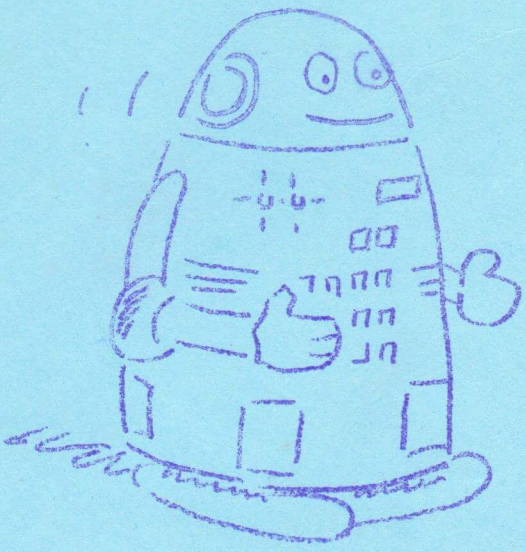
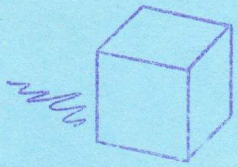
Okresní dům pionýrů a mládeže Jindřichův Hradec





KOSTKA

pro ONDRU

Příručka pro výuku KARLA na mikropočítači ONDRA

Příloha: kazeta mg. s programy "Karel" a "Slovník"

1645/87

KOSTKA PRO ONDRU je stručná příručka k mikropočítači ONDRA vybavenému operačním systémem z Centra pro mládež a techniku ÚV SSM. Obsahuje především povídání o programu Karel a o výuce programování pomocí tohoto programu. Je určena vedoucím kroužků výpočetní techniky a domům pionýrů a mládeže na okrese Jindřichův Hradec. Tato příručka není metodikou výuky, ale podle našeho názoru obsahuje všechny základní informace o KARLOVI na počítači ONDRA. V této kostce je vše v kostce.

O b s a h :

Jindřichův Hradec 26. září 1987

KAREL je výukový program pro výuku základů programování, to jest především algoritmizace. Je prvním stupněm při výuce programování a je vhodný jak pro děti, tak pro dospělé. Hravou formou /učení robota Karla, pohybujícího se ve městě/ se lze velmi rychle naučit základům algoritmizace a strukturovaného programování. Pro výuku programování je vhodné použít některou metodiku pro výuku Karla:

časopisy	Televize	roč.	čís.
	VTM	roč, 1975	čís. 1 - 11
	Sedmička	roč.	čís.
	Zenit pion.	roč. 1987/8	čís. 1 - ...

Dálkový kurs 602. ZO Svazarmu Praha 1.a2.roč.
nebo přímo metodiku KARLA, autor Michal Vejvoda, ZO Svazarmu Č.Krumlov, která je nejvhodnější, i když je psána pro počítač PMD-85. Některé změny v obsluze obou počítačů jsou vysvětleny dále /jde o nahrávání Karla, práci s klávesnicí apod./

1. Nahrávání a spouštění KARLA na ONDROVI

Po zapnutí počítače se na obrazovce objeví nápis "Zdraví Vás ONDRA" a pod ním blikající ukazovatel /kurzor/. Po napsání "Karel" a stisknutí klávesy NOVÝ ŘÁDEK [↵] začne obrazovka blikat a počítač očekává nahrávání z mg.pásky. Jakmile začne načítat program, obrazovka zhasíná. Není to tedy porucha, ale známka, že vše probíhá v pořádku. Program se nahrává v blocích, očíslovaných kódy 1 až 6. V přestávkách mezi čtením bloků získáme z obrazovky informaci, jaký kód z pásky počítač očekává.

Je-li nahrán celý program, sám se spustí. Je tedy pouze nutno vypnout magnetofon.

Pokud počítač nenahrává, obrazovka bliká, nalevo je zadán název a kód bloku, který se očekává, napravo název a kód bloku, který je na pásce.

Dojde-li k chybě čtení, zobrazí se nápis "Chyba čtení" a kód bloku, který byl špatně čten. Je pouze nutno posunout mg. pásku zpět, aby tento blok mohl být čten znovu.

1647/87

Chceme-li přerušit nahrávání, tiskneme klávesu NMI /je to kruhová klávesa zapuštěná do boku počítače a je nutno ji tisknout tužkou/. Pak je ovšem nutno nahrávat znovu.

Tohoto postupu lze použít v případě, že počítač začne nahrávat jiný program /obrazovka zhasíná, objevuje se na ní neočekávaný název programu/. Stane se to zvláště tehdy, jsme-li líní psát název programu a tiskneme pouze klávesu NOVÝ ŘÁDEK.

2. KAREL je nahrán

Hurá. Podíváme se teď, co se děje na obrazovce.

