadí.

Příkaz pro náhodné vykreslení obrázku vidíte na předcházející stránce. Pro vykreslení obrázku s náhodnou velikostí použijeme funkci **změna velikosti obrázku**  . Nová šířka obrázku bude náhodná a bude vycházet z původní šířky obrázku zmenšené asi na polovinu. Novou výšku obrázku nemusíme zadávat, bude automaticky úměrná nové šířce.

Poznámka: Používáme raději větší obrázek a vyšší stupeň zmenšení. Dosáhneme tak vyšší kvality výsledného obrázku než při použití obrázku s menší velikostí.

15 Vybarvi si sám

Naším dalším úkolem na cestě za poznáním světa králíka Petra bude vytvoření grafického editoru umožňujícího vybarvování připravených obrázků. Naučíme se přitom pracovat s myší a se soubory. Naše představa o programu je následující. Program vyhledá ve své složce všechny obrázky (soubory BMP). Uživatel programu může mezi obrázky listovat a vylíváním ploch je vybarvovat (černá barva je přitom rezervována pro obrysy obrázků).



Řekněme si něco o souborech a složkách. **Soubor** je samostatný „kus dat“ uložený v počítači. Může to být obrázek, dopis, tabulka, program. Uživatelské datové soubory často označujeme termínem **dokumenty**. **Složka** je, dá se říct, balík souborů. V Petrovi je obdobou složky **skupina**  . Chceme-li vyjádřit, kde na disku leží některý soubor či složka, používáme k tomu označení zvané **cesta**. Cesta je výpis složek (případně i disku), přes které musíme projít při cestě za souborem (složkou). Oddělujeme je znakem zpětného lomítka „\“, např. **C:\Program Files\Petr\Peter.exe**. Všimněte si, že disk označujeme písmenem a dvojtečkou. Na konci výpisu vidíte jméno programu. Tento výpis nazýváme **plné jméno souboru**. Obsahuje disk, složky, jméno souboru a příponu jména souboru. **Přípona jména souboru** je část za tečkou a udává typ souboru. Například příponou **EXE** označujeme programy.

Nejdříve založte nový program s názvem **Vybarvi**  , popřípadě otevřete již hotový program z ukázkových programů Petra.

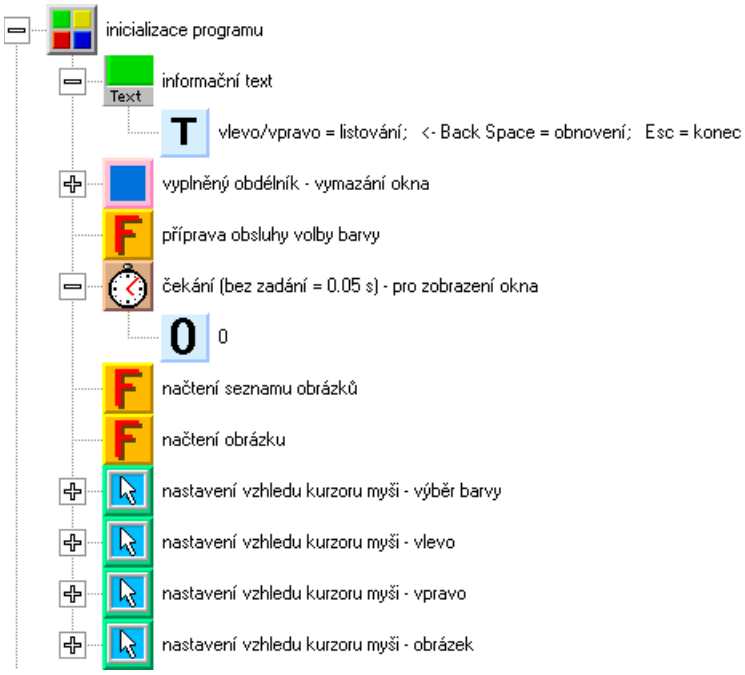
Upravíme velikost plochy. Máme představu, že budeme editovat obrázky se standardní velikostí 20 x 15 políček (640 x 480 grafických bodů). Na volbu barvy a obrázku potřebujeme dole v okně pás o výšce 1 políčka. Nastavte proto velikost hlavní plochy programu na 20 x 16 políček. Je to ještě vyhovující velikost plochy pro režim obrazovky

800 x 600 bodů. Tento režim obrazovky (videomód), ale raději větší, bychom měli doporučovat i uživatelům našich programů jako minimální vhodný.

Vytvořte dva nové předměty - **šipka vlevo** a **šipka vpravo**. Nakreslete do nich obrázky ovládacích tlačítek pro listování vlevo a vpravo, podklad bude šedý. Položte předměty do pravého dolního rohu plochy tak, jak to vidíte na předešlém obrázku.

Vytvořte hlavní smyčku programu - nekonečný **podmíněný cyklus** (v podmínce cyklu bude prvek **ano**). V těle cyklu bude jeden příkaz čekání.

Před hlavní smyčku připravte skupinu s názvem **inicializace programu**. Skupina bude obsahovat přípravné operace nutné pro běh programu.



Prvním příkazem bude nastavení textu nápovědy do stavové lišty na spodním okraji okna. Neobsahuje-li stavová lišta žádný text, je vypnutá. Chceme-li používat v programu stavovou lištu, je dobré ji zapnout ihned na začátku programu. Je trochu nehezké, když se objeví nejdříve okno programu, a až po chvilce naskočí stavová lišta.

Tip: Prázdnou stavovou lištu zajistíte zobrazením znaku *mezera*.

Druhým příkazem je vyplnění plochy okna bílou barvou. V příkazu **vyplněný obdélník** uvedeme jediný parametr - bílou barvu obdélníku. Ostatní parametry nebudeme zadávat, tím zajistíme vyplnění celé plochy okna. Tento zdánlivě zbytečný příkaz zajistí, aby během načítání seznamu obrázků neproblikl před zobrazením prvního obrázku černý podklad. Důvody jsou pouze estetické, avšak nikdy nezanedbávejte podobné přechodové stavy programu. Ze všech těchto drobných negativních dojmů si uživatel vytváří celkový vztah k programu. Problikává-li občas někde něco co nemá, získává uživatel k programu nedůvěru. Stejně tak, když program dostatečně rychle nezareaguje na povel od uživatele nebo když uživateli není hned jasné jeho ovládání. Či snad dokonce když intuitivní příkaz má nežádoucí a neočekávané následky.

Třetím příkazem je funkce **příprava obsluhy volby barvy**. Nadefinuje použité barvy pro kreslení a zobrazí políčka s barvami. Zatím připravte pouze prázdnou funkci.

Následuje příkaz čekání s parametrem **0**. Proč zde je? Jak víte, příkaz čekání zajišťuje zobrazení okna. Při startu programu není okno programu ihned zapnuto. Program čeká se zapnutím okna na první příkaz čekání, kdy předpokládá, že obsah okna je již připraven. Bez této vlastnosti by nám při startu programu problikávaly na displeji různé přechodné děje. Uvedený příkaz čekání s parametrem 0 zajistí zapnutí okna ihned po startu. Následující vyhledávání souborů chvilku trvá. Bez příkazu čekání by se okno zapnulo až po chvilce, po vyhledání všech souborů. Není to sice závada, ale pokud se okno programu neobjeví dostatečně brzo, uživatel může mít pochybnosti, zda program vůbec nabíhá, a zkusí ho spustit znovu.

Funkce **načtení seznamu obrázků** vyhledá všechny soubory BMP s obrázky. Funkce **načtení obrázku** načte do programu aktuální obrázek a zobrazí jej. Zatím připravte jen prázdné funkce. Poslední čtyři příkazy slouží k předefinování vzhledu kurzoru myši, o nich až později.

Spustíte-li teď program, uvidíte bílou plochu, vpravo dole dvě tlačítka a nápisy v nadpisu okna a ve stavové liště. Zatím nic moc, ale jsme teprve na začátku.

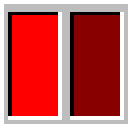
V okně **Společných proměnných a funkcí** vytvořte nový seznam s názvem **seznam barev**. Velikost seznamu nastavte na **36**, to bude počet barev použitých v editoru. Ukazatel v seznamu přejmenujte na **index barvy** a automatický přírůstek na **inkrementace indexu barvy**. Do seznamu přidejte jeden číselný datový prvek **barva**.

