TECHNICKÁ UNIVERZITA LIBEREC

PEDAGICKÁ FAKULTA

Katedra: Informatiky

Kombinace oborů: Matematika - Informatika

Hypertextový výukový program

- metodika práce s kalkulačkou na počítači

Diplomová práce: 95-PF-IF-03

Autor:

Jiří Šikner podpis: .....

Adresa:

Zemská 1439

415 02 Teplice

Vedoucí práce: RNDr. Petr Kolář

Počet:

stran	obrázků	tabulek	příloh
53	3	19	2

V Liberci dne 18. 5. 1995

Vysoká škola strojní a textilní
PEDAGOGICKÁ FAKULTA
461 17 LIBEREC 1, Hálkova 6 Telefon: 329 Telefax: 21301

Katedra: Katedra informatiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

(závěrečného projektu)

diplomant: **Šikner Jiří**
adresa: Zemská 1439, 415 01 Teplice
obor Informatika

Název: Hypertextový výukový program - metodika práce s kalkulačkou na počítači

Vedoucí práce: RNDr. P. Kolář

Termín odevzdání: 26. 5. 1995

Pozn. Podmínky pro zadání práce jsou k nahlédnutí na katedrách. Katedry rovněž formulují podrobnosti zadání. Zásady pro zpracování DP jsou k dispozici ve dvou verzích (stručné, resp. metodické pokyny) na katedrách a na Děkanátě Pedagogické fakulty.

V Liberci dne 20. října 1994

Doc.Ing. V. Kracík, CSc.
vedoucí katedry

Doc. RNDr. V. Pecina, CSc.
děkan

Převzal (diplomant):
Datum: Podpis: Jiří Šikner

Prohlášení a poděkování

Prohlášení o původnosti práce:

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

Liberec, 15.5.1995 podpis .....

Prohlášení o využívání výsledků DP:

Jsem si vědom těchto skutečností:

a) diplomová práce je majetkem školy
b) s diplomovou prací nelze bez svolení školy disponovat
c) diplomová práce může být zapůjčena či objednána (kopie) za účelem využití jejího obsahu

Beru na vědomí, že po pěti letech si mohu diplomovou práci vyžádat v Univerzitní knihovně Technické Univerzity v Liberci, kde bude uložena.

Jméno a příjmení: Jiří Šikner

Adresa: Zemská 1439, Teplice 415 02

Podpis: .....

Poděkování:

Děkuji všem, kteří se zasloužili o zdárné vypracování této diplomové práce. Děkuji ing. Petru Hutařovi za vhodný námět, dále děkuji Miroslavu Uhrovi za dodané knihovny, které sdílím s jeho programem, RNDr. Petru Kolářovi za konzultace a vedení diplomové práce. Děkuji Karlu Taschnerovi, Miroslavu Uhrovi a Janu Zemanovi za zapůjčenou literaturu a za spolupráci při vytváření společného projektu.

Anotace

Hypertextový výukový program - metodika práce s kalkulačkou na počítači.

Diplomová práce je součástí hypertextového výukového programu *HyperMat*. Práce se zabývá využitím počítače ve výuce na základní škole. Cílem práce je vytvořit modul *kalkulačka* pro program *HyperMat*. Diplomová práce dále obsahuje návod na použití kalkulačky a zpracovává metodiku výuky kalkulačky na základní škole.

Hypertext teaching program - methodology of using calculator on a computer.

Diploma thesis is a part of the hypertext teaching program *HyperMat*. The thesis is engaged in using computer in teaching at a basic school. The aim of the thesis is creation of module *calculator* which is the part of the program *HyperMat*. Diploma thesis further contains manual for operating with the calculator and elaborates the methodology of teaching of the calculator at a basic school.

Hypertextliches Unterrichtsprogramm - die Methodik der Arbeit mit dem Kalkulator

Die Diplomarbeit ist ein Bestandteil des hypertextlichen Unterrichtsprogrammes *HyperMat*. Die Arbeit befaßt sich mit der Computerbenutzung an der Grundschule. Das Ziel der Arbeit ist die Gestaltung eines Modules „*der Kalkulator*" für das *HyperMat*-Programm. Die Diplomarbeit umfaßt weiter die Anleitung der Kalkulatorsbenutzung, und in ihr wird auch die Unterrichtsmethodik des Kalkulators an der Grundschule verarbeit.

Obsah

1. Úvod

Cílem diplomové práce bylo vytvořit modul kalkulačka, který je součástí výukového programu *HyperMat*, a vypracovat metodiku použití kalkulačky na základní škole. Program *HyperMat* je zaměřen na podporu výuky matematiky a dalších předmětů na základní škole, a zejména pak na využití počítačů ve výuce.

Je jisté, že použití počítače vede k velkému zefektivnění mnoha činností. Počítač dnes zcela nahrazuje psací stroj, usnadňuje přenos, vyhledávání a uchování informací (počítačové sítě, databanky), urychluje matematické výpočty, návrhářství a technické kreslení (CAD aplikace). Počítače se používají pro zpracování grafiky (televize), v umění (hudba - tisk notových záznamů, malířství, módní návrhářství) a dnes se bez nich neobejde skoro žádná lidská činnost.

Proto je nutné, aby se každý naučil používat počítač. Nemyslím tím, že by se všichni měli učit programovat v nějakém programovacím jazyku, ale důležité je, aby se naučili ovládat běžné aplikace na počítačích. Proto je vhodné použít program *HyperMat* nejen na podporu výuky matematiky, kde by se pro děti jistě stal vítaným zpestřením, ale také při výuce předmětů informatika nebo výpočetní technika. Zde by se žáci naučili používat tento program a navíc by si procvičili práci s počítačem (použití klávesnice, myši, atd.).

1.1. Členění práce

Celá práce je rozdělena na tři hlavní části. V první části je zpracováno téma kalkulačka (historie, rozdělení kalkulaček, ...) a metodika použití kalkulačky na školách. V další části se pak budeme zabývat programem kalkulačka a řešením některých problémů v programu. Část „Popis kalkulačky" je vlastně návodem (manuálem) na použití programu kalkulačka. V přílohách pak jsou výpisy hlavičkových souborů a návod na ovládání kalkulačky pomocí klávesnice.

1.2. Použité softwarové produkty

Program byl napsán v programovacím jazyku *Borland C++, verze 3.1*. Dále byly při psaní práce použity *Paintbrush* a další aplikace pod *Windows* a textové editory *Microsoft Word*, verze 6.0 a *T602*.

1.3. Program kalkulačka

Při práci s počítačem může nastat situace, kdy potřebujeme provést jednoduché matematické výpočty a nemáme u sebe kalkulačku. Ačkoli máme před sebou zařízení, které dokáže provádět složité operace, nemůžeme vypočítat snadné matematické příklady se základními funkcemi a operacemi. Právě v takovém případě najde program kalkulačka své uplatnění.

2. Kalkulačka

2.1. Co je kalkulačka?

Kalkulačka je, z hlediska využití, zařízení, které dokáže ve velmi krátké době vypočítat základní matematické operace a funkce. Z technického hlediska je to v podstatě „degenerovaný" počítač, navržený pro speciální matematické účely.

Ovšem v dnešní době by definice byla daleko obecnější. Pod pojmem kalkulačka bychom totiž mohli uvažovat i diáře nebo programovatelné kalkulačky, které dokážou pracovat s pamětí RAM, mají v sobě zabudovány databáze a tabulkové kalkulátory kompatibilní s programy LOTUS 123 nebo Quattro PRO a navíc je umožněno jejich propojení (přenos dat) s počítači.

V této práci se však budeme řídit podle definice na začátku kapitoly. To znamená, že nebudeme hovořit o programovatelných kalkulátorech ani o diářích, které se v poslední době tak rozšířily, ale jen o přístroji určenému k provádění základních matematických výpočtů.

2.2. Historický vývoj

Opakované provádění základních matematických operací je velice únavné, nepohodlné a pomalé. Vlivem únavy mohou vzniknout chyby, a proto se nelze divit snaze vědců o vytvoření stroje provádějícího rychlé a bezchybné výpočty.

Jeden z prvních takových strojů, fungujících na mechanickém principu, byl vytvořen r. 1623 W. Schickartem a umožňoval provádění čtyř základních operací. Z dnešního hlediska bylo toto zařízení složité a nepřesné. Další počítací stroj byl sestaven r. 1641 známým matematikem a filosofem B. Pascalem, ale ani tuto konstrukci nelze považovat za dokonalou. Až na sklonku 19. století se objevily takové konstrukce, které lze pokládat za přímé předchůdce mechanických kalkulátorů. Hromadná výroba mechanických kalkulátorů byla zahájena až ve 30. letech 20. století.

V 50. letech začala hromadná výroba nového účinného výpočetního prostředku - počítače. Je zajímavé, že výrobci mechanických kalkulátorů si nepovšimli významu objevu elektronických počítačů a první elektrický kalkulátor vyvinula až r. 1962 anglická firma, která nikdy mechanické kalkulátory nevyráběla. Výrobek o velikosti psacího stroje obsahoval na několika deskách plošných spojů tisíce tranzistorů. Tisíce pájených spojů nelze realizovat zcela bezporuchově, a to je příčinou malé spolehlivosti prvních elektrických kalkulátorů.

Vytvoření dnešní formy kapesních kalkulátorů umožnil až vývoj integrovaných obvodů (obvody LSI - Large Scale Integration). Koncem 60. let byla zahájena hromadná výroba obvodů LSI, potřebných pro výrobu kalkulaček. Prvními firmami, které vyrobily skutečně levnou, hromadně vyráběnou kalkulačku, byly americká Comptometer, japonská Omron a kanadská Commodore. Roku 1971 americká firma Bowmar zavedla displej se světelnými diodami.

Víme, že dnes je vývoj mnohem dále. Používá se LCD displej, k napájení se využívá solární energie a kalkulačka

je dnes cenově přístupná snad i pro sociálně nejslabší. Navíc, je to již dávno, kdy jsme se mohli setkat s jednoduchou kalkulačkou na hodinkách.

2.3. Registr a paměť

Registrem rozumíme část kalkulačky schopnou přijmout, uchovat a vydat určitou informaci. Informací může být číslo nebo instrukce. Registr pro uložení instrukce (*instrukční registr*) obsahuje příkaz k provedení operace (např. vynásob, přičti, ...).

Soubor registrů tvoří paměť kalkulačky. Zde je třeba rozlišit operační paměť, kterou kalkulačka používá pro své výpočty, a datovou paměť, která je volně přístupná uživateli a do které se ukládají čísla. Operační paměť se skládá ze dvou a více registrů pro uložení operandů (čísel) a z jednoho a více instrukčních registrů.

Musíme si uvědomit rozdíl mezi pamětí kalkulačky, která se skládá z operační a datové paměti, a paměťmi ve smyslu datových pamětí, do kterých si uživatel může ukládat mezivýsledky nebo pomocné výsledky.