Obrazovka je rozdělena na 4 "okna". V pravém horním okně je robot KAREL ve svém městě. Město je tvořeno čtvercem 10x10 políček /pokud vám Karlovo město připomíná spíše obdélník, nechte si seřídit televizor/. Robot KAREL má na těle šipku, která ukazuje směr, do kterého je Karel natočen. Čtyři směry na obrazovce jsou nazvány SEVER, JIH, VÝCHOD, ZÁPAD jako na mapách. Na začátku je tedy Karel natočen na východ.

Má-li KAREL udělat krok, posune se o jedno políčko ve směru šipky. Karel nemůže procházet zdí; dostane-li příkaz takový, že má projít zdí, zastaví se u ní a ohlásí chybu: narazil jsem.

Karel umí i jiné věci, než jen krok. Umí udělat vlevo vbok, tedy otočit šipku o 90° proti směru pohybu hodinových ručiček /např. z východu na sever, z jihu na východ atd./. KAREL také umí položit značku na místě, kde stojí, může také značku zvednout, je-li tam nějaká. Může však položit na jedno políčko nanejvýš 5 značek.

Jen dřep robot Karel neumí. Ten si ale můžeme udělat za něj my; zvláště je vhodný tehdy, začínají-li se nám za klávesnicí třást ruce nebo vypovídat paměť.

V pravém dolním okně roluje SLOVNÍK Karla, tedy příkazy, které Karel již zná a které umí vykonávat. Ostatní příkazy, které ve Slovníku nejsou, /např. DVOJKROK, KE-ZDI, DOMEK/ jej musíme naučit, vysvětlit jej pomocí známých příkazů, zapsat algoritmus činnosti. To je smyslem výuky na programu Karel.

V levém okně obrazovky rolují příkazy, které zadáváme KARLOVI. Příkazy píšeme do čtvrtého okna, na tzv. dialogový řádek. Je to poslední řádek na obrazovce a nad ním je napsána informace počítače, podle níž poznáme, v jakém režimu program

pracuje /např. nápis "Napiš příkaz"/. Objeví-li se na tomto místě nápis "Jdi domů", okamžitě vypneme počítač a jdeme domů, neboť nás tam čekají. Počítač je velmi chytrý.

3. Zadávání příkazů

Příkazy zadáváme KARLOVI pomocí klávesnice. Je podobná klávesnici psacího stroje, pouze písmena Y a Z jsou vyměněna. Znak klávesy, kterou tiskneme, se okamžitě objeví v dialogovém řádku. Je-li příkaz ukončen, tiskneme tlačítko NOVÝ ŘÁDEK [↵] /new line, jinde též označovaným CR, tedy car revers, návrat vozu/. Příkaz se tímto tlačítkem "odešle" do počítače ke zpracování. Pokud KAREL příkaz zná, začne jej ihned provádět.

V příkazu se nesmí objevit mezera. Proto např. příkaz vlevo vbok nemůže být napsán VLEVO VBOK, ale VLEVO-VBOK.

KAREL používá velká i malá písmena české abecedy. Po tisknutí klávesy se píše velké písmeno; chceme-li, aby se napsalo písmeno malé, musíme současně s klávesou tisknout přeřadovač velkých písmen /shift/, který je vlevo dole a vypadá [⇧]. Je to podobné jako psaní velkých písmen na psacím stroji.

Znak zobrazený na klávese nad písmenem se vypíše tak, že tiskneme současně přeřadovač horních znaků [◇] . Číslice zobrazíme tak, že současně tiskneme přeřadovač číslic [0..9].

Poznámka:

Při psaní malých písmen musíme stále držet klávesu SHIFT. Pomocí současného stlačení kláves [CTRL] a [⇧] lze dosáhnout toho, že se malá písmena budou vypisovat přímo a velká písmena se SHIFTEM /velká a malá písmena se "vymění"/. Původního stavu pak lze dosáhnout opět těmito klávesami.

Česká abeceda používá běžně písmena s diakritickými znaménky /č,ý,ů,Š/, která na klávesnici nejsou. Zobrazíme je tak, že předtím, než tiskneme příslušnou klávesu, stlačíme přepínač písmen české abecedy [ČS] . Tento přepínač platí pouze pro jeden znak /např. chci-li psát ČÍMĚŘ, musím tisknout po sobě:

[ČS] [C] [ČS] [I] [M] [ČS] [E] [ČS] [R] /

POZOR!

Některá písmena mají dvě používaná diakritická znaménka např. E,É,Ě. Jen jedno z nich najdu pod odpovídající klávesou,

ostatní musím "objevit" pod jiným písmenem. Např.:

Ě	-	ČS	W
ů	-	ČS	j
Ľ	-	ČS	K

apod.