Vytvoříme obsah funkce **příprava obsluhy volby barvy**. Přepněte se do ní dvojklikem myší na ikonu funkce. Úplný obsah funkce vidíte na obrázku na další stránce. Po pravé straně je rozvinutý cyklus pro vytvoření políček barev.

Nejdříve naplníme seznam barev. Nastavte index barvy na **0** a inkrementaci indexu barvy na **1**. Tím zajistíme, že se barvy budou do seznamu ukládat od začátku a po každém nastavení barvy se ukazatel seznamu automaticky zvýší o 1. Následují příkazy nastavení barev. Zvolte **36** barev dle vlastního výběru (ne černou - tu použijeme pro obrysy obrázků). Proč zrovna 36? Pro volbu barvy budeme mít k dispozici 18 políček a na každém políčku budou 2 barvy.

Políčka volby barev zobrazíme pomocí předmětů. Předměty jsou v této chvíli vhodnější než grafické vykreslení, protože při vylévání ploch by nám zřejmě mohlo natéct i do okének s barvami. Namísto ručního kreslení předmětů použijeme vytvoření programové. Je to snazší než upravovat 36 okének a především lze snadněji udržet soulad mezi okénky a skutečnými barvami, pokud barvy v programu někdy změníme.

Vytvořte si ve **Společných proměnných a funkcích** obrázek (ne předmět) **okénko volby barev** o velikosti 1x1 políčko. Nakreslete do něj okénka pro volbu 2 barev, asi takto:



Předměty budeme vytvářet tak, že obrázek vykreslíme do okna programu, vyplníme barvami, sejmeme z okna jako předmět a uložíme do plochy. K ukládání do plochy použijeme Lucku. Proto ji nastavíme na první políčko (horizontálně na 0) a otočíme doprava. Následuje cyklus pro všechny předměty barev (polovina počtu barev).

Prvním příkazem v cyklu je vykreslení obrázku okénka volby barev. V příkazu uvedeme pouze jediný parametr - obrázek k vykreslení. Nejsou-li ostatní parametry uvedeny, vykreslí se obrázek v levém dolním rohu okna.

Druhým příkazem je vylití levého okénka barvou ze seznamu barev. Souřadnice místa volíme ve středu okénka. Stejně tak vylijeme pravé okénko druhou barvou, opět do středu druhého okénka. Pamatujte, že ukazatel barev se posouvá automaticky.

Vytvořený obrázek předmětu sejmeme z okna funkcí **úschova výřezu z plochy jako předmět** a uložíme ho na pozici s Luckou. Potom Lucka popojde na další políčko.

Po vytvoření všech políček barev změňte inkrementaci indexu barvy na **0**, od této chvíle již automatickou inkrementaci nebudeme v programu používat.

Připadá vám, že děláme v okně příliš mnoho operací a že snad takové operace musí být viditelné? Ve skutečnosti jsou tyto přípravné vykreslovací operace velmi rychlé a my máme dost času, než se obsah okna vykreslí na displeji prvním příkazem čekání. Vykreslený obrázek zůstane skryt pod prvním předmětem volby barev, takže se dokonce ani nemusíme starat o závěrečné vymazání obrázku.

Teď můžete program spustit. Je-li vše v pořádku, objeví se na spodní straně okna pás s barevnými okénky volby barev.

V okně **Společných proměnných a funkcí** připravte seznam pro uchování jmen všech nalezených souborů (podle obrázku níže). Dejte mu jméno **seznam jmen obrázků**.

Velikost seznamu nastavte na 1000. Je to možná příliš velká rezerva, ale z hlediska zaplnění paměti je to poměrně nepodstatné. Obecně každá proměnná seznamu zabere v paměti 4 bajty, pouze číselná proměnná 8 bajtů a logická 1 bajt. Náš seznam tedy zabere 4000 bajtů. Můžeme počítat s volnou pamětí až jednotky milionů bajtů.

Ukazatel v seznamu pojmenujte **index obrázku**, automatický přírůstek ukazatele přejmenujte na **inkrementace indexu obrázku**. Do seznamu přidejte textovou proměnnou s názvem **jméno obrázku.**

Vedle seznamu obrázků vytvoříme ještě číselnou proměnnou s názvem **počet obrázků**. Seznam zřejmě nezaplníme celý, proto bude tato proměnná uchovávat skutečný počet obrázků v seznamu.

Pustíme se do funkce **načtení seznamu obrázků**. Máte-li otevřen ukázkový program **Vybarvi**, můžete v něm vidět trochu složitější funkci. V ukázkovém programu je pro podporu víceuživatelského prostředí zajištěno načítání souborů i z domovské složky programu. My se spokojíme s jednodušší variantou a budeme soubory s obrázky načítat pouze z aktuální složky.

Připravíme si místní proměnné funkce. Jména nalezených souborů budou uložena do textové proměnné **seznam jmen nalezených obrázků**. Seznam abecedně setřídíme, pro tento účel budeme potřebovat ještě dvě textové proměnné - **jméno jednoho obrázku** a **jméno jednoho obrázku 2**.

Na dalším obrázku vidíte obsah funkce. Ve funkci načteme jména souborů obrázků z aktuální složky, přidáme jména souborů do seznamu a nakonec seznam jmen obrázků abecedně setřídíme.

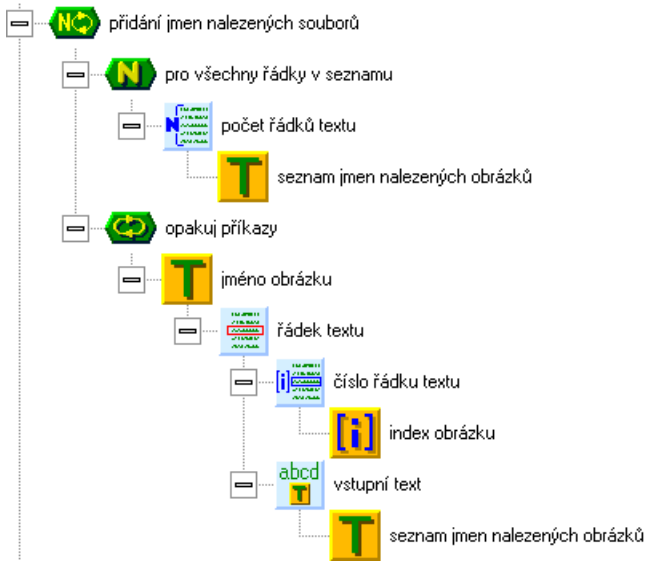
K vyhledání souborů ve složce slouží funkce **seznam souborů**  . Jako parametr funkce zadáváme v textové podobě specifikaci souborů k vyhledání. Při zadání se používá tzv. hvězdičková konvence (otazník „**?**“ znamená libovolný znak, hvězdička „*****“ představuje libovolnou skupinu znaků). Lze zadat více specifikací, jednotlivé specifikace se oddělují znakem středníku „**;**“. Například všechny soubory s obrázky

BMP vyhledáme zadáním textu ***.bmp**, všechny soubory BMP i JPG (to je také formát obrázků podporovaný Petrem) vyhledáme zadáním textu ***.bmp;*.jpg**.

Funkce **seznam souborů** navrací víceřádkový text, ve kterém každý řádek obsahuje jméno jednoho nalezeného souboru včetně přípony jména souboru (ne ale cestu k souboru). Například při hledání souborů BMP bychom mohli dostat tento text:

Autíčko.bmp
Beran.bmp
Býk.bmp
Domeček.bmp
Drak.bmp

Jména jednotlivých souborů přidáme do seznamu jmen obrázků. Nejdříve nastavíme **index obrázku** na **0** a **inkrementaci indexu obrázku** na **1**, využijeme automatické zvyšování indexu seznamu. Jako číslo řádku můžeme použít index obrázku, index se automaticky zvýší až po uložení jména.