2.4. Typy kalkulaček

2.4.1. Základní rozdělení

Obecně nazýváme operací vytvoření výsledků na základě jednoho nebo více údajů - tzv. operandů. Operace mohou být jednočlenné (unární), dvojčlenné (binární) nebo vícečlenné. Operaci charakterizuje tzv. operátor (např. +, -, sin, cos, log, ...).

Při zápisu unární operace lze umístit operátor před operand (tzv. *prefixový* způsob zápisu), což je v matematické symbolice běžné, nebo za operand (tzv. *postfixový* zápis operace), což se používá většinou na kalkulačkách. Porovnejme tyto způsoby zápisu podle tab. 1:

Tab. 1 Srovnání prefixového a postfixového způsobu zápisu

operace	prefixový způsob	postfixový způsob
$\sin x$	sin x	x sin
$\ln x$	ln x	x ln
$\sqrt{x}$	√ x	x √

Při zápisu binární operace můžeme navíc umístit operátor i mezi operandy (tzv. *infixový* zápis). Viz tab. 2:

Tab. 2 Srovnání infixového a postfixového způsobu zápisu

operace	infixový způsob	postfixový způsob
a + b	a + b	a b +
a / b	a ÷ b	a b ÷
mocnina a^b	a y^x b	a b y^x

Způsob zápisu binárních operací je kritériem pro základní rozdělení kalkulaček. Postfixový zápis používají kalkulačky s *reverzní polskou* logikou (dále budeme psát jen polská logika). U těchto kalkulátorů se používá postfixový zápis i pro unární operace. Polská logika se používá například u registračních pokladen. Infixový zápis používají kalkulačky s logikou *aritmetickou* a *algebraickou*. Rozdíl mezi aritmetickou a algebraickou logikou je dán tím, že kalkulačky

s aritmetickou logikou nedodržují priority operací, ale počítají průběžně každou operaci (další operátor uzavírá předchozí operaci), zatímco kalkulačky s algebraickou logikou priority operací dodržují. Tento rozdíl ještě lépe objasní tabulka 3:

Tab. 3 Srovnání aritmetické a algebraické logiky

aritmetická logika	výsledek	algebraická logika	výsledek
8 [-] 5 [=]	3.	8 [-] 5 [=]	3.
8 [-] 5 [×] 6 [=]	18.	8 [-] 5 [×] 6 [=]	-22.

- Násobení má vyšší prioritu než odčítání, a proto se u algebraické logiky provede nejprve násobení.
- Nejlepší je, když si sami ověříme s jakou logikou kalkulačka pracuje. Kalkulátor HP 17 B II umožňuje výpočty s reverzní polskou logikou a s aritmetickou logikou, avšak aritmetická logika je chybně nazvána jako algebraická (*algebraic mode*).

Unární operace se u aritmetických systémů zadávají postfixově, tedy operátor až za operandem. U algebraických systémů se unární operace zadávají dvojím způsobem. Nejrozšířenější je již uvedený postfixový způsob. Druhý způsob rozděluje unární operace do dvou tříd:

1) operace zadávané prefixově $\sin x, \cos x, \ln x, \sqrt{x}, \sqrt[3]{x}, \ldots$
2) operace zadávané postfixově $x^2, x^{-1}, x!, \ldots$

- Všimněme si, že způsob zápisu odpovídá matematickému zápisu a čtení zadání příkladu (prefixově - sinus ***x***, cosinus ***x***, odmocnina z ***x***, ...; postfixově - ***x*** na druhou, ***x*** faktorial, ...)
- Tento druhý způsob je z metodického hlediska nejlepší, protože odpovídá matematické logice zápisu. Program kalkulačka pracuje s touto logikou zápisu.

Pro prioritu operací dodržují všechny algebraické

systémy tuto úmluvu:

1) Nejprve se vyhodnotí operace v závorkách
2) potom unární operace
3) pak binární operace (mocnina)
4) násobení a dělení
5) sčítání a odčítání
6) nejnižší úroveň má operátor [=]

Některé kalkulátory používají pro zápis binárních operací operátor za i mezi operandy, mají tzv. *smíšenou* (obchodní) logiku. Pro součet a rozdíl se vkládá operátor za operandy, u dělení a násobení mezi operandy. Příklad:

$(8-5)\times 4$ zadáme takto ... 8 [+] 5 [−] [×] 4 [=]

♦ Pro školní účely je tento systém naprosto nevhodný.

Na závěr si porovnejme všechny jmenované logiky podle tabulky 4:

Tab. 4 Porovnání logických systémů

logika	$(8-5)\times 4$	$8-5\times 4$
polská	8 [↑] 5 [−] 4 [×]	8 [↑] 5 [↑] 4 [×] [+]
aritmetická	8 [−] 5 [×] 4 [=]	5 [×] 4 [+/−] [+] 8 [=]
		8 [−] [(] 5 [×] 4 [)] [=]
algebraická	[(] 8 [−] 5 [)] [×] 4 [=]	8 [−] 5 [×] 4 [=]
smíšená	8 [+] 5 [−] [×] 4 [=]	5 [×] 4 [=] [−] 8 [+]

♦ Všimněme si, že algebraický zápis odpovídá běžnému matematickému zápisu. Proto se také doporučuje použití algebraického systému.

♦ Symbol zobrazený na klávese [↑] je často nahrazen nápisem ENTER resp. INPUT.

2.4.2.Rozdělení podle paměti

2.4.2.1.Kalkulačky s jedním operačním registrem

Základem každého kalkulátoru jsou dva registry. Registr vstupní, který je zpravidla spojen s displejem, a registr operační. Každý kalkulátor samozřejmě potřebuje alespoň tyto

dva registry, aby mohl provádět binární operace, tedy operace se dvěma operandy. U polské a aritmetické logiky je tento počet registrů postačující.

2.4.2.2.Kalkulačky se dvěma a více operačními registry

Při některých výpočtech je potřeba vložit několik operandů a teprve potom s nimi provádět požadované operace. K těmto výpočtům je potřeba použít několik operačních registrů.

Kalkulačky s aritmetickou logikou mohou počítat se závorkami, mají-li více operačních registrů.

Kalkulačky s algebraickou logikou nutně vyžadují více operačních i instrukčních registrů. V paměti jsou zpravidla čtyři až osm operačních registrů a až šestnáct instrukčních registrů.

2.4.3.Další možnosti kalkulaček

2.4.3.1.Vynechání operátoru násobení

Vynechat operátor pro násobení lze:

a) Před závorkou.

Příklad $5\times(3+4)$ můžeme zadat jednoduše takto -

5 [(] 3 [+] 4 [)] [=]

b) Při násobení čísla proměnnou (číslem v datové paměti), při násobení dvou proměnných a při násobení čísla Ludolfovým číslem.

Výrazy 2π, $3a$, bcd vypočítáme na kalkulačce takto:

2 [π], 3 [RCL] [A], [RCL] [B] [RCL] [C] [RCL] [D]

Součin vkládaný bez operátoru mívá obvykle vyšší prioritu než součin s operátorem. Výraz $\frac{4a}{8bc}$ vypočítáme následovně:

4 [RCL] [A] [÷] 8 [RCL] [B] [RCL] [C] [=]

c) Před unární funkcí zadávanou prefixově.

Příklad $3\sin 5+2\cos 5$ zadáme takto:

3 [sin] 5 [+] 2 [cos] 5 [=]

2.4.3.2. Vkládání závorek

V matematickém zápisu vždy požadujeme, aby ke každé levé závorce existovala i závorka pravá. Navíc pak tvarem rozlišujeme, které dvojice závorek k sobě patří. Ve výpočetní technice se používá pouze jeden druh závorek a matematický výraz se vyhodnocuje od vnitřního páru závorek. Pokud je na kalkulačce otevřen větší počet levých závorek, pak pravou závorkou vždy vyčíslíme výraz mezi naposled vloženou levou závorkou a právě zadanou pravou závorkou a výsledek tohoto výrazu se objeví na displeji.

Některé kalkulačky nevyžadují vložení pravých závorek a po stisku tlačítka  si samy pravé závorky doplní a výraz vyhodnotí.

2.4.3.3. Vnitřní mantisa

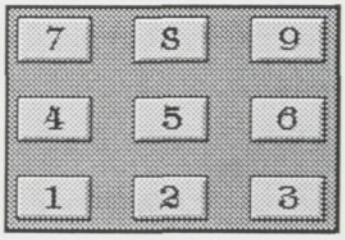
Lepší typy kalkulaček obvykle počítají s mantisou o jednu nebo více cifer větší, než lze zobrazit na displeji. Takové kalkulačky pak umožňují přesnější výpočty. Číslo zobrazené na displeji se u těchto typů kalkulaček zaokrouhluje.

2.5. Hry a příklady

2.5.1. Bitva

Zapišme na kalkulačku jedno celé kladné číslo[1] (je vhodné volit jej mezi hodnotami 50 - 150). Úkolem dvou hráčů je střídavě odečítat od čísla na displeji jedno jednomístné číslo (samozřejmě kromě nuly), které však musí na klávesnici kalkulátoru sousedit s číslem, které volil předchozí hráč. Když si jeden z hráčů zvolil klávesu 4, pak protihráč může volit mezi klávesami 1, 2, 5, 8 a 7. Zvolí-li jeden hráč pětku, pak protihráč může volit všechny klávesy kromě pětky.

Obr. 1 Rozložení kláves 1 - 9



Prohrává hráč, u kterého se poprvé na displeji objeví záporné číslo.

2.5.2. Nula vyhrává

Zapišme do kalkulačky šestimístné číslo, u kterého je první číslice menší než 5 a jednotlivé číslice jsou různé. V každém kroku si pak můžeme zvolit jednu ze čtyř základních operací (sčítání, odčítání, násobení nebo dělení) a jedno libovolné dvouciferné číslo. Vyhrává ten kdo má na displeji nulu po čtyřech krocích.

Tato hra je velmi vhodná pro celou třídu. Žáci se snaží dojít k výsledku nula s co nejmenším počtem kroků. Ze začátku samozřejmě nemůžeme očekávat, že žáci dokáží vyřešit úlohu na čtyři kroky. Vyhrává žák resp. žáci s nejmenším počtem kroků (můžeme hodnotit nejlepší žáky známkou).

---

[1] U kalkulaček, které to umožňují, můžeme použít generování náhodného čísla.

Například: Do kalkulátoru zapíšeme číslo 368524;

1) přičteme číslo 26, výsledek je 368550;
2) vydělíme číslem 50, výsledek je 7371;
3) vydělíme číslem 91, výsledek je 81;
4) odečteme číslo 81, výsledek je 0.