Dialogový řádek se dá při chybě opravovat. Záleží přitom na poloze ukazovatele /kurzoru/ - blikající obdélník. Bližší informace v 4.kapitole.

Poznámka: Příkazy lze do dialogového řádku zapisovat zkratkou. Zkratka je začátek slova zakončený tečkou. Pak KAREL hledá ve Slovníku první slovo,x které začíná stejně jako zkratka. Příklad - chceme-li napsat zkratkou VLEVO-VBOK, stačí napsat "V." , chceme-li napsat zkratku slova MGČTENÍ, musíme psát "MG." - nestačí napsat "M." , protože ve slovníku je před tímto příkazem již příkaz MĚSTO.

4. Editace v dialogovém řádku

Někdy se stane, že třaslavá ruka uživatelova stiskne omylem jiné tlačítko a na dialogovém řádku se objeví příkaz-chudinka, příkaz pohmožděný, který ani Karel nezná a nechce tudíž vykonat. Takovýto příkaz je nutno opravit. K editaci užíváme klávesy šipek, které posunují kurzorem po dialogovém řádku.

ŠIPKA VLEVO	-	kurzor o 1 místo vlevo
Šipka VPRAVO	-	kurzor o 1 místo vpravo
CTRL + ŠIPKA VLEVO	-	vymazání písm.před kurzorem
CTRL + ŠIPKA NAHORU	-	vymazání písm. pod kurzorem
CTRL + ŠIPKA VPRAVO	-	vymazání konce řádku za kurz.
ŠIPKA DOLŮ	-	vymazání celého řádku
NOVÝ ŘÁDEK	-	potvrzení řádku, odeslání do počítače

5. Užití tlačítka NMI - nemaskované přerušení

Během programu se může vyskytnout chyba /nemusí být naší vinou/ a počítač přestane spolupracovat, "dělá si co chce". K poslušnosti ho přinějeme tlačítkem NMI.

16/0/87

Tlačítko NMI je červené, kulaté a je zapuštěno vlevo v boku počítače a stlačit jej můžeme tužkou. Tisknete-li NMI, dojde k přerušení práce KARLA a k novému nastartování programu KAREL.

Použití:

NMI + ŠIPKA VLEVO — nebo
NMI + ŠIPKA VPRAVO — nastartuje se KAREL, ale smaže se slovník /studený start programu/

NMI — nastartuje se KAREL a slovník zůstane zachován /teplý start programu/

POZOR!

NMI + ŠIPKA DOLŮ — přeruší se program KAREL, řízení počítače se předá MONITORU. Návrat zpět přes studený start programu.

Nereaguje-li tlačítko NMI ani NMI + ŠIPKA DOLŮ, je nutno vypnout počítač a počkat asi 20 sekund, teprve potom opět zapnout zdroj a začít znovu nahrávat KARLA.

V další kapitole se věnujeme práci programu KAREL.

6. Režimy práce KARLA

Program KAREL pracuje ve třech režimech:

A. ZÁKLADNÍ REŽIM - provádění známého příkazu,
B. REŽIM DEFINOVÁNÍ NOVÉHO PŘÍKAZU - zapsání do slovníku a opravy příkazů ve slovníku,
C. REŽIM MĚSTO - nastavení počátečních podmínek v Karlově městě.

A. REŽIM PROVÁDĚNÍ ZNÁMÉHO PŘÍKAZU

Poznáme jej podle toho, že nad dialogovým řádkem je napsáno "Napiš příkaz".

Pokud napíšeme příkaz, který KAREL "zná", tzn. tento příkaz je ve slovníku, okamžitě po odeslání jej Karel začne provádět. Provádění příkazu lze zrychlit tlačítkem R, zpomalit /např. pro názorné vysvětlování funkce a průběhu příkazu/ lze klávesou P, zastavit tlačítkem S.

Pokud napíšeme příkaz, který KAREL nezná /není ve Slovníku/, automaticky program přejde do režimu B - definování nového příkazu, tzn. KAREL se tento nový, jemu neznámý příkaz, chce "naučit". Je tedy velmi důležité přesně příkaz zapsat tak, jak je ve Slovníku - i s čárkami, pomlčkami apod. I chyby /např. POLOZ místo POLOŽ, VLEVO_VBOK místo VLEVO-VBOK/ chápe KAREL jako nový příkaz.

Podaří-li se nám "odeslat" do počítače takovouto chybu, je možno ji opravit příkazem CHYBA /nebo zkratkou "C."/. Chybný příkaz se zruší.

Primitiva. Jsou to slova, která KAREL zná "od narození", tedy od spuštění programu Karel. Ta jsou vypsána ve Slovníku v pravém dolním okně a můžeme si je kdykoliv prohlédnout příkazem SLOVNÍK.