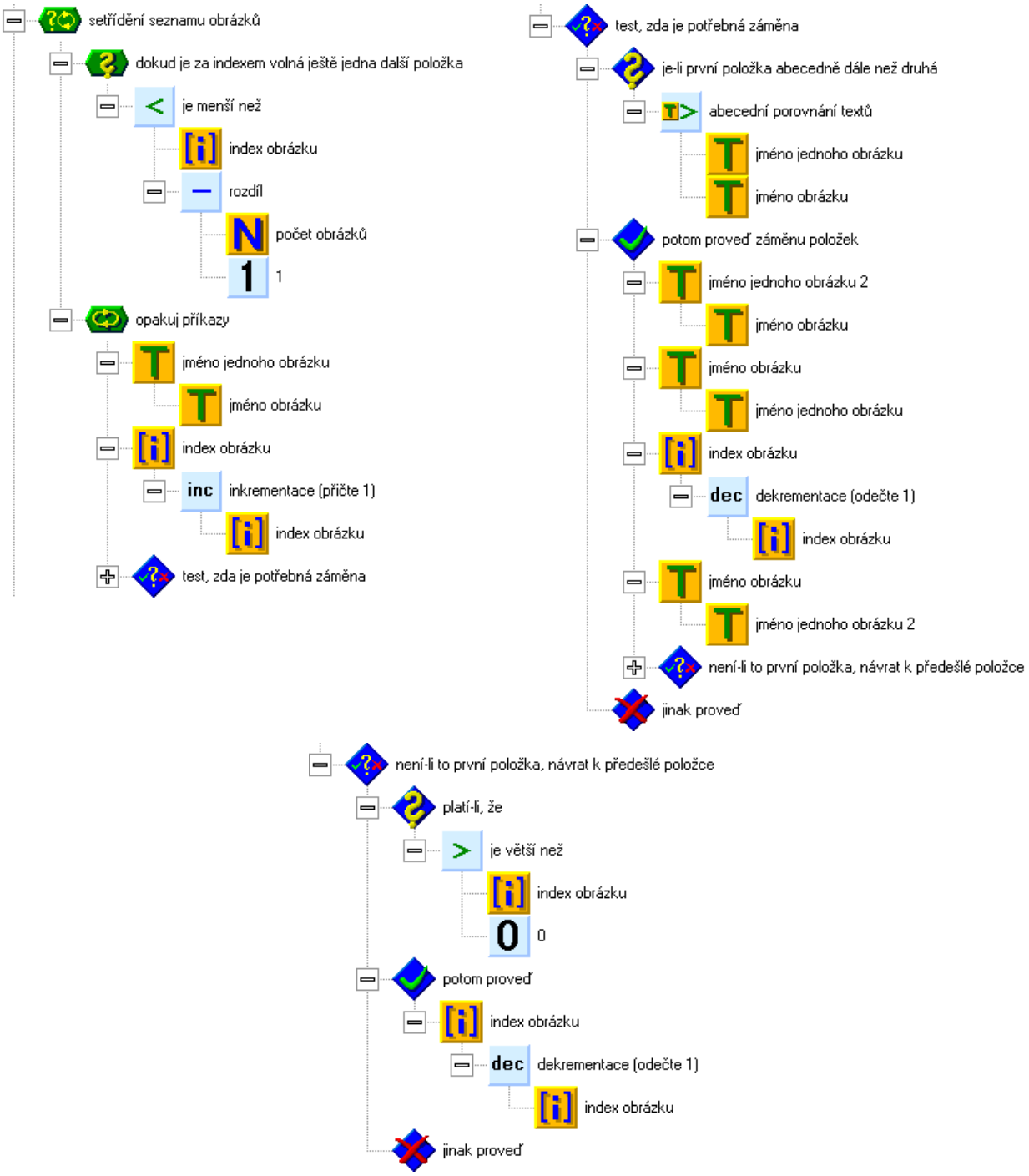
Po uložení jmen všech souborů do seznamu použijeme konečný stav indexu obrázku k nastavení proměnné **počet obrázků** a vynulujeme inkrementaci indexu obrázku.

Následuje abecední setřídění seznamu jmen obrázků. Budeme postupovat od začátku seznamu ke konci a porovnáme vždy dvě sousední jména. Narazíme-li na nesetříděný pár, jména zaměníme a navrátíme se k předešlému páru, abychom zajistili případný odsun prvního jména směrem dolů k začátku seznamu.

Třídění proběhne v podmíněném cyklu. V podmínce cyklu testujeme, zda je ještě k dispozici další pár jmen. Na začátku cyklu uchováme první jméno do pomocné textové proměnné **jméno jednoho obrázku**. Zvýšíme ukazatel v seznamu jmen na další jméno a zjistíme, zda je první jméno abecedně dál (abecedně vyšší) než druhé jméno. Pokud ano, provedeme záměnu jmen.

Při záměně jmen uchováme do pomocné proměnné **jméno jednoho obrázku 2** druhé jméno (na které je nyní nastaven ukazatel seznamu). Na jeho místo uložíme uschované první jméno. Snížíme ukazatel zpět k prvnímu jménu a uložíme původní druhé jméno na pozici prvního. Tím je záměna jmen ukončena.

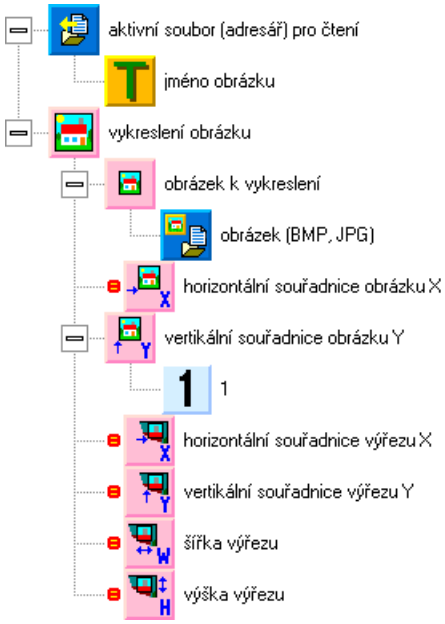
Nakonec snížíme ukazatel v seznamu, abychom zajistili případné další odsunutí prvního jména směrem k začátku seznamu, není-li ještě správně zatříděno.



Dobře si metodu třídění prohlédněte a ujasněte její funkci. Nepatří mezi nejrychlejší, ale při své jednoduchosti je dostatečně dobrá.

Posledním příkazem ve funkci **načtení seznamu obrázků** je nastavení ukazatele seznamu jmen obrázků na hodnotu **0**. To bude aktuální obrázek k zobrazení.

Vytvoříme obsah funkce **načtení obrázku**. Je poměrně jednoduchá, jak vidíte na obrázku níže. Potřebujeme k tomu pár prvků pro práci se soubory ze skupiny **soubory**. Nejdříve určíme **aktivní soubor pro čtení** podle jména obrázku ze seznamu jmen obrázků. Potom uvedeme prvek **obrázek** (podskupina **data**) v příkazu pro vykreslení obrázku. To je vše.

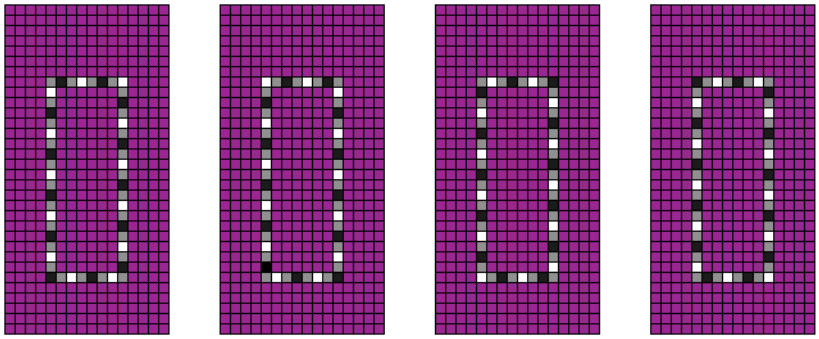


Možná vás napadlo, co se stane, když nebude nalezen ani jeden soubor s obrázkem. V tom případě bude jméno obrázku obsahovat prázdný text a funkce pro čtení či ukládání obrázku nic neprovedou. V ukázkové verzi programu je navíc doplněn výpis textu informujícího uživatele, že nebyl nalezen žádný obrázek.

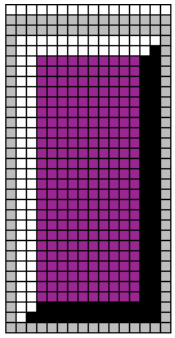
V této chvíli je již vhodné připravit si pár obrázků. Stačí alespoň jeden. Můžete k tomu použít grafický editor Petra. Velikost obrázku nastavte na rozměry **20x15** kroků (tj. **640x480** bodů). Obrysy kreslete černou barvou. Dbejte na to, aby čáry ohraničovaly uzavřené plochy, jinak vám barva „vyteče“. Po nakreslení obrázku uložte obrázek do **Knihovny proměnných a funkcí** a pomocí **Průzkumníka** obrázek přeneste (ze složky typicky **C:\Dokumenty\Petr\Picture**) do stejné složky, jako je program **Vybarvi**. Můžete též použít vzorové obrázky Petra, najdete je ve skupině **[vzory]\Kresby**.