2.5.3.Sudý a lichý

Jeden hráč, kterého určíme losem, bude sudý, protihráč pak bude lichý. Hráči si napíší na papír číslice 0 až 9. Lichý hráč začne tak, že si vybere jednu číslici a napíše jí do kalkulačky. Vybranou číslici pak škrtne ze seznamu na papíru. Druhý hráč si pak vybere jednu, ještě nepřeškrtnutou číslici a tu k číslu na displeji přičte resp. odečte nebo vynásobí resp. dělí číslem na displeji. Dělit může jen v případě, že výsledek je celé číslo. Vybranou číslici opět vyškrtne ze seznamu. První hráč pokračuje podobně a hráči se střídají dokud jeden z nich neprohraje.

A nyní nejdůležitější pravidla:

- V okamžiku, kdy hráč předává protihráči, musí být na displeji sudé číslo u sudého hráče a liché číslo u hráče lichého. Není-li tomu tak je hra skončena a hráč prohrává.
- Každé číslo se smí použít maximálně jednou (ke kontrole slouží papír, na kterém přeškrtáváme čísla)

Ukázka hry:

- LICHÝ: Musí vybrat lichou číslici, jinak by prohrál hned v prvním kole. Vybral 1.
- SUDÝ: Přičte 7 a výsledek je 8.
- LICHÝ: Dělí 4 a výsledek je 2.
- SUDÝ: Přičítá 8 a výsledek je 10.
- LICHÝ: Dělí 2 a výsledek je 5.
- SUDÝ: Odečítá 3 a výsledek je opět 2.
- LICHÝ: Přičte 9 a na displeji je číslo 11.
- SUDÝ: Násobí 0 a na displeji 0.
- LICHÝ: Přičítá číslo 5 a na displeji je 5.
- SUDÝ: Násobí 6 a vyhrává, protože lichý už nemá tah.

2.5.4.Textové údaje

Při otočení kalkulačky zjistíme, že některé číslice, čteme-li je po hlavě, se podobají znakům písmen.

- nula: písmeno O nebo D
- jedna: písmeno I nebo malé l
- dvě: písmeno Z
- tři: písmeno E
- čtyři: malé písmeno h
- pět: písmeno S
- sedm: písmeno L
- osm: písmeno B
- devět: písmeno G

Necháme žáky experimentovat a tvořit slova (les, gel, goool, atd.). Samozřejmě můžeme očekávat, že žáci již znají slova jako osel, blb, ... I když to nejsou zcela vhodné výrazy na škole, věřím, že žáky pobaví a motivují (probudí) k další práci.

2.5.4.1.Hra s čísly

Zvolíme libovolné číslo x. Vypočteme hodnotu x^2-1. Potom násobíme číslem 150. Dále dělíme výrazem x+1 a opět násobíme, tentokrát hodnotou 25. Nakonec vydělíme číslem x-1. Vždy nám vyjde výsledek 3750. Po otočení kalkulačky můžeme číst „OSLE".

Žáci by měli sami přijít na to, že jsme vlastně použili vzorec, a proto se po vykrácení výrazu x^2-1 výrazy x-1 a x+1 výpočet zjednodušil jen na násobení čísel 150 a 25. Tento příklad je vhodný ke zpestření výuky v osmé třídě ZŠ, kde se probírají vzorce typu:

$$a^2 - b^2 = (a-b) \times (a+b)$$
$$(a+b)^2 = a^2 + 2ab + b^2$$
$$\vdots$$

2.6. Metodické poznámky

2.6.1. Jak vyučovat

Na základní škole se děti dovídají o kalkulačce v předmětech matematika a fyzika. Na podrobnou výuku však vůbec není čas. Proto by bylo potřeba přímo zařadit předmět kalkulačka do osnov základní školy, nejlépe do osmého nebo devátého ročníku, nebo vyučovat kalkulačku v rámci předmětu výpočetní technika.

Další ušetření času by umožnilo nahrazení používání matematických tabulek kalkulačkou. Dodnes mi totiž uniká smysl používání těchto tabulek a navíc jsem se i mezi učiteli setkal s názory, že používání tabulek je v mnoha případech nesmyslné.

V předmětu kalkulačka bych pak doporučil:

1) Seznámit s historickým vývojem.
2) Vysvětlit pojmy registr, paměť, displej a princip kalkulačky.
3) Popsat rozdíly mezi logikami kalkulaček a doporučit algebraickou logiku (nejlépe pak doporučit tuto kalkulačku přímo rodičům žáků).
4) Dále pak porovnat tyto logiky v praxi.
5) Procvičovat příklady na kalkulačce s algebraickým systémem (základní operace, použití závorek, počítání s úhly, výpočty se zlomky[2], mocniny, goniometrické a cyklometrické funkce).
6) Zařadit do výuky pro zpestření hry na kalkulátorech.

2.6.2. Doporučení

Kalkulačky s polskou a smíšenou logikou jsou na základních školách zcela nevhodné. Kalkulačky s aritmetickou logikou jsou na školách nejrozšířenější. Důvodem je asi jejich nižší cena, zvláště pak v minulosti. Dnes jsou však už i kalkulačky s algebraickou logikou cenově dostupné, a navíc jsou optimálním typem pro základní a střední školy, protože

---

[2] Dodaný program kalkulačka výpočty se zlomky umožňuje.

respektují priority matematických operací. Proto bych právě algebraické systémy doporučil na školách.

Ještě vhodnější je z metodického hlediska použití systému s přímou algebraickou logikou, tedy s logikou umožňující prefixové zadávání operací (operace před operandem) [KKŠ, str. 13].

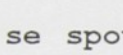
Velmi výhodné pak je, umí-li kalkulačka počítat se zlomky a úhly ve tvaru *celá část - minuty - vteřiny* a doporučuje se, aby prováděla základní funkce, které se používají na základních školách.

Při práci s kalkulačkou je dobré vést žáky k tomu, aby předem odhadovali výsledky a aby používali papír a tužku ke psaní mezivýsledků. Odhad výsledků před výpočtem umožní závěrečnou kontrolu výpočtu a navíc vede žáky k procvičení numerického počítání (vidíme, že navzdory některým názorům použití kalkulačky nemusí vést k „matematické degeneraci" žáků). Psaní mezivýsledku později umožní najít chyby ve výpočtu (učiteli to pak usnadní kontrolu žáků). Je také dobré, naučit žáky opakovat pro kontrolu aspoň jednou výpočet na kalkulačce.

Je důležité, aby se každý učitel dokonale seznámil s typy kalkulátorů. Vyučující, který toto nezvládne, není schopen zodpovědět dotazy žáků. (většina žáků má svou kalkulačku a jejich systémy se liší). Není dobré nutit žáky, aby používali školní kalkulátory, naopak, pro každého je výhodné naučí-li se pracovat s vlastní kalkulačkou a zvykne-li si tuto kalkulačku používat.

3. Popis programu kalkulačka

3.1. Základní informace

Program kalkulačka je napsán v programovacím jazyku *Borland C++, verze 3.1.* Součástí programu jsou soubory „CALC.EXE", „CALC.PCX", „EGAVGA.BGI". Program je vytvořen pro počítače řady PC AT pro operační systém MS-DOS (zde bych rád napsal, že jsem se s podobným programem v tomto OS nesetkal - hledal jsem i v archivech sítě CesNet a Internet[3]). Program se spouští příkazem „CALC" a opouští se stiskem kláves  nebo stiskem klávesy F10 na klávesnici počítače.

Program je velmi snadné upravit tak, aby nebyl samostatným spustitelným souborem, ale součástí jiné aplikace. Při psaní jsem totiž využil výhody objektového programování, které jazyk *Borland C++* nabízí. Proto je velmi snadné zakompilovat objekt Calc do jakéhokoliv jiného programu (viz výpis souboru „DIPL.CPP" v příloze 1 této práce).

3.1.1. Ovládání programu

Program je možné ovládat jak pomocí myši tak i klávesnice. Chceme-li program ovládat pomocí myši, pak je nutné nahrát před spuštěním ovladač myši do paměti. Není-li ovladač myši nahrán v paměti, vypíše se varovné hlášení, ale po stisku libovolné klávesy je možno v programu pokračovat, avšak ovládat ho můžeme jen pomocí klávesnice.

Při použití funkcí kalkulačky doporučuji ovládání pomocí myši, protože je to jistě rychlejší a neuděláme tolik chyb. Ovšem používáme-li pouze základní operace, pak je práce s klávesnicí mnohem rychlejší. Z toho neplyne nic jiného, než že je nejlepší použití klávesnice a myši vhodně kombinovat.

---

[3] Vynaložené úsilí na napsání programu se zřejmě nevyplatí - každý si raději koupí kalkulačku než program. Cena kalkulačky srovnatelné s tímto programem je asi 600 Kč.

Popis jak ovládat kalkulačku pomocí klávesnice je umístěn v příloze 2 této práce.

3.1.2.Hardwarové a softwarové nároky

Program je určen pro počítače řady PC AT, tedy od počítače s procesorem 80286 a výše, s 1 MB RAM a s grafickou kartou VGA nebo SVGA (program pracuje v grafickém režimu v rozlišení 640 × 480). Dále se doporučuje vybavení počítače myší (ovšem není nutné). Program pracuje v OS MS-DOS.

Program byl testován na počítačích s těmito konfiguracemi:

- 286, 1 MB RAM, VGA, MS-DOS 5.00, 2-tlačítková myš s ovladačem MMOUSE.COM
- 486 DX 50 MHz, 4 MB RAM, SVGA Cirrus Logic 5428, MS-DOS 6.00, 3-tlačítková myš Genius s ovladačem GMOUSE.COM i bez ovladače
- 486 DX2 66 MHz, 8 MB RAM, SVGA Cirrus Logic 5428, MS-DOS 6.20, 2-tlačítková myš s ovladačem MMOUSE.COM
- 386 SX 25 MHz, 4 MB RAM, VGA Trident, MS-DOS 6.00, 3-tlačítková myš Genius s ovladačem MMOUSE.COM
- Apricot 386 SX 20 MHz, 6 MB RAM, VGA, MS-DOS 6.00, 2-tlačítková myš s ovladačem MMOUSE.COM

3.2. Členění programu

3.2.1.Objekty

Program je logicky členěn do tří objektů:

- Calc: Je hlavní objekt celého programu. Zde se inicializují některé proměnné a také objekty programu. Procedura PAINT se postará o nakreslení kalkulačky. Zde je také umístěna smyčka, která přijímá zprávy myši a klávesnice a dále je posílá objektu Task. Další funkce v tomto objektu se starají o zobrazení čísla na displeji, o překreslení kláves při stisku a o konverzi čísla na řetězec ve správném formátu.
- Task: Objekt zpracovává veškeré vstupy, které přichází z objektu Calc. Postará se o vytvoření dynamického seznamu, ve kterém je vlastně uložen celý příklad, a průběžně tento seznam zpracovává. To znamená, že podle priorit kalkulačky průběžně počítá funkce a provádí operace.
- Objekty typu TCalcString: V těchto objektech se provádí operace s řetězci v závislosti na typu řetězce. Například zápis mantisy se liší od zápisu exponentu. Jsou zde funkce, které se starají o sčítání dvou řetězců, přidání znaku k řetězci, odebrání znaku z řetězce, atd.