Seznam Primitiv Karla:

KROK	KAREL se posune o 1 políčko ve směru šipky
VLEVO-VBOK	KAREL se otočí o 90° vlevo /v kladném smyslu/
POLOŽ	KAREL položí značku na místě, kde stojí
ZVEDNI	KAREL zvedne značku
MĚSTO	přechod do režimu C - režimu Město
SLOVNÍK	prohlížení Slovníku - primitiv i nově naučených slov
MGZÁPIS	zápis Slovníku na mg. pásku
MGČTENÍ	čtení nahraného Slovníku z magnetofonu
CHYBA	opravení námi způsobené chyby při definování příkazu. Příkaz má dvojí použití - v režimu A vymaže poslední slovo ve Slovníku, v režimu B vymaže řádek vypisovaného příkazu, a to řádek poslední.
ROZKLAD	výpis určeného příkazu s možností jeho opravy. Tímto příkazem můžeme opravit libovolný příkaz ve Slovníku /pochopitelně kromě primitiv/. Rozkladem program přechází do režimu oprav příkazů. Nechceme-li příkaz opravovat, tiskneme NOVÝ ŘÁDEK.
KONEC	stačí tisknout klávesu NOVÝ ŘÁDEK. Příkaz KONEC ukončuje deklaraci /zavedení, definování/ nového příkazu, struktury, procedury, cyklu, rozhodovacího bloku.
OPAKUJ	příkaz cyklu /příkazy uvedené za ním až do příkazu KONEC se opakují několikrát - lze volit počet opakování/
KDYŽ	příkaz rozhodování
DOKUD	příkaz cyklu /příkazy se opakují do splnění podmínky/

1982/83

Nové příkazy zapisujeme pomocí příkazů již KARLOVI známých a přehledně je zapisujeme do KOPENOGRAMŮ - viz kap. 7.

Nové příkazy se pochopitelně s vypnutím počítače zruší. Je možno však všechny nové příkazy uchovat na mg.pásku a později je opět nahrát. K tomu slouží příkazy MGZÁPIS a MGČTENÍ. Je vhodné je nahrát pod jménem SLOVNÍK /kvůli přehlednosti/.

B. Režim definování nového příkazu

Tento režim práce programu Karel slouží k deklaraci nového příkazu nebo opravě některého příkazu ze Slovníku.

Nastavení tohoto režimu signalizuje nápis nad dialogovým řádkem: "Napiš ZNÁMÝ příkaz".

Do tohoto režimu přechází program automaticky po napsání slova, které není ve Slovníku - chápe jej jako příkaz, který "nezná a chce se jej naučit". Název nového příkazu se ihned vypisuje na konec Slovníku.

Příkaz zapisujeme pomocí KARLOVI známých příkazů; příkaz "odeslaný" do počítače se ihned zobrazuje v levém okně na konci nově vypisovaného příkazu. Deklaraci nového příkazu ukončujeme klávesou NOVÝ ŘÁDEK, která znamená příkaz KONEC. Po něm přejde program do základního režimu A.

Uděláme-li chybu, můžeme příkazem CHYBA vymazat poslední řádek deklarovaného příkazu. Chceme-li smazat řádek uprostřed příkazu, musíme smazat všechny řádky za ním, jedině tak se dostaneme k tomuto řádku.

Někdy by bylo takto mazat řádky velice zdlouhavé, je proto lepší příkazy KONEC ukončit tento špatný příkaz a v základním režimu jej příkazem CHYBA smazat celý najednou a začít jej psát znovu.

Pomocí tohoto režimu lze též smazat příkaz uprostřed Slovníku, což základní režim neumožňuje. Stačí pouze po ROZKLADU postupně vymazat všechny řádky tohoto příkazu a tím se celý příkaz zruší. Má to význam zvláště při tvorbě "reprezentačního Slovníku", v němž nechceme mít špatný příkaz.

STRUKTURY. Při psaní /deklaraci/ nového příkazu používáme tzv. struktury, které zlepšují přehlednost zápisu a zkvalitňují práci programátora. Mezi struktury počítáme podprogramy /procedury/, cykly, rozhodovací bloky. Jejich použití je krátce vysvětleno v 7. kapitole, blíže vysvětleno v metodikách. Každou strukturu musíme ukončit příkazem KONEC. Proto se někdy stane, že příkaz končí několika KONCI - tedy vlastně příkaz končí až tím posledním koncem.