Máte-li ve složce programu **Vybarvi** připravený nějaký obrázek, můžete program na zkoušku spustit. Abecedně první z obrázků by se měl zobrazit v okně.

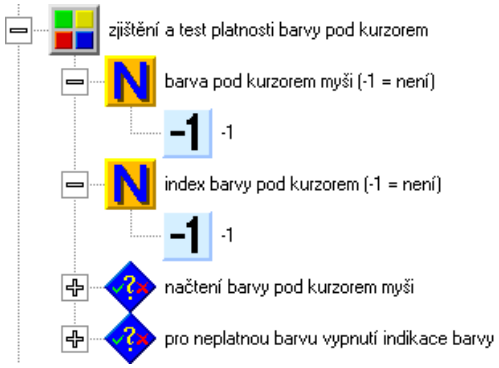
V programu budeme používat Petříka jako indikátor, která barva je právě zvolená. Vyvolejte editaci sprajtu Petříka (dvojklikem na ikoně **Petřík** v okně **Společných proměnných a funkcí**). Změňte parametry sprajtu (tlačítkem **Vlastnosti**) na následující hodnoty: prodleva mezi fázemi = **110**, fází na krok = **0**, fází pro klid = **4**, fází pro pohyb = **0**, směrů = **1**, šířka obrázku = **0.5**. Vyhoďte obrázky Petříka ze sprajtu a dvojklikem do prvního obrázku vyvolejte editaci obrázku. Do obrázku nakreslete rámeček indikátoru, využijte střídající se barvy bílá-šedá-černá-šedá. Zkopírujte obrázek do dalších políček sprajtu a posuňte body vždy o 1 bod ve směru hodinových ručiček. Při zkoušce sprajtu uvidíte, jak rámeček „teče“ po obvodě.



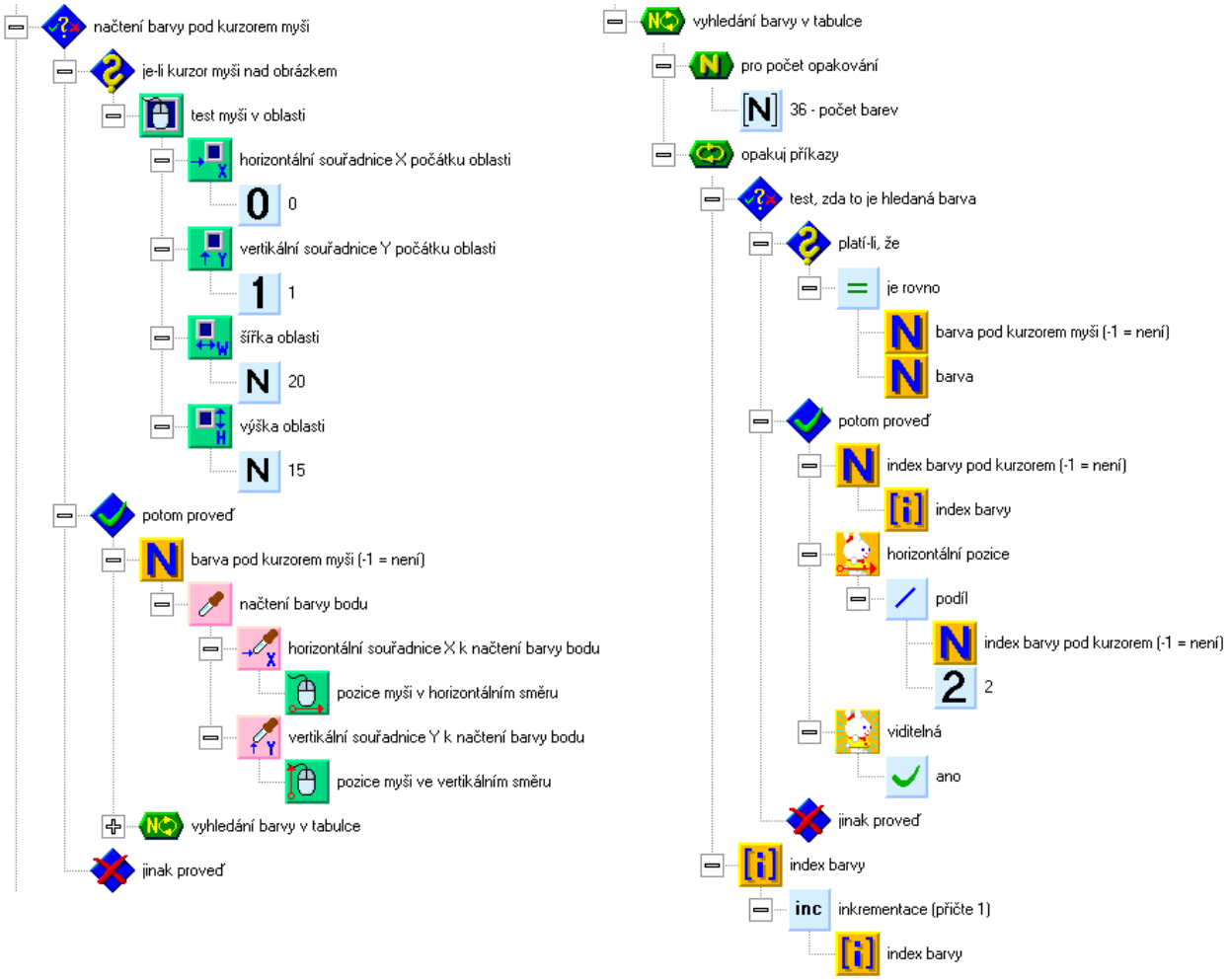
Lucku použijeme jako indikátor barvy pod kurzorem myši. Vyvolejte editaci sprajtu Lucky a nastavte parametry sprajtu: fází na krok = **0**, fází pro pohyb = **0**, směrů = **1**, šířka obrázku = **0.5**. Překreslete obrázek ve sprajtu tak, aby vytvořil dojem vyvýšeného okénka volby barvy. Střed přitom ponechejte průhledný.



V hlavní smyčce programu (před příkazem čekání) připravte skupinu s názvem **zjištění a test platnosti barvy pod kurzorem**. Zde budeme průběžně testovat barvu pod kurzorem myši, abychom mohli pro příslušnou barvu zapnout indikaci. Zjištěnou barvu budeme používat v programu i u dalších obsluh.



Ve **Společných proměnných** připravte dvě číselné proměnné: **barva pod kurzorem myši** a **index barvy pod kurzorem**. Na začátku skupiny tyto dvě proměnné nastavte na hodnotu **-1** k indikaci, že pod kurzorem myši není platná barva. Následuje podmíněný příkaz, testující zda je kurzor myši nad plochou obrázku. Test zajistíme prvkem **test myši v oblasti** (skupina **ovládání**, podskupina **myš**) s nastavenými parametry podle obrázku v okně. Je-li kurzor myši nad obrázkem, načteme barvu pod kurzorem funkcí **načtení barvy bodu** (skupina **grafika**). Barvu uchováme v proměnné **barva pod kurzorem myši**. Souřadnice bodu k načtení barvy převezmeme ze souřadnic myši **pozice myši v horizontálním směru** a **pozice myši ve vertikálním směru**. V programech systému Petr je zajištěno, že informace o myši (stejně i o jiných zařízeních) se až do další obsluhy čekání nezmění.

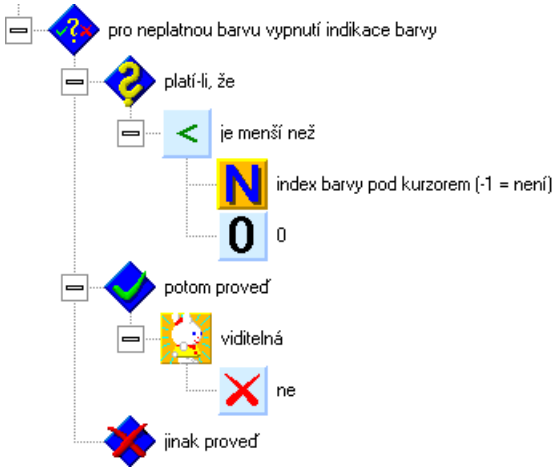


Po načtení bodu se pokusíme barvu vyhledat v tabulce barev. Cyklus vyhledání barvy vidíte na obrázku vpravo. Cyklus prochází celou tabulku barev, každou barvu porovnává se zjištěnou barvou pod kurzorem. Nalezne-li odpovídající barvu, uschová její index do proměnné **index barvy pod kurzorem**, nastaví Lucku (jako indikátor barvy pod kurzorem) nad okénko s barvou a zapne viditelnost Lucky.