3.2.2.Zdrojové soubory

V této kapitole popíšu zdrojové soubory programu. Ke každému souboru, kromě souboru DIPL.CPP, existuje ještě hlavičkový soubor (s příponou .H), ve kterém jsou deklarovány funkce a proměnné. Výpis hlavičkových souborů a souboru DIPL.CPP je v příloze 1.

- DIPL.CPP: V tomto souboru jsou provedeny všechny počáteční inicializace, tzn. inicializace grafiky, myši a kalkulačky. Po inicializaci kalkulačky dále běží program pomocí smyčky v souboru CALC.CPP. Na konec programu se ještě ukončí práce zavřením grafiky.
- CALC.CPP: Zde je definována třída TCalc, definován typ TButton a také deklarován objekt Task. Je zde také funkce CalcMouse, která zpracovává zprávy myši.

- CALCSTR.CPP: Zde je definována třída TCalcString.
- CALCMATH.CPP: Zde je umístěna třída TTask a její součástí je také typ PRegister. Proměnné typu PRegister jsou pak prvky seznamu.
- J_MATH.CPP: Zde jsou definovány některé matematické funkce, které nenabízí standardní knihovna MATH (převod číselných soustav, faktorial, ...).
- CONST.CPP: Zde je deklarována externí proměnná CS_Error a konstanty.

V souboru TYPES.H jsou definovány některé datové typy. Některé soubory (MOUSE.LIB, BMP.CPP, PLANES.CPP) jsem převzal. Tyto soubory se starají o vykreslení obrázku CALC.PCX a o odchycení zpráv myši.

3.3. Techniky výpočtu některých funkcí

3.3.1. Goniometrické funkce

Při psaní programu jsem si všiml, že počítač se dopouští u výpočtu goniometrických funkcí jedné nepřesnosti, které lze velmi snadno zabránit. Při výpočtu funkce sinus má být v periodě $k\pi \mid k \in Z$ výsledek 0, avšak standardní funkce jazyka *Borland C++* vrací hodnotu pouze blízkou nule (řádově 10^{-16}). Stejně je tomu u funkce cosinus v periodě $\frac{\pi}{2} + k\pi \mid k \in Z$ a u funkce tangens v periodě $k\pi \mid k \in Z$. Odstranit tuto chybu je velmi snadné. Stačí „odchytit" tyto periody. Všiml jsem si, že této chyby se dopouští i standardní kalkulačka ve WINDOWS, i když odstranění chyby je tak snadné.

3.3.2. Hyperbolické funkce a funkce k ním inverzní

V této kapitole pouze popíšu jak jsou matematicky definovány hyperbolické funkce. U inverzních hyperbolických funkcí jsem byl nucen tyto definice použít v programu, protože standardní knihovny jazyka *Borland C++* je nenabízí.

- Hyperbolické funkce

$$\sinh x = \frac{1}{2}\left(e^{x} - e^{-x}\right)$$

$$\cosh x = \frac{1}{2}\left(e^{x} + e^{-x}\right)$$

$$\tanh x = \frac{\sinh x}{\cosh x}$$

- Inverzní hyperbolické funkce

$$\operatorname{argsinh} x = \ln\left(\sqrt{x^2 + 1} + x\right)$$

$$\operatorname{arg\,cosh} x = \ln\left(\sqrt{x^2 - 1} + x\right) \mid x \geq 1$$

$$\operatorname{argtanh} x = \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right) \mid \mid x \mid < 1$$

3.3.3. Převod souřadnic

♦ **Převod z kartézských do polárních souřadnic:**

$(x,y) \rightarrow (r,\theta)$

$r = \sqrt{x^2 + y^2}$

$x = 0 \;\&\; y > 0 \Rightarrow \theta = \frac{\pi}{2}$

$\&\; y < 0 \Rightarrow \theta = \frac{3\pi}{2}$

$x > 0 \;\&\; y \geq 0 \Rightarrow \theta = \operatorname{arctg}\frac{y}{x}$

$\&\; y < 0 \Rightarrow \theta = 2\pi + \operatorname{arctg}\frac{y}{x}$

$x < 0 \Rightarrow \theta = \pi + \operatorname{arctg}\frac{y}{x}$

♦ **Převod z polárních do kartézských souřadnic:**

$(r,\theta) \rightarrow (x,y)$

$x = r \times \cos\theta$

$y = r \times \sin\theta$

4. Popis kalkulačky

4.1. Displej

Obr. 2 Displej

Je-li číslo zobrazené na displeji v intervalu $\pm 0.000000001 \sim \pm 9999999999$, kalkulačka zobrazuje pouze mantisu. Jinak se čísla převedou na exponenciální tvar. Na funkčním displeji jsou zobrazeny instrukce výpočtu a symboly.

M :

Udává, že v nezávislé paměti je uloženo číslo.

2ndF :

Objeví se na displeji po stisknutí klávesy 2ndF a udává, že funkce zobrazené žlutě jsou přístupné.

HYP :

Objeví se na displeji po stisknutí klávesy hyp a udává, že hyperbolické funkce resp. funkce k ním inverzní (2ndF) jsou přístupné.

DEG RAD GRAD :

Udává úhlové jednotky.

FIX SCI ENG :

Udává formát zápisu čísla na displej.

STAT :

Udává, že byl nastaven mód *statistika*.

4.2. Před použitím kalkulačky

Kalkulačka provádí výpočty s použitím přímé algebraické logiky - D.A.L. (*Direct Algebraic Logic*). Systém vstupu D.A.L. umožňuje zadání příkladu tak jak ho čteme, např. $\sin 30 + \cos 45$ se jednoduše zadá jako [sin] 30 [+] [cos] 45 [=]. Výpočty, od nejjednodušších až k nejrozsáhlejším vzorcům, mohou být prováděny velmi jednoduchým a snadno srozumitelným způsobem.

4.2.1. Popis kláves použitý v tomto manuálu

V manuálu jsou všechny operace s klávesnicí popsány následovně:

ln

funkce ln

funkce e^x

číslo E v hexadecimální soustavě

x

používá se při výpočtech s pamětí

Čísla nejsou zobrazena jako klávesy, ale jako běžná čísla.

4.2.2.Mazání

Kalkulačka umožňuje tyto tři způsoby mazání:

Maže všechna čísla a instrukce (všechny registry) v kalkulačce (kromě statistických dat v módu STAT a pamětí v normálním módu). Při zadávání čísla maže jen hodnotu na displeji. Ke smazání všech hodnot a instrukcí v kalkulačce musí být  stisknuto dvakrát.

Maže všechny hodnoty a instrukce v kalkulačce, kromě obsahu nezávislé paměti v normálním módu. Maže všechna data v módu STAT.

Maže poslední číslici zadávaného čísla.

Kalkulačka navíc umožňuje přepisování poslední operace, a to při dodržení priorit. Navzájem přepisovat lze tyto funkce

1. Y^X, $\sqrt[X]{Y}$, nCr, nPr , $\times$, $\div$, $+$, $-$, AND, OR, XOR, XNOR
2. Funkce, po kterých následuje argument (sin, cos, atd.)

4.2.3.Priority

Kalkulačka provádí výpočty podle následujících priorit:

1. π , vyvolání paměti
2. Funkce, kterým předchází argument[4] (%, x^{-1}, x^2, n!, atd.)
3. Y^X, $\sqrt[X]{Y}$, nCr, nPr [5]
4. Funkce, po kterých následuje argument[6] (sin, cos, atd.)

---

[4] funkce zadávané prefixově

[5] funkce zadávané infixově

[6] funkce zadávané postfixově

5. ×, ÷

6. +, −

7. AND

8. OR, XOR, XNOR

9. =, M+, M-, ⇒M, ▷D, ▷R, ▷G, →BIN, →OCT, →HEX, →DEC, DATA, CD, →rθ, →xy a další operace ukončující výpočet.

♦ Při použití závorek mají závorky přednost před všemi ostatními funkcemi a operacemi.

4.3. Základní nastavení

4.3.1. Výběr módu

Kalkulačka může pracovat v jednom ze dvou módů popsaných níže. Pro výběr módu se používá klávesa Mode. Klávesa funguje jako přepínač.

1. Normální mód (STAT není zobrazeno na displeji) - je používán k provádění aritmetických operací a funkcí.
2. Statistický mód (STAT se ukáže na displeji) - je používán k provádění statistických výpočtů.

4.3.2. Formát zobrazení hodnoty na displeji

Tato kalkulačka má následující čtyři formáty zobrazení čísel (*display notation*):

FIX :

Formát s pevnou řádovou čárkou (*fixed decimal point system*). Číslo je zobrazeno se zvoleným počtem desetinných míst (viz další kapitola). Na displeji se objeví nápis FIX.

SCI :

Exponenciální tvar zobrazení čísla. Někdy se též používá vědecká notace[7] (*scientific notation*). Číslo je zobrazeno v exponenciálním tvaru resp. ve tvaru mantisa-exponent. Mantisa je zobrazena se zvoleným počtem desetinných míst. Na displeji se objeví nápis SCI.

[7] Tento doslovný překlad se mi však zdá naprosto nevhodný.

ENG :

Technický tvar zobrazení čísla[8] (*engineering notation*). Jedná se vlastně o exponenciální tvar s tím rozdílem, že exponent je navíc nastaven na násobek tří (např. 1.234 E+14[sci]=123.4 E+12[eng]). Na displeji se objeví nápis ENG.

Základní :

Formát pohyblivé řádové čárky (*Floating point system*). Číslo je zobrazeno bez ohledu na zvolený počet desetinných míst. Ani jeden z nápisů FIX, SCI nebo ENG se nezobrazí. Není-li číslo v rozmezí $\pm 0.000000001 \sim \pm 9999999999$, pak je zobrazeno v exponenciálním tvaru.

Formát zobrazení čísla se mění stiskem kláves [2nd F] [FIX], pokud je výsledek výpočtu na displeji nebo po vymazání všech hodnot v kalkulačce klávesou [CE/C]. Po každém stisknutí [2nd F] [FIX] se formát čísla změní v pořadí FIX→SCI→ENG→Floating point system→FIX→...

4.3.3.Nastavení počtu desetinných míst

Je-li nastaven FIX, SCI nebo ENG formát zobrazení, pak může být počet desetinných míst nastaven na hodnotu 0 až 9. Po nastavení počtu desetinných míst je číslo zaokrouhleno na odpovídající počet číslic. Nastavit počet desetinných míst lze následujícím způsobem. Nejprve je třeba stisknout klávesy [2nd F] [FIX], je-li zobrazen výsledek výpočtu nebo jsou-li hodnoty v kalkulačce vymazány klávesou [CE/C], a potom stisknout odpovídající číselnou klávesu.