REKURZE. Protože okamžitě po zadání názvu nového příkazu se tento zapisuje do Slovníku, lze jej v deklaraci také používat. můžeme tedy v zápisu nového příkazu zapsat jeho název, jako by ho KAREL již znal. Provádění a definování příkazu pomocí sebe sama se nazývá rekurze a její zvládnutí je pro začínajícího programátora poměrně obtížné. Rekurze patří k vrcholu algoritmizace a teprve po jejím perfektním zvládnutí je možno přejít k programování s datovými soubory, /tzn. v našem případě k výuce PASCALU/. Používání rekurze vyžaduje jistou opatrnost. Bližší vysvětlení je v kap. 7 a ve Slovníku v příkazech PIRUETA, POLOVINA atd.

Příkaz obsahující rekurzi nelze vymazat ze Slovníku. Chceme-li jej přesto vymazat a zrušit, musíme nejprve příkaz "opravit"tak, že vymažeme řádek obsahující rekurzi. Potom lze příkaz zrušit.

Poznámka: pro příkaz nově deklarovaný můžeme nazvat jakkoliv /např. když příkaz KAKTUS bude znamenat KROK, pak KAREL na příkaz KAKTUS udělá krok/, ale pokud možno zadáváme názvy mnemotechnicky - kvůli čitelnosti.

C. Režim Město

Do tchoto režimu se přechází ze základního režimu příkazem MĚSTO. Místo robota KARLA se v jeho městě objevuje malý kurzor, s nímž lze pohybovat pomocí tlačítek ŠIPEK a lze na jeho místě pokládat značky, stavět zeď uprostřed města a zase je vybírat. Návod na použití příkazů režimu Město svítí v levém okně obrazovky.

Tento režim slouží k nastavení počátečních podmínek pro vykonání příkazu robotem. Např. chceme-li vyzkoušet příkaz na vyčištění města od značek, můžeme je tímto režimem do města položit, kam budeme chtít. Můžeme také např. postavit bludiště, postavit dům doprostřed města, vytvořit menší město atd. Příkazem NOVÉ MĚSTO - klávesou N lze nastavit počáteční stav ve městě, tj. KAREL v levém dolním rohu otočen na Východ. Nastavením počátečních podmínek testujeme, zda nový příkaz je správný, jestli pracuje za jakýchkoliv podmínek.

Ukončení práce v režimu Město se provede příkazem ROBOT - tj. klávesou 'R'. Robot KAREL se objeví na místě kurzoru.

7. Strukturované programování, kopenogramy

Moderní programování se řídí zápisem programů /příkazů/ pomocí struktur. Mezi struktury, které vyučujeme v programu Karel, řadíme:

Procedura /podprogram/
Cyklus /opakování/
Rekurze /deklarace příkazu pomocí sebe sama/
Rozhodovací blok /větvení programu/

KOPENOGRAMY /též strukturogramy/ slouží ke grafickému zápisu algoritmů - obdoba vývojových diagramů, ovšem podstatně vhodnější, což vynikne zvláště u složitějších programů. Zapisují se do obdélníků a používají barvy kvůli přehlednosti.

Příkl.

v počítači:

Dvojkrok znamená
KROK
KROK
KONEC

hlava programu

tělo programu

Hlava programu obsahuje jeho název a je vybarvena žlutě, tělo programu obsahuje struktury a výkonné příkazy a je bílé.

Výkonné příkazy , např. KROK, VLEVO-VBOK, POLOŽ, ZVEDNI a všechny nové příkazy ve slovníku. V kopenogramu jsou vybarvovány červeně.

VPRAVO-VBOK_2 znamená
ČELEM-VZAD
VLEVO-VBOK
KONEC

Ky Procedury /podprogramy/ výrazně zpřehledňují zápis programu. Podprogram je již dříve deklarovaný příkaz zapsaný ve slovníku. Např. v příkazu VPRAVO-VBOK_2 je procedurou příkaz ČELEM-VZAD, který musí být předem deklarován. Použití podprogramů v složitějších příkazech je ukázáno v příkazech JEŘÁB, DŮM, STAVBA.

Při tvorbě programu si úkol dělíme na jednodušší úkoly /např. dům dělíme na přízemí, patro a překlad/, potom tyto dílčí příkazy vyřešíme. Při zapisování do počítače musíme však postupovat opačným způsobem: nejprve deklarovat podprogramy, potom hlavní program pomocí podprogramů.

16/7/87

Dáváme k uvážení čtenáři, zda je přehlednější zápis příkazu s procedurami nebo bez nich:

DŮM znamená
PŘÍZEMÍ
PŘEKLAD
PATRO
PŘEKLAD
DO-DVEŘÍ

V tomto sloupci je zapsán celý program Dům pomocí podprogramů. Spolu s deklaracemi podprogramů zabere asi 70 řádků.

DŮM znamená
VLEVO-VBOK
POLOŽ
POLOŽ
POLOŽ
POLOŽ
POLOŽ
KROK
POLOŽ
POLOŽ
POLOŽ
POLOŽ
POLOŽ
ČELEM-VZAD
KROK
VLEVO-VBOK
KROK
VLEVO-VBOK
POLOŽ
POLOŽ
POLOŽ
POLOŽ
POLOŽ
VPRAVO-VBOK
KROK

atd.