Všimněte si blíže, jak jsme cyklus použili. Index barvy v seznamu barev určuje barvu vybranou uživatelem. Cyklus projede celý seznam barev jednou dokola, po ukončení cyklu bude mít index barvy přesně stejnou hodnotu, jakou měl před zahájením cyklu. Využili jsme přitom automatického návratu ukazatele seznamu na začátek po překročení konce seznamu.

Pozastavme se u Lucky. V celém programu ponecháváme vertikální souřadnici Lucky trvale na výchozí hodnotě 0. Horizontální souřadnici nastavujeme na poloviční hodnotu indexu barvy pod kurzorem, protože okénka barev jsou široká polovinu políčka. Pokud vás zaráží, že Lucka může stát i mezi políčky, tak je tomu skutečně tak. Při nastavování pozice Lucky zadáním souřadnic můžeme používat libovolné hodnoty, včetně hodnot mimo plochu okna, neboť Lucka je běžným sprajtem. Pouze při pohybování Lucky pomocí příkazu **krok** se Lucka (stejně i Petřík) omezuje na plochu okna a cílová souřadnice je zarovnána na nejbližší políčko.

Za podmíněným příkazem pro načtení barvy následuje podmíněný příkaz zajišťující vypnutí indikátoru barvy, není-li pod kurzorem myši platná barva. To může nastat nejen v případech, kdy je kurzor myši mimo plochu obrázku, ale také je-li pod kurzorem myši černá (nebo nějaká neznámá) barva.



Program spusťte a vyzkoušejte. Měla by již být funkční indikace barvy pod kurzorem myši. Najedete-li myší v obrázku nad nějakou barvu, okénko s příslušnou barvou vystoupí nad povrch okna.

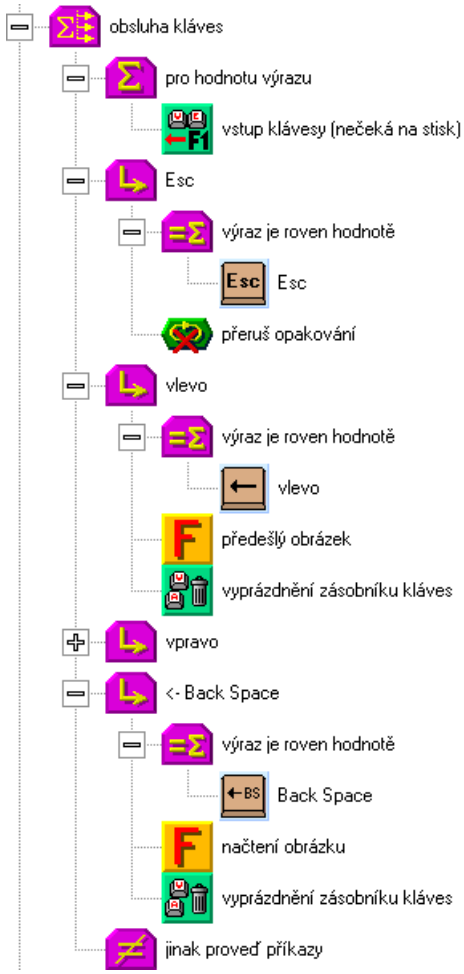
Začneme s ovládáním, a to nejdříve s ovládáním pomocí klávesnice. Doplňte za skupinu pro zjištění barvy prvek **vícestupňové větvení příkazů** se jménem **obsluha kláves**. V testované hodnotě použijeme funkci **vstup klávesy (nečeká na stisk)**.

První větev větvící konstrukce vytvoříme pro klávesu **Esc**. Jako obsluhu větve použijeme příkaz **přeruš opakování**, který přeruší hlavní smyčku programu.

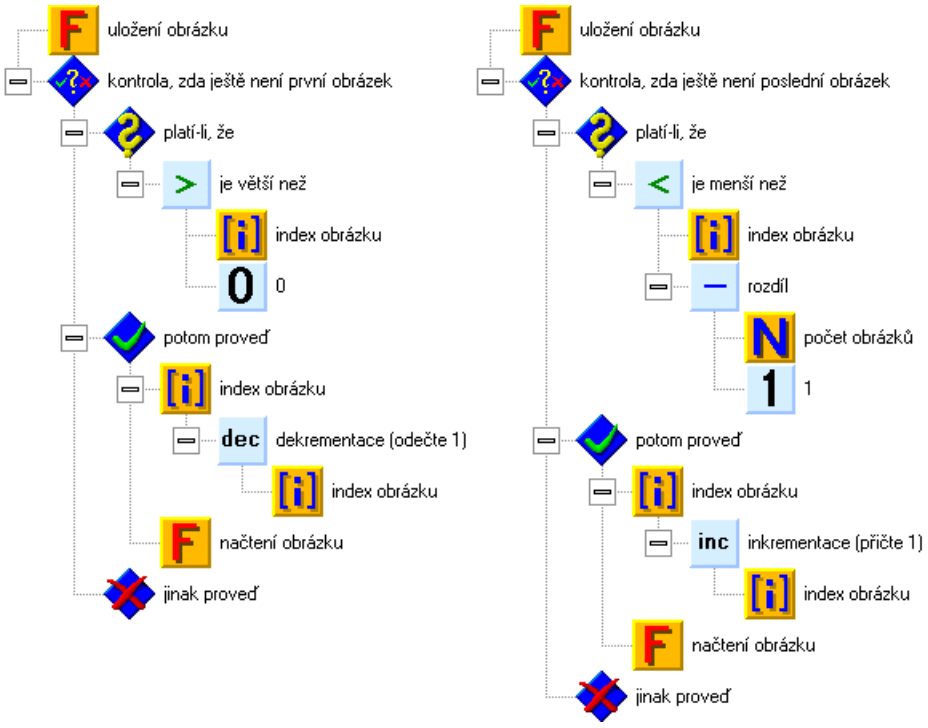
V druhé větvi obsloužíme klávesu **vlevo**. V obsluze větve uveďte funkci **předešlý obrázek**, předtím funkci vytvořte (zatím prázdnou). Za funkci doplňte příkaz **vyprázdnění zásobníku kláves**. Načítání obrázku chvíli trvá. Pokud uživatel klávesu drží, přichází kódy kláves rychleji, než je program schopen přepínat stránky. Program by tak pokračoval v listování obrázky i po uvolnění klávesy. Ovládání by mělo „setrvačnost“, což není příjemné.

Podobně obsloužíme klávesu **vpravo**, jen použijeme funkci **další obrázek**. Můžete ještě obsloužit další listovací klávesy, jako například **Home** pro skok na první obrázek, **End** pro skok na poslední obrázek, **Ctrl+vlevo**/**vpravo** na listování po 10 obrázcích.

Poslední klávesou bude **Back Space**. S její pomocí může uživatel obnovit obrázek do původní podoby jako před editací. Obsluhu zajistíme funkcí **načtení obrázku**, která znovu načte obrázek ze souboru. Následuje opět vyprázdnění zásobníku kláves.



Na obrázku níže vidíte obsluhu funkcí **předešlý obrázek** (po levé straně) a **další obrázek** (vpravo). Na začátku funkcí se uloží aktuální obrázek, pokud byl změněn. Zatím vytvořte jen prázdnou funkci **uložení obrázku**.