4.3.4.Nastavení úhlové jednotky

Kalkulačka umožňuje výpočty v následujících třech úhlových jednotkách.

---

[8] v české literatuře jsem nenalezl ekvivalent, a proto používám téměř doslovný překlad

Stupně: Na displeji se zobrazí nápis DEG.

Radiány: Na displeji se zobrazí nápis RAD.

Grady: Na displeji se zobrazí nápis GRAD.

Nastavit úhlovou míru lze klávesou DRG. Po každém stisku klávesy DRG se jednotky úhlu mění v pořadí DEG → RAD → GRAD → DEG → ...

4.4. Výpočty na kalkulačce

4.4.1. Aritmetické operace

Tab. 5 Aritmetické operace

Příklad	Stisknuté klávesy	Displej	Poznámky
$45+285\div 3=$	45 + 285 ÷ 3 =	140.	
$\frac{18+6}{15-8}=$	C/CE (18 + 6) ÷ (15 − 8 =	3.428571429	Pravá závorka před = , M+, (a dalšími operacemi ukončující výpočet) může být vynechána.
$42\times(-5)+120=$	42 × +/- 5 + 120 =	-90.	
$(5\times 10^{3})\div$ $(4\times 10^{-3})=$	5 Exp 3 ÷ 4 Exp +/- 3 =	1250000.	

4.4.2. Funkce

Před provedením výpočtů je důležité vymazat kalkulačku stisknutím klávesy C/CE, a nastavit úhlovou jednotku. (viz kap. 4.3.4).

Tab. 6 Funkce

Příklad	Stisknuté klávesy	Displej	Poznámky
$\sin 60^{\circ}=$	C/CE sin 60 =	DEG 0.866025403	DEG
$\cos\frac{\pi}{4}[\text{rad}]=$	C/CE cos (2nd F π ÷ 4 =	RAD 0.707106781	RAD
$\arctan 1=[\text{grad}]$	C/CE 2nd F tan 1 =	GRAD 50.	Výsledek v GRAD

Řetězení výpočtů (výsledek posledního výpočtu je použit jako první operand).

Tab. 7 Řetězení výpočtů

Příklad	Stisknuté klávesy	Displej	Poznámky
$25+5=30$ $\sin 30^\circ =$	C/CE 25 + 5 = sin = =	DEG 30. DEG sin 30. DEG 0.5	**DEG** Po stisku klávesy ukončující výpočet je možno vkládat funkce i postfixově.

♦ Výsledky cyklometrických funkcí jsou zobrazeny v následujícím rozsahu (tab. 4):

Tab. 8 Obor hodnot cyklometrických funkcí

	$\theta = \arcsin x$ $\theta = \arctan x$	$\theta = \arccos x$
DEG	$-90 \le \theta \le 90$	$0 \le \theta \le 180$
RAD	$-\frac{\pi}{2} \le \theta \le \frac{\pi}{2}$	$0 \le \theta \le \pi$
GRAD	$-100 \le \theta \le 100$	$0 \le \theta \le 200$

Tab. 9 Aritmetické operace a funkce

Příklad	Stisknuté klávesy	Displej	Poznámky
$(\cosh 1{,}5 + \sinh 1{,}5)^2 =$	C/CE hyp cos 1.5 + hyp sin 1.5 = x^2	20.08553692	
$\operatorname{argtanh}\frac{5}{7} =$	C/CE 2nd F ahyp tan (5 ÷ 7 =	0.895879734	
$\ln 20 =$	C/CE ln 20 =	2.995732274	
$\log 50 =$	C/CE log 50 =	1.698970004	
$e^3 =$	C/CE 2nd F e^x 3 =	20.08553692	
$10^{1.7} =$	C/CE 2nd F 10^x 1.7 =	50.11872336	

Tab. 9 Aritmetické operace a funkce

Příklad	Stisknuté klávesy	Displej	Poznámky
$\frac{1}{6}+\frac{1}{7}=$	C/CE 6 2nd F x^2 + 7 2nd F x^2 =	0.309523809	
$8^{-2}-3^4\times5^2=$	C/CE 8 y^x +/- 2 − 3 y^x 4 × 5 x^2 =	-2024.984375	
$\left(12^3\right)^{\frac{1}{4}}=$	C/CE 12 y^x 3 y^x 4 2nd F x^2 =	6.447419591	
$\sqrt{49}-\sqrt[4]{81}=$	C/CE $\sqrt{x}$ 49 − 4 2nd F $\sqrt[y]{x}$ 81 =	4.	
4! = (4 × 3 × 2 × 1 =)	C/CE 4 2nd F n!	24.	n!= n.(n-1).(n-2) ... 2.1
${}_{10}P_3=$ $nPr=\frac{n!}{(n-r)!}$	C/CE 10 2nd F nPr 3 =	720.	
${}_5C_2=$ $nCr=\frac{n!}{r!(n-r)!}$	C/CE 5 nCr 2 =	10.	
500 × 25% =	500 × 25 % =	125.	
120 ÷ 400 = ?%	120 ÷ 400 % =	30.	
500 + (500 × 25%)	500 + 25 % =	625.	
400 − (400 × 30%)	400 − 30 % =	280.	

4.4.3.Náhodná čísla

Náhodné číslo s třemi platnými číslicemi může být generováno stisknutím kláves 2nd F RND .

4.4.4.Převod úhlových jednotek

Úhlové jednotky je možno měnit v následujícím pořadí: stupně → radiány → grady → stupně → ...

Tab. 10 Převod úhlových jednotek

Příklad	Stisknuté klávesy	Displej	Poznámky
90° → ? [rad] → ? [grad]	90	DEG 90.	DEG
	DRG▸	RAD >R 1.570796327	$\frac{\pi}{2}$ [rad]
	DRG▸	GRAD >G 100.	100[grad]
	DRG▸	DEG >D 90.	90°
$\sin^{-1} 0.8 = ?°$ → ? [rad] → ? [grad]	2nd F sin		
	0.8 =	DEG 53.13010235	°(DEG)
	DRG▸	RAD >R 0.927295218	[RAD]
	DRG▸	GRAD >G 59.03344706	[GRAD]

$180° = \pi$ [rad] = 200 [grad] ($\pi = 3.141592654$)

4.4.5.Počítání s pamětí

Tento typ kalkulátoru má jednu nezávislou paměť (M), jednu paměť, do které se ukládá poslední výsledek (ANS - *last answer memory*), a šest pomocných pamětí (A~D, X, Y).

4.4.5.1.Nezávislá paměť (M)

M paměť je přístupná v normálním módu. Do této paměti lze přímo zapisovat hodnotu, přičítat do ní hodnotu nebo od ní odečítat hodnotu.

STO M

Vymaže nezávislou paměť a uloží do ní zobrazenou hodnotu.

RCL M

Vyvolá číslo z nezávislé paměti. Je-li na displeji číslo, přepíše ho.

M+

Přičte zobrazené číslo k číslu, které je uloženo v paměti M.

2nd F M-

Odečte zobrazené číslo od čísla uloženého v paměti.

- Paměť se vymaže stiskem kláves 0 STO M (tzn. do paměti se uloží 0).

4.4.5.2.Pomocné paměti (A až D, X, Y)

Paměti A ~ D, X a Y jsou dostupné v normálním módu.

STO A ~ D, Y, X

Uloží zobrazené číslo do specifikované pomocné paměti. (A ~ D, X, Y).

RCL A ~ D, Y, X

Vyvolá číslo z pomocné paměti. Je-li na displeji číslo, nepřepíše ho, ale násobí číslem z paměti.

- Pomocné paměti jsou vymazány uložíme-li do nich nulu.

4.4.5.3.Paměť s posledním výsledkem

Výsledek, získaný stiskem klávesy = nebo jiné klávesy ukončující výpočet, je automaticky uložen v ANS paměti.

ANS

Vyvolá obsah ANS paměti.

Tab. 11 Výpočty s využitím paměti

Příklad	Stisknuté klávesy	Displej	Poznámky
$24 \div \underline{(8 \times 2)} =$ $\underline{(8 \times 2)} \times 5 =$	8 × 2 STO M 24 ÷ RCL M = RCL M × 5 =	M =>M 16. M 1.5 M 80.	Nezávislá paměť
$23 + 45 + 78 =$ $+) \; 52 - 31 + 43 =$ $-) \; 64 + 73 - 12 =$	23 + 45 + 78 STO M 52 − 31 + 43 M+ 64 + 73 − 12 2nd F MC RCL M 0 STO M	M =>M 146. M M+ 64. M M- 125. M M 85. =>M 0.	Nezávislá paměť Vymazání nezávislé paměti (zmizí znak M v pravém horním rohu displeje)
1£ = 42,68 Kč 32010 Kč = ? £	42.68 STO A 32010 ÷ RCL A =	=>A 42.68 750.	Pomocná paměť
2700 £ = ? Kč	2700 × RCL A = nebo 2700 RCL A	115236. (*) 115236.	Pomocná paměť Operace násobení může být při vyvolání čísla z pomocné paměti vynechána. Operace násobení může být vynechána také při vyvolání čísla π . (viz další příklad)

Tab. 11 Výpočty s využitím paměti

Příklad	Stisknuté klávesy	Displej	Poznámky
$r = 3$ cm $2\pi r = ?$ $(r \rightarrow A)$	3 STO A 2 2nd F π RCL A =	=>A 3. (*) 6.283185307 (*) 18.84955592 18.84955592	Pomocná paměť Operace násobení může být při vyvolání čísla z pomocné paměti vynechána. Operace násobení může být vynechána také při vyvolání čísla π.
$\frac{24}{4+6} = 2{,}4$ (A) $3 \times 2{,}4(A) +$ $60 \div 2{,}4(A) =$	24 ÷ (4 + 6) = 3 × ANS + 60 ÷ ANS =	2.4 * 2.4 / 2.4 32.2	ANS paměť (poslední výsledek)

Upozornění:

Při výpočtu funkcí $\rightarrow r\theta$, $\rightarrow xy$ jsou výsledky automaticky uloženy do pomocných pamětí X a Y. Proto je třeba dávat pozor při použití těchto pamětí.

4.4.6. Zlomky

$a^{b}/_{c}$

Tato klávesa se používá pro vložení zlomku nebo pro převod zlomku na desetinné číslo.

2nd F $d/_{c}$

Uvedená kombinace kláves se používá k převodu smíšeného čísla na zlomek nepravý.

- Desetinné číslo nebo exponenciální číslo nelze zadat jako zlomek
- Největší možná délka čísla je šest číslic pro čitatele a tři číslice pro jmenovatele.
- V případě smíšených čísel je maximální délka tři číslice pro každou část (celou část, čitatele a jmenovatele).