V tomto sloupci je zapsána asi jedna [redacted] programu Dům , pokud nepoužijeme podprogramy. Tento program by byl dlouhý zhruba na 300 řádků.

Obrovskou výhodou psaní programů pomocí procedur je při hledání chyb /tzv. ladění programu/. Posuďte sami, jak by se chyba hledala v pravém sloupci, kdyby byl program zapsán celý na těch 8 stránek. Používejme tedy procedury.

1076/87

Cyklus - opakování se užívají tam, kde je třeba nějakou činnost několikrát zopakovat. Cyklus dělíme na:

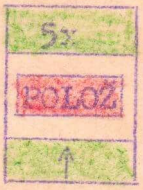
cyklus se známým počtem opakování - příkazy zapsané v těle cyklu se opakují tolikrát, kolikrát určíme.

cyklus s neznámým počtem opakování - příkazy se opakují do té doby než je splněna podmínka. Tyto cykly je vhodné vyučovat až po zvládnutí rekurze, proto o nich podrobněji budeme psát později.

Cyklus se známým počtem opakování se skládá ze záhlaví cyklu, které se vybarvuje zeleně, a z těla cyklu, v němž jsou příkazy, které se mají opakovat. V záhlaví je psán počet opakování.

Tento cyklus se zavádí příkazem OPAKUJ.

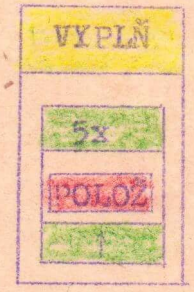
V kopenogramu:



v počítači:

```
OPAKUJ 5 KRÁT
 POLOŽ
KONEC
```

Použití cyklu v příkazu:




```
VYPLŇ znamená
 OPAKUJ 5 KRÁT
  POLOŽ
 KONEC
KONEC
```

Pozn. Jeden KONEC je konec cyklu, druhý je konec příkazu.

Rekurze - deklarování příkazu pomocí sebe sama. Je potřebné ji vyučovat dříve než rozhodovací bloky a ostatní druhy cyklů. Protože rekurze je vlastně název programu, vybarvuje se stejně jako hlava programu žlutou barvou.


```
PIRUETA znamená
 VLEVO-VBOK
 PIRUETA
KONEC
```

Mechanismus provádění rekurze počítačem:

Každý příkaz KAREL provádí tak, že jej "čte" odshora dolů a pokud ~~kem~~ příkaz na daném řádku je primitivum /na ř.KROK/ vykoná jej. Pokud to není primitivum, "podívá se do slovníku,

co tento příkaz znamená".

Použití rekurze předvedeme na příkazu PIRUETA. KAREL dostane příkaz PIRUETA, hledá ve Slovníku, co to znamená. Čte, že PIRUETA znamená VLEVO-VBOK, udělá tedy vlevo vbok a dále čte, že má udělat příkaz PIRUETA. Podívá se tedy do Slovníku a zjistí, že má udělat vlevo vbok a potom piruetu. Vlevo vbok udělá a na piruetu se opět podívá do Slovníku. Tak to pokračuje pořád dál a vlastně dělá pořád vlevo vbok. To znamená, že se točí dokolečka, dělá piruetu. .

Je patrné, že tento příkaz bude KAREL provádět donekonečna. Podobně pracuje příkaz DVOREK nahraný na kazetě. Takových situací, kdy lze k popisu práce využít rekurzi, je kolem nás mnoho, např. ZATLUČ HŘEBÍK znamená

UDEŘ KLADÍVKEM DO HŘEBÍKU,
ZATLUČ HŘEBÍK

nebo pokládání chodníku lze napsat pomocí rekurze jako POLOŽ DLAŽDICI, a nyní rekurzi POKLÁDÁNÍ CHODNÍKU.

Rozhodovací blok umožňuje větvení programu. Někdy je výhodné část programu vykonávat pouze tehdy, když je splněna nějaká podmínka /např. KDYŽ je rozbitý vysavač, opravím ho - - nebudu ho opravovat, není-li rozbitý./

Je-li splněna podmínka, vykoná KAREL příkazy, které jsou napsány pod podmínkou. Není-li splněna, vykoná příkazy napsané pod šipkou.

Rozhodovací blok se uvádí příkazem KDYŽ. Má záhlaví, které se vybarvuje modře, a dvě větve znázorněné vedle sebe.