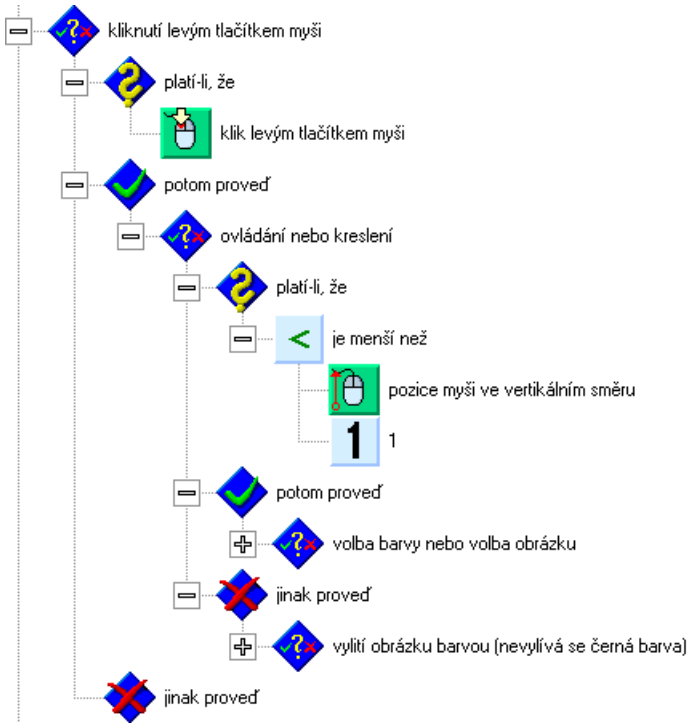


Ve funkci **předešlý obrázek** ověříme, zda nepracujeme s prvním obrázkem. Pokud ne, můžeme posunout ukazatel obrázků na předešlý obrázek a provést načtení a zobrazení nového obrázku. Pro první obrázek se neprovede nic, ale mohli bychom nastavit ukazatel na poslední obrázek (je to **počet obrázků - 1**) a přepínat obrázky dokola.

Podobně ve funkci **další obrázek** otestujeme, zda to není poslední obrázek. Číslo posledního obrázku získáme z počtu obrázků snížením o 1. Není-li to poslední obrázek, zvýšíme ukazatel na další obrázek a načteme nový obrázek. U posledního obrázku bychom mohli nastavit první obrázek a zajistit tak cyklické listování.

Program vyzkoušejte. Obrázky je teď možné listovat pomocí kláves **vlevo**/**vpravo** a program ukončit klávesou **Esc**.

Zatím nebudeme vytvářet funkci pro ukládání, abychom si nepovedeným pokusem nepřepsali nějaký obrázek. Vytvoříme nejdříve obsluhu ovládání myší. Do hlavní smyčky za obsluhu kláves přidejte podmíněný příkaz s názvem **kliknutí levým tlačítkem myši**.



Do testu podmínky vložte **klik levým tlačítkem myši** . Je to logický příznak a je nastaven vždy, když uživatel klikne levým tlačítkem myši do okna programu. Vypnutí příznaku se provede jeho čtením, například použitím v podmínce. Chcete-li příznak kliknutí testovat vícekrát, uschovejte si jej do logické proměnné .

Pokud v testu podmínky zjistíme, že uživatel kliknul myší, rozlišíme v dalším podmíněném příkazu podle vertikální souřadnice myši, zda se jednalo o kliknutí do obrázku nebo do pásu s barvami a tlačítky. Hranicí rozlišení je **1**, protože pás je vysoký **1** políčko.

Pro obsluhu volby barvy připravíme novou funkci s názvem **volba barvy**. Do vstupních proměnných funkce dáme číselnou proměnnou **pozice kurzoru myši X**. S její pomocí budeme funkci předávat horizontální souřadnici myši v poli volby barev.

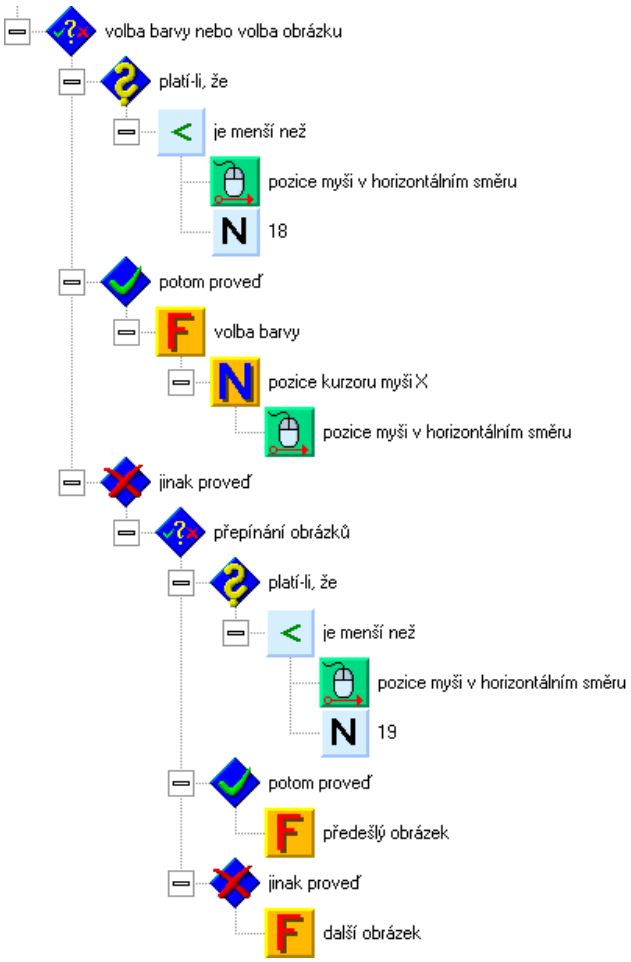
Ve funkci převezmeme hodnotu vstupní proměnné, vynásobením číslem **2** souřadnici převedeme na číslo barvy, funkcí **celočíselná část** odstraníme nepotřebnou desetinnou část a výsledek použijeme k nastavení nového indexu barvy. Nastavíme horizontální souřadnici Petříka na poloviční hodnotu indexu barvy, což odpovídá souřadnici okénka s barvou. Petříka v programu používáme jako indikátor vybrané barvy. Jeho vertikální souřadnice zůstává trvale nastavena na **0**, proto si jí nemusíme všímat.

Zjistíme-li z vertikální souřadnice myši, že uživatel kliknul do pásu tlačítek a volby barev, rozlišíme dále podle horizontální souřadnice myši, zda kliknul do volby barev nebo na tlačítka. Pro horizontální souřadnici menší než **18** je obsloužena volba barvy. Zavoláme funkci **volba barvy** s horizontální souřadnicí myši jako parametr. Jinak se jedná o tlačítka přepínání obrázků. Pro horizontální souřadnici menší než **19** je tlačítko vlevo, voláme funkci **předešlý obrázek**, jinak voláme funkci **další obrázek**. Celou obsluhu volby barvy nebo tlačítek vidíte na obrázku na další stránce.

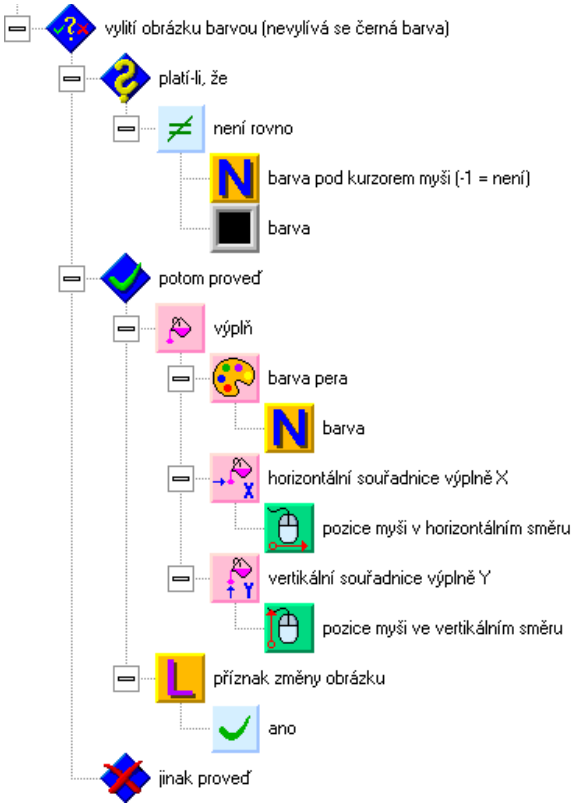
Program můžete vyzkoušet. Levým tlačítkem myši je možné přepínat obrázky a v pásu volby barev měnit vybranou barvu.