- Není-li absolutní hodnota zobrazeného čísla v rozmezí $1/999 \sim 998999/999$ nebo výsledné číslo nelze zapsat při dodržení předchozích podmínek (přetečení čitatele nebo jmenovatele), pak kalkulačka zobrazí výsledek jako desetinné číslo.

Tab. 12 Výpočty se zlomky

Příklad	Stisknuté klávesy	Displej	Poznámky
$3\frac{1}{2}+\frac{4}{3}=$	3 a b/c 1 a b/c 2 +		
	4 a b/c 3 =	4/5/6	Smíšené číslo
$4\frac{5}{6}\left(\frac{29}{6}\right)$	a b/c	4.833333333	Desetinné číslo
	2nd F d/c	29/6	Zlomek nepravý
$\sqrt{2\frac{8}{9}}=$	CE √		Výsledek pouze jako desetinné číslo.
	2 a b/c 8 a b/c 9		
	=	1.699673171	

4.4.7. Číselné soustavy

Na kalkulačce lze provádět převody mezi dvojkovou, osmičkovou, desítkovou a šestnáctkovou číselnou soustavou. Kalkulátor dále dokáže provádět čtyři základní aritmetické operace (+, −, ×, ÷) s použitím pamětí a závorek. Navíc, ve dvojkové, osmičkové a šestnáctkové soustavě, umí používat logické operace NOT, AND, OR, XOR a XNOR.

Převod do jiné číselné soustavy se provádí pomocí následujících kláves:

2nd F →BIN

Převod do dvojkové soustavy (*binary system*). Objeví se ->BIN a v pravém horním rohu displeje (místo exponentu) písmeno B, které říká, že počítáme ve dvojkové soustavě.

2nd F →OCT

Převod do osmičkové soustavy (*octal system*). Objeví se ->OCT a písmeno O.

[2nd F] [HEX]

Převod do šestnáctkové soustavy (*hexadecimal system*). Objeví se ->HEX a písmeno H.

[2nd F] [DEC]

Převod do desítkové soustavy (*decimal system*). Objeví se ->DEC. Nezobrazí se žádné písmeno na místě exponentu.

- Čísla A ~ F v šestnáctkové soustavě jsou zadávána klávesami [y^x] - A, [$\sqrt{}$] - B, [x^2] - C, [log] - D, [ln] - E a [n!Cr] - F. Ve dvojkové, osmičkové a šestnáctkové soustavě nelze zadávat desetinná čísla. Je-li převáděno desetinné číslo z desítkové soustavy, pak je jeho desetinná část ignorována. Stejně tak, je-li výsledkem některé operace (v úvahu připadá dělení) desetinné číslo, pak i v tomto případě je desetinná část ignorována (nezaokrouhluje se). Záporná čísla ve dvojkové, osmičkové a šestnáctkové soustavě jsou zobrazena jako doplněk (viz příklad dále).

$$-1_{dec} = 1111111111_{bin} = 7777777777_{oct} = FFFFFFFFFF_{hex}$$
$$-2_{dec} = 1111111110_{bin} = 7777777776_{oct} = FFFFFFFFFE_{hex}$$
$$\vdots$$
$$-512_{dec} = 1000000000_{bin} = 7777777000_{oct} = FFFFFFFE00_{hex}$$
$$\vdots$$

Tab. 13 Operace s číselnými soustavami

Příklad	Stisknuté klávesy	Displej	Poznámky
$25_{dec} \xrightarrow{BIN}$	[C/CE] [2nd F] [DEC]		DEC
	25 [2nd F] [BIN]	=>BIN 11001.^{B}	BIN
$1AC_{HEX} \xrightarrow{BIN}$	[C/CE] [2nd F] [HEX]	1AC.^{H}	HEX
$\xrightarrow{OCT}$	1AC		
$\xrightarrow{DEC}$	[2nd F] [BIN]	=>BIN 110101100.^{B}	BIN
	[2nd F] [OCT]	=>OCT 654.^{O}	OCT
	[2nd F] [DEC]	=>DEC 428.	DEC

Tab. 13 Operace s číselnými soustavami

Příklad	Stisknuté klávesy	Displej	Poznámky
BIN (1010 – 100) ×11 =	C/CE 2nd F BIN (1010 − 100) × 11 =	 110.B 10010.B	BIN
-111_{BIN} =	C/CE 2nd F BIN NEG 111 =	NEG −B 1111111001.B	NEG (-7)
$1FF_{HEX}$ + 512_{OCT} = HEX →	C/CE 2nd F HEX 1FF 2nd F OCT + 512 = 2nd F DEC	 1511.O =>HEX 349.H	HEX OCT HEX
2FEC – 2C9E = +)2000 – 1901 =	C/CE 0 STO M 2nd F HEX 2FEC − 2C9E M+ 2000 − 1901 M+ RCL M	 M M+ 34E.H M M+ 6FF.H M M 4AD.H	Vymazání paměti HEX Využití paměti
1011 AND 101=	C/CE 2nd F BIN 1011 AND 101 =	 1.B	BIN
5A OR C3 =	C/CE 2nd F HEX 5A OR C3 =	 DB.H	HEX
NOT 10110 =	C/CE 2nd F BIN NOT 10110 =	 1111101001.B	BIN
24 XOR 4 =	C/CE 2nd F OCT 24 XOR 4 =	 20.O	OCT
B3 XNOR 2D =	C/CE 2nd F HEX B3 XNOR 2D =	 FFFFFFF61.H	HEX

4.4.8.Čas, minuty a vteřiny

Na kalkulačce lze převádět desetinnou část čísla na minuty a vteřiny a naopak. Dále lze provádět čtyři základní aritmetické operace (rozumějme sčítání, odčítání, násobení a dělení) s čísly zadanými jako *celá část - minuty - vteřiny*.

Tab. 14 Čas, minuty a vteřiny

Příklad	Stisknuté klávesy	Displej	Poznámky
12°39'18"05 DEG →	12 [DMS] 39 [DMS] 18 [DMS] 5 [2nd F] [DMS]	12.65501389	12.6550138°
3h30m45s + 6h45m36s =	3 [DMS] 30 [DMS] 45 [+] 6 [DMS] 45 [DMS] 36 [=]	10°16'21.00	10h16m21s
123.678 → Des Min Sec	123.678 [DMS]	123°40'40.80	123°40'40"80
3h45m - 1.69h =	3 [DMS] 45 [−] 1.69 [=] [DMS]	2°03'36.00	2h3m36s
sin 62°12'24" =	[CCE] [sin] 62 [DMS] 12 [DMS] 24 [=]	DEG 0.884635235	DEG

4.4.9.Převod souřadnic

V normálním módu je možno provádět převody z kartézských souřadnic do polárních a naopak.

- Před prováděním výpočtu je důležité zvolit správnou úhlovou jednotku. (viz kap. 4.3.4)

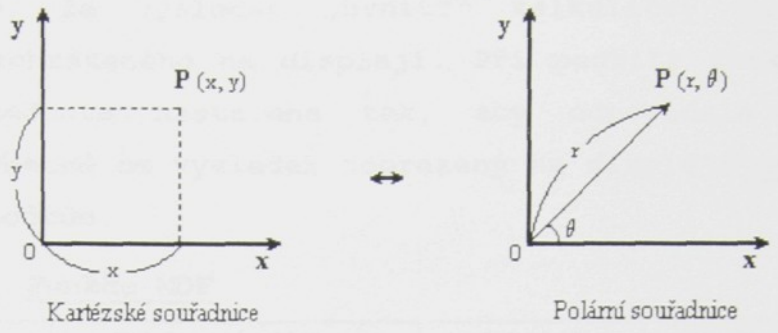
[→rθ]

převádí z kartézských souřadnic (x, y) do polárních (r, θ)

[2nd F] [→x]

převádí z polárních souřadnic (r, θ) do kartézských (x, y).

Obr. 3 Převod souřadnic



2nd F [ ]

Používá se pro zadání hodnoty y u kartézských souřadnic resp. hodnoty θ u polárních souřadnic.

[ ]

Přepíná mezi hodnotami x a y resp. r a θ.

♦ Výsledek výpočtu je automaticky uložen do pamětí X a Y. Hodnoty x a r jsou uloženy v paměti X a hodnoty y a θ jsou uloženy v paměti Y.

Tab. 15 Převod souřadnic

Příklad	Stisknuté klávesy	Displej	Poznámky
$(x = 6, y = 4) \rightarrow$ $(r = ?, \theta = ?)$	6 2nd F [ ] 4	4.	DEG
	→rθ	r 7.211102551	
	[ ]	θ 33.69006753	
	2nd F [ ]	x 6.	
	[ ]	y 4.	
$\left(r = 14, \theta = \frac{\pi}{5}\right)$ $\rightarrow (x = ?, y = ?)$	14 2nd F [ ] 2nd F [ ] ÷ 5 2nd F [ ] [ ]	x 11.32623792	RAD
		y 8.228993532	

4.4.10. Funkce MDF[9]

V kalkulačce probíhají všechny výpočty s přesností na 16 platných číslic pro mantisu. Avšak na displeji je

[9] MDF z anglického *modify* (modifikovat, upravit).

zobrazeno maximálně[10] 10 platných číslic pro mantisu, a proto je zřejmé, že výsledek „uvnitř" kalkulačky se liší od výsledku zobrazeného na displeji. Při použití funkce MDF je vnitřní hodnota nastavena tak, aby odpovídala zobrazené hodnotě. Obecně se výsledek zobrazený na displeji nepoužívá k dalším výpočtům.

Tab. 16 Funkce MDF

Příklad	Stisknuté klávesy	Displej	Poznámky
$5 \div 9 = (Z)$	5 ÷ 9 =	0.6	FIX, TAB = 1
$(Z) \times 9 =$	× 9 =	5.0	0.5555555555555555 × 9
	5 ÷ 9 =		příklad MDF FIX, TAB = 1
	2nd F MDF	0.6	
	× 9 =	5.4	0.6 × 9

[10] Závisí na formátu zobrazení čísla a na nastaveném počtu des. míst (viz kap. 4.3.2, 4.3.3).

4.5. Statistické výpočty

Statistické výpočty jsou prováděny ve statistickém módu. Do statistického módu se přepínáme stiskem klávesy [Mode].