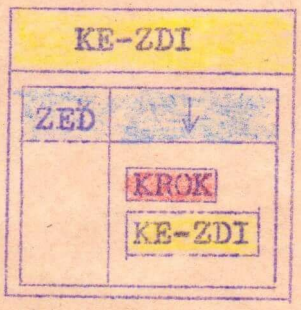

```
UKÁZKA znamená
 KDYŽ je ZNAČKA
  ZVEDNI
 KONEC, JINAK
  POLOŽ
 KONEC
KONEC
```


```
ZVEDNI ZNAČKU znamená
 KDYŽ je ZNAČKA
  ZVEDNI
 KONEC, JINAK
 KONEC
KONEC
```

Na příkazu ZVEDNI ZNAČKU je patrné, že kopenogramy jsou mnohem více přehledné než zápis v počítači a že menší zdržení s vybarvením kopenogramu se nám jistě vyplatí.

Použitím rozhodovacího bloku lze také ukončit rekurzívní programy, tzn. udělat z nich programy konečné.

Příkaz — ke zdi, kdy má KAREL dojít ke zdi, aniž by narazil:



KE-ZDI znamená
KDYŽ je ZEĎ
KONEC, JINAK
KROK
KE-ZDI
KONEC
KONEC

V tomto příkazu se bude rekurzí KROK opakovat tak dlouho, než bude zeď. Karel tedy dojde ke zdi.

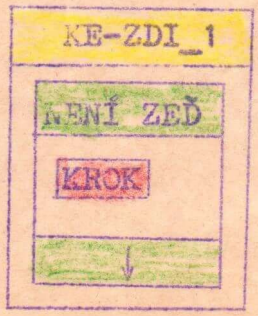
V záhlaví rozhodovacího bloku KAREL testuje, zda je splněna podmínka. KAREL umí testovat tyto podmínky:

je ZNAČKA - KAREL stojí na značce
je ZEĎ - na políčku před KARLEM /ve směru šipky/ je zeď
je SEVER - šipka na KARLOVI směřuje na sever
je JIH, VÝCHOD, ZÁPAD - podobné

Cyklus s neznámým počtem opakování je cyklus, kdy se provádí opakovaně blok příkazů zapsaných v těle cyklu tak dlouho, dokud je splněna podmínka. Teprve při jejím nesplnění cyklus končí.

Tento cyklus se vybarvuje také zeleně, do záhlaví se píše podmínka pro rozhodování.

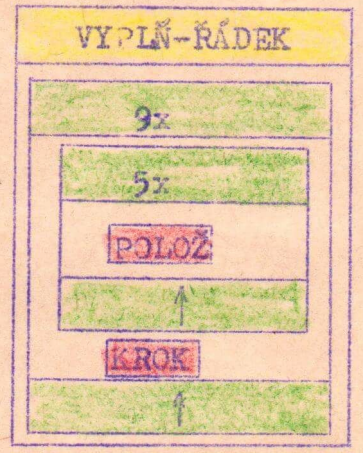
Cyklus se uvádí příkazem DOKUD /DOKUD je, DOKUD není/. Pomocí tohoto cyklu se dá zapsat i příkaz KE-ZDI:

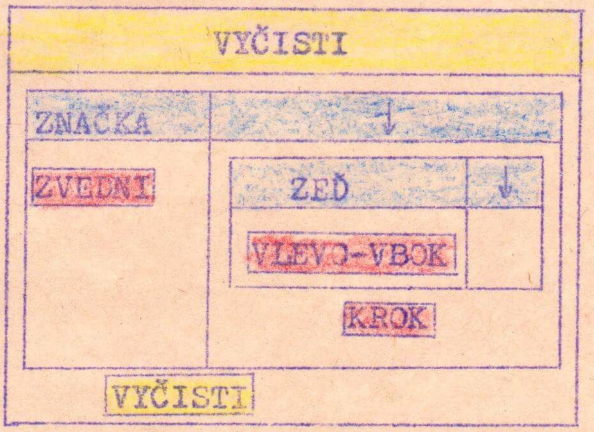


KE-ZDI_1 znamená
DOKUD není ZEĎ
KROK
KONEC
KONEC

Vnořování struktur

Jednotlivé struktury lze do sebe vnořovat. Uvedené příklady to dokumentují:





Používání všech těchto struktur vede k přehlednosti a snadné opravě i kontrole programů. Proto vyžadujeme výuku strukturovaného programování v kroužcích výpočetní techniky.

Po KARLOVI bude následovat jako první programovací jazyk PASCAL, protože používá všechny tyto struktury a má i další výhody. BASIC struktury většinou nemá, proto jej neučíme. Pokud známe PASCAL, používání BASICu si lehce osvojíme.