Nyní obsloužíme případ, kdy uživatel kliknul levým tlačítkem myši do oblasti obrázku. Základem obsluhy bude podmíněný příkaz **vylití obrázku barvou** (na další stránce dole). V testu podmínky ověříme, zda pod kurzorem myši není černá barva. Černou barvu používáme pro obrysy obrázků a nesmíme ji vylít. Barvu pod kurzorem myši jsme si uschovali do pomocné proměnné na začátku hlavní smyčky programu.

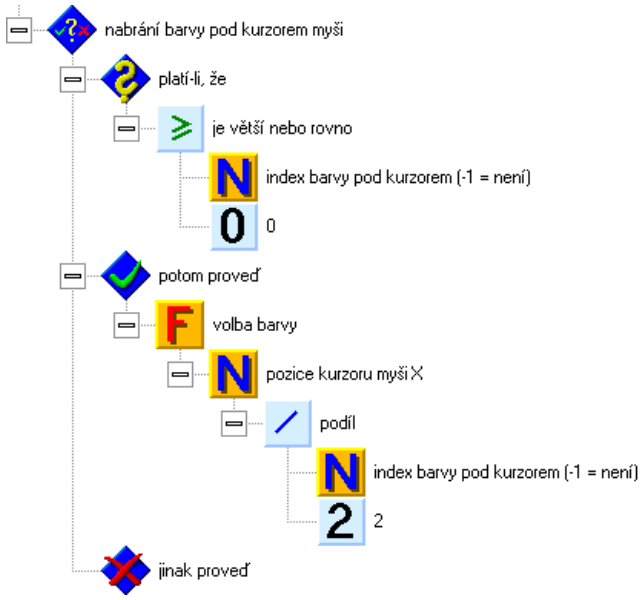
V obsluze vylití obrázku barvou použijeme příkaz **Výplň**  . Barvu určí proměnná **barva** ze seznamu barev. Souřadnice výplně převezmeme ze souřadnic myši. Nakonec nastavíme **příznak změny obrázku**, který připravíme v okně **Společné proměnné a funkce**. Příznak indikuje, že je potřeba obrázek uložit.



Program spusťte. Vyzkoušejte vylévání obrázku vybranou barvou a ověřte, že nelze vylít černý obrys obrázku.



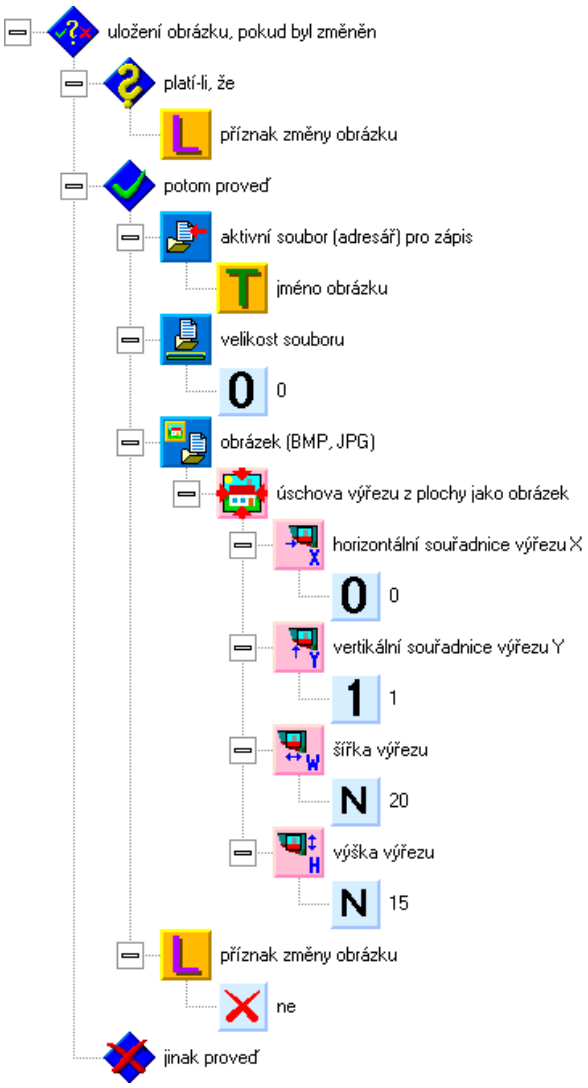
Obsluha pravého tlačítka myši bude vypadat velmi podobně, proto můžeme zkopírovat celou obsluhu levého tlačítka myši. V testu podmínky nahradíme prvek testu kliknutí levým tlačítkem prvkem pro pravé tlačítko. Obsluha výběru funkce nebo barvy zůstává nezměněna. Obsluhu vylití obrázku barvou vyhodíme a na její místo dáme obsluhu **nabrání barvy pod kurzorem myši**. V obsluze otestujeme proměnnou **index barvy pod kurzorem**, zda je pod kurzorem myši platná barva (zda to není černá nebo nestandardní barva). Ukazuje-li kurzor myši na platnou barvu, nastavíme ji jako novou vybranou barvu (přepočteme barvu na souřadnici X).



Zkouškou programu můžete ověřit ovládání volby barvy a obrázku pravým tlačítkem myši stejně jako u levého tlačítka. Kliknutím pravým tlačítkem myši do plochy obrázku lze „nabrat“ barvu pod kurzorem myši a nastavit ji tak jako vybranou barvu.

Editace je hotova a konečně se dostáváme k funkci **uložení obrázku**. Obsah funkce vidíte na obrázku na další stránce. Nejdříve otestujeme, zda byl obrázek změněn a jestli je tedy nutné jeho uložení. Pokud ano, použijeme jméno obrázku (ze seznamu jmen obrázků) jako **aktivní soubor pro zápis**. Dále nastavíme **velikost souboru** na **0**, aby v souboru za obrázkem nezůstala případná stará data. Obrázek uložíme tak, že funkcí **úschova výřezu z plochy jako obrázek** sejmeme obrázek z okna a předáním do prvku **obrázek** provedeme zápis obrázku do souboru. Závěrem vypneme **příznak změny obrázku**.

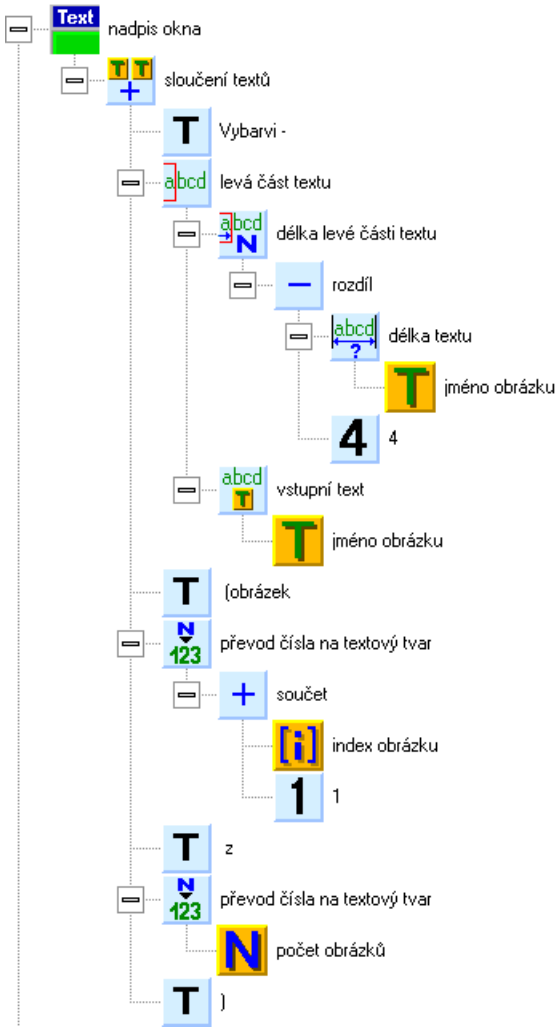
Funkci **uložení obrázku** doplňte za hlavní smyčku programu, aby se obrázek uložil při ukončení klávesou **Esc**. Nyní je program po stránce funkčnosti kompletně hotov a můžete ho vyzkoušet. Ověřte, zda se změny v obrázcích ukládají při přepínání obrázků i při ukončení programu klávesou **Esc**.