4.5.1. Statistické funkce

Kalkulačka dokáže vypočítat následující statistické funkce:

Aritmetický průměr [RCL] [$\bar{x}$]

$$\bar{x} = \frac{\sum x}{n}$$

Směrodatná odchylka [RCL] [sx]

$$sx = \sqrt{\frac{\sum x^2 - n\bar{x}^2}{n-1}}$$

Výběrová směrodatná odchylka [RCL] [σx]

$$\sigma x = \sqrt{\frac{\sum x^2 - n\bar{x}^2}{n}}$$

Součet (suma) prvků [RCL] [Σx]

$$\sum x = x_1 + x_2 + \ldots + x_n$$

Součet druhých mocnin prvků [RCL] [Σx^2]

$$\sum x^2 = x_1^2 + x_2^2 + \ldots + x_n^2$$

n - počet prvků [RCL] [n]

4.5.2. Zadání dat

Zadaná data jsou v paměti dokud nejsou vymazána stiskem kláves [2nd F] [CA]. Před zadáváním nových dat je nutné vymazat obsah paměti.

Data se zadávají následujícími dvěma způsoby:

Hodnota [DATA].

Hodnota [×] počet prvků dané hodnoty [DATA] (zadání určitého počtu prvků stejné hodnoty).

4.5.3.Oprava dat

Oprava před stiskem klávesy DATA:

Vymazat nesprávnou hodnotu lze klávesou CE resp. klávesou →.

Oprava po stisku klávesy DATA:

Jestliže byla stisknuta klávesa DATA, ale pak už žádná jiná klávesa, můžeme vymazat poslední zadaná data klávesou 2nd F CE. V případě, že po stisku klávesy DATA byly prováděny další operace, je třeba znovu zadat data, která chceme opravit (vymazat), a pak stisknout kombinaci kláves 2nd F CE.

Tab. 17 Statistické příklady

Příklad		Stisknuté klávesy	Displej		Poznámky
Věk	Počet žáků			STAT	STAT
		6 DATA			n
6	1				
7	2	7 × 2 DATA			
8	7	8 × 7 DATA			
9	9				
10	23	9 × 9 DATA	n	1.	
11	56		n	3.	
12	70	10 × 23 DATA	n	10.	
13	21		n	19.	
		11 × 56 DATA	n	42.	
			n	98.	
		12 × 70 DATA	n	168.	
			n	189.	
		13 × 21 DATA			
		RCL Σx			
			Σx	2116.	
		RCL Σx²	Σx2	24016.	
		RCL n	n	189.	Σx
		RCL x̄	ϕx	11.1957672	Σx²
			sx	1.316338841	n
		RCL sx	σx	1.312851845	$\bar{x}$
		RCL σx			sx
					σx

4.6. Chyby a rozsah kalkulačky

4.6.1. Chyby

Při výpočtech dochází k chybě pokud operace přesáhne rozsah kalkulačky (přetečení) nebo při nepovolené matematické operaci (dělení nulou, druhá odmocnina ze záporného čísla, ...). V případě, že dojde k chybě, nelze pokračovat ve výpočtech, dokud není chyba vymazána stiskem klávesy  .

- Číslo následující za nápisem „Error" udává typ chyby. Pomocí tabulky 13 je možno zjistit co znamená tento typ chyby.

Tab. 18 Chybová hlášení na kalkulátoru

1	♦ Nepovolená operace (n! - n musí být celé číslo) Syntaktická chyba (pokus o převod souřadnic, i když jsme zadali pouze jednu souřadnici : 2 )
2	♦ Absolutní hodnota mezivýsledku nebo konečného výsledku početní operace je větší nebo rovna 10^{100}.
3	♦ Přetečení paměti počítače (nedostatek paměti). K této chybě by prakticky vůbec nemělo dojít (pokud aplikace běží samostatně).
4	♦ Dělení nulou.

4.6.2. Rozsah kalkulačky.

Vstupy a výstupy: $\pm 10^{-99} \sim \pm 9.999999999 \times 10^{99}$ a 0.

Mezivýsledky a výsledky: $\pm 10^{-99} \sim \pm 9.999999999 \times 10^{99}$ a 0.

- Je-li absolutní hodnota vstupu, mezivýsledku nebo výsledku menší než 10^{-99}, pak je toto číslo pokládáno za nulu, jak na displeji, tak i uvnitř kalkulačky.

Tab. 19 Rozsah kalkulačky při výpočtu funkcí

Funkce	Rozsah argumentu
$\sin x$ $\cos x$ $\tan x$	DEG: $\lvert x \rvert < 4.5 \times 10^{10}$ $(\tan x: \lvert x \rvert \neq 90(2n-1) \mid n \in Z)$ RAD: $\lvert x \rvert < \frac{\pi}{40} \times 10^{10}$ $(\tan x: \lvert x \rvert \neq \frac{\pi}{2}(2n-1) \mid n \in Z)$ GRAD: $\lvert x \rvert < 5 \times 10^{10}$ $(\tan x: \lvert x \rvert \neq 100(2n-1) \mid n \in Z)$

Tab. 19 Rozsah kalkulačky při výpočtu funkcí

Funkce	Rozsah argumentu
asin x acos x	$\lvert x \rvert \leq 1$
ln x log x	$10^{-99} \leq x \leq 9.999999999 \times 10^{99}$
e^x	$-10^{100} < x \leq 230.2585092$
10^x	$-10^{100} < x < 100$
x^2	$\lvert x \rvert < 10^{50}$
acosh x	$1 \leq x < 9.999999999 \times 10^{99}$
atanh x	$\lvert x \rvert < 1$
$\sqrt{x}$	$0 \leq x \leq 9.999999999 \times 10^{99}$
x^{-1}	$0 < \lvert x \rvert \leq 9.999999999 \times 10^{99}$
$n!$	$0 \leq n \leq 69 \quad n \in Z$
nCr nPr	$0 \leq r \leq n \quad r, n \in Z$
→ DEG D°M'S	$0 \leq \lvert x \rvert \leq 9999.999998$
$r, \theta \rightarrow x, y$	$0 \leq r \leq 9.999999999 \times 10^{99}$ DEG: $\lvert \theta \rvert < 4.5 \times 10^{10}$ RAD: $\lvert \theta \rvert < \frac{\pi}{40} \times 10^{10}$ GRAD: $\lvert \theta \rvert < 5 \times 10^{10}$
→ BIN → OCT → HEX	BIN: $-512 \leq x \leq 511$ OCT: $-536870912 \leq x \leq 536870911$ HEX: $-549755813888 \leq x \leq 549755813887$

- Maximálně 255 závorek smí být vnořeno do sebe. Dalším omezením je paměť počítače.

5. Závěr

V souladu se zadáním diplomové práce byl program vypracován pro prostředí operačního systému MS-DOS. Jak jsem se již v práci zmínil, ještě jsem se s podobným programem v OS MS-DOS nesetkal.

Dodaný program kalkulačka byl vytvořen podle kalkulátoru SHARP EL-520G a může tuto kalkulačku zcela nahradit. Logické systémy programu a kalkulátoru se téměř shodují. Design a rozložení kláves odpovídá standardu kalkulátorů SHARP. V programu byla odstraněna chyba, které se kalkulátor SHARP dopouští při přepisu operací.

Ve školách může program posloužit například pro srovnání logických systémů (srovnání programu kalkulačka s kalkulačkou používanou na škole). Dále je vhodné program použít v souvislosti s výukou ovládání počítače.

Seznam pramenů

[AČS] Hais K., Hodek B.: Velký anglicko-český slovník I-IV, 2. vydání, Praha, Academia 1991-1993

[CCC] Kalivodová M. : C - jazyk, obecná verze jazyka. 1. vydání. Slušovice, JZD Slušovice 1987

[ČAS] Poldauf I.: Česko anglický slovník, 2. vydání, Praha, SPN 1990

[ČUA] Kulič V. : Člověk - učení - automat. 2. vydání, Praha, SPN 1989

[DIP] Vild J.: Metodické pokyny k diplomovým pracím na pedagogické fakultě. 2. vydání, Liberec 1994

[HSK] Mrázek J.: Hry s kalkulátory. 2. vydání, Praha, SPN 1986

[KKŠ] Ježek J.: Kapesní kalkulátory na škole. 1. vydání, Praha, SPN 1986

[KOM] Renner G. : Borland C++ Kompendium. 1. vydání, Brno, UNIS 1992

[OBJ] Nenadál K. - Václavíková,D. : Borland C++ objektové programování a popis jazyka. 1. vydání, Praha, Grada 1992