1660/87

8. Místo závěru pár poznámek k postupu výuky

Cílem výuky KARLA je vychovat novou generaci programátorů, kteří budou schopni obsluhovat výpočetní techniku za 20 let, konstruovat a vybavovat programy počítače 5. a 6. generace, kteří pozdvihnou československou výpočetní techniku na lepší světový průměr, tedy tam, kam bychom jako moderní průmyslově vyspělý stát měli patřit. Tomuto cíli je nutno podřídit vše. Vzhledem k tomu, že naše školství zatím bohužel vzhledem k charakteru používaných počítačů a nevhodnosti metodiky výuky vychovává špatné programátory "bejzikovce" a vzhledem k nedostatku programů nevychovává ani uživatele výpočetní techniky, proto je tento výchovný úkol na nás, klubech výpočetní techniky SSM a Svazarmu.

Je proto nanejvýš důležité vychovávat mladou programátorskou generaci poctivě. Máme k dispozici /vzhledem k používané počítačové technice/ špičkovou světovou metodiku výuky programování. Abychom její sílu dobře využili, je nutno držet se těchto zásad:

1/ vychovejme u dětí vztah a úctu k počítačům. Počítač není morče ani vysavač. Vyžadujme slušné zacházení s ním - čistota na pracovišti, přezůvky do místnosti k počítačům, odpojení počítače ze sítě po skončení schůzky, jeho přikrytí atd.

2/ dbejme na kázeň na pracovišti, neukázněný programátor zkazí víc než udělá nového. Vyžadujme klid, ticho, zásadu, že počítač obsluhuje vždy jen jeden - po chvíli se vystřídají.

3/ důsledně zapisujme veškeré příkazy v kopenogramech a vyžadujme to i po dětech. Dbejme na grafickou úroveň kopenogramů, jejich přehlednost , používání struktur a jejich správné vybarvení. Sami si tím usnadňujeme práci - kontrolu činnosti dětí.

4/ nechme samotné děti přijít na algoritmus příkazu; ošizovali bychom je o radost z vyřešené úlohy, z objevené cesty. Pokud děti samy úkol nevyřeší, předložíme jim podobný úkol, který je lehčí, a po jeho splnění vyžadujme znovu vyřešit úkol původní. Jestliže budeme dětem "prozrazovat" řešení, za chvíli nebudou řešit nic, budou spoléhat na to, že vedoucí jim řešení stejně ukáže, že úkol za ně vyřeší někdo jiný. Tím bychom programování degradovali na mačkání klávesnice.

5/ velmi důležitá je názornost. Děti by si samy měly vyrobit šachovnici 10x10 polí a postavu robota KAREL /stačí šachový jezdec/ a měly by odlaďovat své programy zde ještě dříve, než usednou k počítači. Tak se naučí hledat chyby ve svých programech sami. To je velké umění.

6/ při výuce struktur nešetříme času a pořádně a několikrát podrobně projdeme s dětmi provádění takového příkazu. Zvláště u rekurze se budeme muset k principu činnosti rekurze několikrát vracet.

7/ je známo, že má-li dítě práci, která ho baví, nezlobí. Nejlépe udržíme zájem, nejlépe motivujeme děti soutěžemi /krátkodobými - kdo první sestaví kopenogram, může jít k počítači- i dlouhodobými/, jejichž výsledky budeme zveřejňovat.

8/ úkoly, které zadáváme dětem, by měly být na hranici schopností dětí. Nejvíce motivují k činnosti, zatímco příliš lehký úkol k aktivitě nepovede, stejně jako příliš těžký úkol.

9/ Počítačové hry chápejme jako odměnu za práci v kroužku, jako občasné zpestření činnosti a ukázku toho, co počítač umí. Hrám je vhodné se věnovat vždy asi každou 10. schůzku /slavnostní, o jarních prázdninách, vánoční, na závěr roku.../. Časté hry velmi rychle zbavují motivace a rozkládají kázeň. Žádný člověk se nestal programátorem tím, že prohánět po obrazovce ufouny.

10/ Prosíme Vás, abyste všechny nápady, připomínky, problémy sdělovali k nám do J.Hradce. Např. chceme pořádat okresní soutěž programátorů v KARLOVI na ONDRECH. Budeme velmi rádi, budete-li si s námi vyměňovat nové příkazy, které jste nebo Vaši žáci vymysleli - mohou být i nerešené, jako hádanky - které budeme moci případně i v takovýchto soutěžích použít.

Jakékoliv náměty zasílejte nebo telefonujte na adresy:

Pavel Pokorný
AGRODAT J.Hradec
tel. 2341 - 253

nebo

Jiří Vaníček
1.ZŠ J.Hradec
tel. 2345

Děkujeme předem za spolupráci.

Poslední poučka:
Selhaly-li všechny pokusy, je na čase přečíst si návod.

... a teď již dost řečí. Zasedněme k počítači. ONDRA se zapíná vypínačem na zdroji do polohy 1 ...

1062/87