Náš program můžeme ještě dále vylepšovat. Například zajistíme, aby se v nadpisu okna objevovalo jméno obrázku, číslo obrázku a celkový počet obrázků. Obsluhu doplníme na začátek funkce **načtení obrázku** podle obrázku na další stránce. Výsledkem bude text jako například „**Vybarvi - Domeček (obrázek 4 z 50)**“. Na obrázku to není patrné, ale nezapomeňte na patřičných místech v textech mezery (na konci textu „Vybarvi - “, na obou stranách „ (obrázek “ a také „ z “). Jméno souboru zobrazíme bez přípony, proto z konce textu odstraňujeme 4 znaky (tečku a text BMP).

Dalším vylepšením bude předefinování kurzoru myši. V okně **Společné proměnné a funkce** si připravte 4 předměty s obrázky kurzorů - šipky, kapátko a výplň.

Obrázky mají černý obrys, okolí je průhledné a vnitřek vyplněn bílou barvou. Na vrcholy kurzorů kapátka a výplně doplníme žlutou tečku coby ukazovací místo kurzoru (s inverzí barev). U ostatních kurzorů zůstane ukazovací místo implicitně uprostřed obrázku.



Ve skupině **inicializace programu** určíme pomocí příkazů **nadefinování vzhledu kurzoru myši**  , jaký vzhled ve které části okna bude kurzor myši mít. Pro pole výběru barvy nadefinujeme kapátko, pro ovládací tlačítka šipky vlevo a vpravo, pro plochu obrázku kurzor výplně.

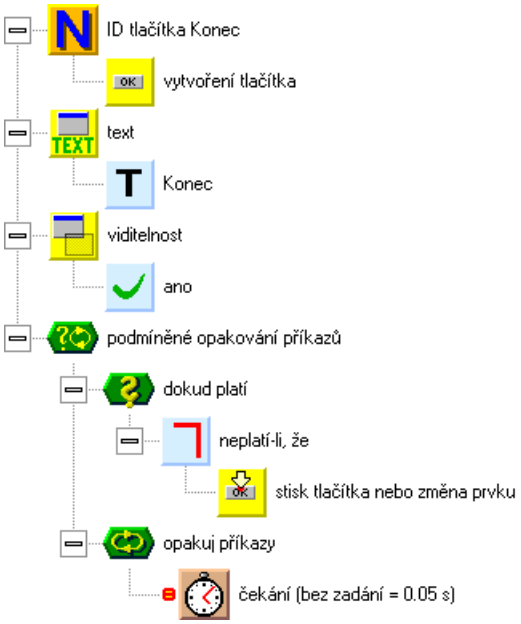
Abychom si ujasnili funkci příkazu pro definování vzhledu kurzoru myši, můžeme si představit oblasti definice vzhledu myši jako obdélníky, které postupným přidáváním překrývají staré definice. Platí vždy definice nejvíce na vrcholu. Nezadáme-li žádný obrázek myši, použije se standardní vzhled myši v okně. Máme-li ve Windows nadefinován barevný kurzor, použije se tento barevný kurzor. Nezadáme-li rozměry oblasti, předefinuje se kurzor myši pro celé okno. Všechny definice můžeme zrušit uvedením příkazu bez jakýchkoliv parametrů.

16 Vedeme dialogy

Po úspěšných pokusech v oblasti grafiky se podíváme na novou oblast - dialogy. Nejedná se ovšem o tlachání s přáteli, jak by se mohlo zdát. Dialogem nazýváme okno sloužící ke komunikaci s uživatelem. Jeho prostřednictvím rozhoduje uživatel o dalším chodu programu, předává programu informace a také informace z programu získává.

Vytvořte nový program s názvem **Dialogy**. Prvky pro práci s dialogy najdete ve skupině **ovládání**, podskupině **dialogy**. Začneme s vytvořením jednoduchého tlačítka.

V okně **Společné proměnné a funkce** vytvořte číselnou proměnnou s názvem **ID tlačítka Konec** a v hlavní funkci sestavte program podle následujícího obrázku.



Když takový program spustíte, uvidíte šedou plochu a uprostřed ní tlačítko s textem **Konec**. Kliknutím na tlačítko se program ukončí. Co se v programu děje?

Začneme od číselné proměnné. Označení **ID tlačítka Konec** znamená „*Identifikační kód tlačítka Konec*“. Každý prvek v okně má své identifikační číslo, kterým se můžeme na prvek odkazovat. Identifikační číslo je každému prvku přiděleno při vytvoření prvku automaticky. Hodnota identifikačního čísla nás nezajímá, pro nás to je náhodné číslo. Číslo si pouze uschováme a při odkazu na prvek použijeme.

Prvním příkazem v programu je vytvoření běžného tlačítka. Funkce pro vytvoření okenního prvku navrací identifikační kód, který si uschováme pro pozdější použití.

V okamžiku vytvoření prvního okenního prvku se program přepne do dialogového módu. Zmizí grafická plocha programu a objeví se jednobarevná plocha okna (obvykle světle šedá). Od této chvíle probíhají vstupy od uživatele přes okenní ovládací prvky. Grafický mód bude opět navrácen zpět až po zrušení posledního okenního prvku.

Při práci s dialogy se příkazy a funkce vztahují k dialogovému prvku, který je vybraný. Vybraný prvek určujeme prvkem **číslo prvku**  , kterému předáme identifikační číslo prvku. Nezaměňujte vybraný prvek s prvkem se vstupním zaměřením od uživatele, jako je třeba aktivní textové pole, do kterého bude probíhat zadávání textu uživatelem. Vybraný prvek je pouze vnitřním ukazatelem programu, se kterým prvkem bude nyní manipulováno. Při vytvoření nového prvku je nový prvek automaticky nastaven jako vybraný, takže v našem prográmku nemusíme zatím vybraný prvek nastavovat.

Druhým příkazem v programu je nastavení textu **Konec** do tlačítka. K nastavení textu slouží prvek **text**  . U tlačítek se text zobrazí uprostřed tlačítka jako popisný text tlačítka. Jiné prvky mohou mít také text - název přepínače, nadpis rámečku, text editačního řádku či editačního pole, vybraný řádek seznamu, nadpis okna.

Třetím příkazem **viditelnost**  prvek zviditelníme. Všechny prvky (i okna) jsou při svém vytvoření neviditelné. Je to z důvodu, abychom mohli nastavovat potřebné vlastnosti prvků a prvek nám přitom „necestoval“ po okně, což by jistě nebylo hezké. Proto nezapomínejte po nastavení vlastností prvků zapnout viditelnost prvku.

Jak jsme si řekli, ovládání programu je nyní předáno okenním prvkům. Hlavní smyčka programu bude proto vypadat trochu jinak. Použijeme opět podmíněný cyklus, podmínkou bude **neplatí-li, že**  **je stisk tlačítka nebo změna prvku**  . Prvek **stisk tlačítka nebo změna prvku** indikuje, že s prvkem byla provedena určitá akce. Jaká akce, to závisí na prvku. U tlačítek to znamená stisknutí, u přepínače přepnutí, u editačního řádku změnu textu v řádku. Příznak je otestováním vypnut. Okno má také svůj příznak změny, oznamující nastavení příznaku změny některého z prvků okna.

Nelíbí-li se vám, že tlačítko pro ukončení programu je uprostřed okna, doplňte za vytvoření tlačítka, ale ještě před jeho zviditelnění, prvek **vertikální souřadnice**  s číselným parametrem **1**. Tím tlačítko posunete k dolnímu okraji okna.

Při používání dialogových prvků se setkáte s omezením počtu barev. V režimu obrazovky 256 barev je počet barev u dialogových prvků omezen na 20 základních barev Windows. Spuštěné programy se v tomto videorežimu musí spolu dělit o možných 256 barev. Dialogová okna běžně nevyžadují vyšší barevnost, proto jsou jejich barvy omezeny na základní barvy Windows. Přesto je možné i v dialogovém okně použít více barev, a to pomocí prvku **vytvoření obrázku**  . Prvek obrázku používá plný rozsah barev, stejně jako v grafickém režimu programu.

Nejvíce zkušeností při programování získáte vlastním experimentováním. Nebude-li vám funkce některého prvku jasná, označte ho kliknutím myší a stiskněte klávesu **F1**. Zobrazí se podrobná nápověda k prvku. Dalším dobrým zdrojem informací jsou ukázkové programy. Po otevření ukázkového programu označte požadovaný prvek v okně **Základní prvky, koš**. V pravém dolním rohu okna Petra se zobrazí počet, kolikrát je prvek v programu použit. Stiskem tlačítek **Předešlé použití**  a **Další použití**  se můžete přemisťovat po výskytech prvku v programu.