[POL] Polák J.: Přehled středoškolské matematiky. Praha, SPN 1983

[MAT] Učebnice matematiky pro 5. až 8. ročník základní školy

[MAN] SHARP® Scientific calculator Model EL-520G - *Operating manual*

Příloha 1

Výpisy souboru DIPL.CPP a hlavičkových souborů

```
// Soubor DIPL.CPP
/*
Součást modulu kalkulačka
 (c) Jiří Šikner 1995
hlavní soubor
*/

#include "calc.h"
#include "mouse.h"
#include "paleta.h"
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <iostream.h>
#include <alloc.h>

int main()
{

 // graphics init
// int gdriver=DETECT,gmode,errorcode;
 int gdriver = VGA, gmode = VGAHI;
 initgraph(&gdriver, &gmode, "");
//  grerror("Inicializace grafického režimu");
//  atexit(closegraph);
 int errorcode = graphresult();
 if (errorcode != grOk)
 {
   printf("Graphics error: %s\n", grapherrormsg(errorcode));
   printf("Press any key to halt:");
   getch();
   return 1;
 }

   cleardevice();
   setbkcolor(Black);

// Mouse Init
  if (!MouseInit())
  {
      cout << "Není nahrán ovladač myši - kalkulačku můžeš ovládat jen z
klávesnice";
    getch();
  }
  TCalc Calc(100, 25);  // inicializace kalkulačky - volám constructor TCalc
  // Bude-li to nutné je možno zde kalkulačku pouze inicializovat
  // a pak spustit odkudkoliv příkazem Calc.Run();
  // Nyní je Run přímo v consructoru kalkulačky Oh yeah!
  closegraph();
  return 0;
} // program ends
```

```
// Soubor CALC.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace objektů TButton a TCalc
*/


#if !defined (_CALC)
#define _CALC

#include "types.h"
#include "calcstr.h"
#include "consts.h"

struct TButton
{
        int Left, Top, Right, Bottom; // souřadnice čudlíku
        int Key; // číslo čudlíku - pro práci s klávesnicí
        byte ShadowPalette; // barvy čudlíku: gray, pink ...
};

class TCalc {

public:
int Col;
// variables
    int X, Y;       // pozice okna
// methods
    TCalc(int PosX, int PosY); // Constructor
    ~TCalc(void); // Destructor
    void Run(void);

private:
// calc status
  char far * DispOp; // operation on display
  boolean CS_2ndF;
  enum {DEC, HEX, OCT, BIN} CS_NumSys; // číselné soustavy
  enum {FLOAT, FIX, SCI, ENG} CS_Display; // Display notation
  enum {REAL, FRACTION, IMPROPER} CS_FracDisp; // zobrazení zlomku
  boolean CS_Key; // pro klávesy typu F1, F2 ...
  boolean CS_Exp; // Exponent
  boolean CS_Num; // naposledy stisknuto číslo 0 .. F nebo .
  boolean Comma; // zda byla stisknuta des. čárka či ne
  boolean CS_Rcl;
  boolean CS_Sto;
  boolean CS_Hyp;
  boolean CS_Stat;
  boolean CS_Tab;
  boolean CS_Data;
  byte Tab;
  byte CS_Frac;
// variables
  TCalcString DNum, DExp; // mantisa, exponent
  TCalcString DMin, DSec, DSSec; // minuty, sekundy, setiny sekundy
  TCalcString StrNum;     // celé číslo
  TButton  Button[BTNCOUNT];
  byte DefBtn;
  void far * Bitmap; // slouží metodám Press a Release
```

```
  // vstupy
  // výpočty
  double Number;
// Ovládání kalkulačky
        void InitDefault(void);
        void Paint(void);  // nakreslí kalkulačku
        void CalcEvent(void);
        void CalcHelp(void);
        void PaintDefBtn(byte, boolean); // zobrazí čudl jako default/normal
        void Press(byte B);   // překreslí čudl na stisk
        void Release(byte B); // překreslí čudl do původní polohy
// početní funkce a procedury + display
        void CalcDisplay(void);
        void BtnPressed(byte Btn);
        void PressedNum(const char Num);
        void GetNumber(void); // převod DNum, DExp na číslo
        void SetDisplay(void); // převod Number na display
        void GetDisplay(void); // čte číslo je-li CS_Num==TRUE
};
#endif
```

```
// Soubor CALCMATH.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace objektů PRegister a TTask
*/

#if !defined (_CALCMATH)
#define _CALCMATH

#include "types.h"
#include "consts.h"

class PRegister {
  friend TTask;
  friend TCalc;
  byte Function;   // funkce na operandu
  double Number;   //  je-li zde levá závorka pak Number=PAR
  byte Operator;  // operand, není li dán explicitně je to operace
                  // násobení
  PRegister * Next; // ukazatel na další register
};

class TTask {
  friend TCalc;
  PRegister * Root;     // to abych neztratil seznam
  PRegister * Register;
  PRegister * Reg2; // to abysom mohol počítať

  char far * DispMem; // memory on display
  enum {DEG, RAD, GRAD} CS_Ang; // Angular unit
  enum {DEC, MIN, SEC, SSEC} CS_DMS; // DMS
  enum {DISABLED, INPUT, RESULT} CS_CC; // coordinate conversions
  enum {DISABLE, ENABLED, CALCUL} CS_CF; // calculate with fractions
   double Memory, AMem, BMem, CMem, DMem, XMem, YMem, AnsMem; //
paměti
    double DataN, DataSumX, DataSumX2, DataValue, DataCount; // datové
paměti
  byte ParCount; // počet závorek
  byte TopBit; // pro logické funkce
  boolean CS_Equal; // příklad ukončen (=, M+, M-, ->OCT ...)
  boolean Init(void);
  void Done(void); // opět si z toho uděláme Pascal Oh yeah
  void Clear(void); // smaže všechny registry
  void ShowTask(void); // vypíše všechny registry
  void Add(void); // přidání prázdného registru
  void Del(void); // vymazání registru
  void DelReg(PRegister*); // vymazání daného registru
  double CalcInput(byte); // spočítej se resp. great chaos
  void CalcSelf(void);
  void CalcPriority(void);
  void CalcFnc(byte);
  void CalcOp(byte);
  void CheckRange(double &);
  boolean PrepGon(double &);
  void Rad2Ang(double &);
  void InsertNum(double); // vloží číslo do registru
  double Convert2xy(void);
  double Convert2rf(void);
```

```
  double MemRecall(double, boolean, const char far *);
        // vloží číslo z paměti do registru
  double MemAdd(double); // přičte číslo do nezávislé paměti
  double DegRadGrad(void);
// data methods
  void ClearData(void);
  void AddData(void);
  void GetMeanOfSamples(void);
  void GetSampleDeviation(void);
  void GetPopulationDeviation(void);
};

#endif
```

```
// Soubor CALCSTR.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace objektu TCalcString pro práci s řetězci
*/

#if !defined (_CALCSTR)
#define _CALCSTR
#include "types.h"

class TCalcString {
 friend TCalc; // only TCalc can use this class Oh yeah
   char far * PStr; // ukazatel na obsah řetězce
   byte Length, MaxLen; // aktuální a maximální délka řetězce
   boolean Rotate; // chová se na vstupu jinak
   void Init(byte, boolean); // inicializace
   void Done(void); // smazání
   void Put(const char far *Co); // vloží řetězec na tvrdo
   void Put(const TCalcString Co); // kopie TCalcString na tvrdo
   void Neg(void); // +/-
   boolean Empty(void);
   boolean Zero(void);

   void AddChar(const char far);
   TCalcString& operator+=(const char far); // přidá znak
   void AddString(const char far *);
   TCalcString& operator+=(const char far *); // přidá řetězec
   TCalcString& operator--(void);          // umaže, jestli má co
   void PutSum(TCalcString, TCalcString);
     // vloží součet mantisa + exponent na tvrdo
   void PutSum(const char far *, const char far *);
     // vloží součet 2 řetězců na tvrdo
   void GetStrings(TCalcString &, TCalcString &);
     // rozloží na mantisu a exponent , True - je-li užit exponent
   void ClearZeros(void);
   void MoveComma(int);

   byte FracPart(byte);
   boolean FracLine(void);
     // vrací true je-li na konci řetězce znak FRACLINE
   double GetFrac(void);
     // převede řetězec ve tvaru zlomku na double
   boolean SetFrac(double, boolean);
     // pokusí se převést double na řetězec ve tvaru zlomku, když se to
     // nepovede vrací FALSE
     // boolean parametr udává zda jde o smíšené číslo nebo nevlastní
     // zlomek - TRUE je nevlastní zlomek
};
#endif
```

```
// Soubor CONSTS.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace konstant a proměnné CS_Error
*/

#if !defined _consts
#define _consts

#include "types.h"
#include <math.h>

// errors
#define ILLEGAL 1     // nepovolené číslo na vstupu (argument)
#define CALCRANGE 2         // out of calculator range
#define HEAP 3        // not enough heap space
#define DIVZERO 4     // divide by zero
#define PARAM 5       // parameter overflow
#define STRING 6      // input string error

// CalcMath ---------------------------------------- CalcMath
// Number
const double UNDEF=10e+201; // nedefinované číslo v registru
const double MAXCALC=9.999999999e+99;
const double MINCALC=1e-99;
const double MAXEXP=232;
const double MAXGON=M_PI_4*1e9;
// Priority
#define BINFNC 20     // binární funkce
#define MULDIV 151    // *, /
#define PLUSMINUS 154       // +, -
#define LOGAND 155  // AND
#define LOGFNC 159  // OR, XOR, XNOR
#define PAR 190       // pravá závorka
#define EQUAL 200     // =
// Operator
#define O_YX  20
#define O_XSQR 21
#define O_nCr  22
#define O_nPr  23
#define O_MUL  150    // násobení
#define O_DIV  151    // dělení
#define O_PLUS 153    // sčítání
#define O_MIN  154    // odčítání
#define O_AND  155    // AND
#define O_OR   157    // OR
#define O_XOR  158     // XOR
#define O_XNOR 159     // XNOR
#define L_PAR  191    // levá závorka
#define EQUAL  200    // =
// functions
#define F_SIN         40
#define F_COS         41
#define F_TAN42
#define F_ASIN        45
#define F_ACOS  46
#define F_ATAN  47
#define F_SINH  50
```

```
#define F_COSH  51
#define F_TANH  52
#define F_ASINH 55
#define F_ACOSH 56
#define F_ATANH 57
#define F_LN    60
#define F_LOG   61
#define F_EXP   62
#define F_10X   63
#define F_SQR   65
#define F_SQR3  66
#define F_NOT   70
#define F_NEG   71

// Calc ---------------------------------------------------- Calc
// const
#define GRAYPAL 1
#define PINKPAL 2
#define YELLOWPAL 3
#define BTNCOUNT 44

#define FRACLINE '/'

// to be modified later
#define DispSize 10

// messages
#define MOUSE 1000
#define MOUSE_DOWN 1001
#define EXIT  1002


extern byte CS_Error;

void Beep(int F);

#endif
```

```
// Soubor J_MATH.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace matematických funkcí, které neumí knihovna math.h
*/

#if !defined _J_MATH
#define _J_MATH
#include "types.h"
#include "consts.h"

const MAXSTRLEN=50;
/*
 převod čísla z desítkové a do desítkové číselné soustavy
 parametry 1), 2)
 1) řetězec, který převádím na číslo
 2) číslo - udává horní bit, abych poznal jde-li o kladné nebo záporné
    číslo; př. X=3 (111) - záporné, X=4 (111) - kladné
            - max. povolená hodnota je MAXSTRLEN, jinak dojde k chybě
 funkce při chybě nastaví CS_Error a vrací UNDEF
*/
double Hex2Real (const char far *, byte);
double Oct2Real (const char far *, byte);
double Bin2Real (const char far *, byte);

/*
 funkce vrací číslo typu double
 parametry 1), 2)
 1) číslo, které převádím na řetězec
 2) číslo - udává horní bit
            - max. povolená hodnota je MAXSTRLEN, jinak dojde k chybě
 funkce vrací řetězec, který chrakterizuje číslo v
 HEX resp. OCT resp. BIN soustavě
 při chybě vrací NULL
*/
char far * Real2Hex(double, byte);
char far * Real2Oct(double, byte);
char far * Real2Bin(double, byte);

// další matematické funkce
// funkce při chybě nastaví CS_Error

// n!  číslo n, max. povolené číslo n explicitně - pro kalkulačku 69
// implicitní maximum je 169
double Faktorial(double, byte);

double XnaY(double, double);
double NnadK(double, double);
double NvariaceK(double, double);

// logické funkce
// byte udává horní bit
double LogAND(double, double, byte);
double LogOR(double, double, byte);
double LogXOR(double, double, byte);
double LogXNOR(double, double, byte);

#endif
```

```
// Soubor TYPES.H
/*
 Součást modulu kalkulačka
 (c) Jiří Šikner 1995
 deklarace typů
*/

#if !defined(__TYPES_H)
#define __TYPES_H
enum boolean {FALSE, TRUE};
#endif
typedef unsigned char byte;
typedef unsigned int  word;
typedef unsigned long dword;
```

Příloha 2

Ovládání kalkulačky pomocí klávesnice

Esc	o	F3	F4	Back Space	Delete
h	F5	F6	F7	k	%
a	b	c	d	e	f
p	\	'	r	s	m
7	8	9	(	)	
4	5	6	*	/	
1	2	3	+	−	
0	.	n	End	=, Enter	