Amatérské RADIO

ZÁKLADY PROGRAMOVÁNÍ NA KALKULÁTORU TI 58/59

Jiří Ježek

Obsah:

1. Úvod

Podle vybavení a podle výpočetních možností můžeme současné minikalkulátory rozdělit do tří skupin:

a) přístroje schopné vykonávat základní aritmetické operace (+, −, x, :). Bývají vybaveny datovou pamětí, tj. registrem, do kterého se dá uložit číslo zobrazené na displeji a ze kterého se může číslo přesunout zpět na displej. Samostatným tlačítkem též přičítáme či odčítáme číslo z displeje k obsahu paměti (tzv. aritmetika v paměti). Některé typy obsahují i hodinky, stopky a kalendář nebo zařízení k měření délek.

b) Přístroje, které jsou navíc vybaveny pevně naprogramovanými funkcemi nebo vzorci. Bývají to funkce sin x, cos x, tg x, $\sqrt{x}$, 1/x, ln x, Int x a funkce k nim inverzní. Např. při stisku tlačítka sin proběhne výpočet sinu úhlu, který byl předtím vložen na displej, a po skončení výpočtu se výsledek objeví na displeji (výpočet probíhá dosazením do mnohočlenu aproximujícího příslušnou funkci). Nejednotnost panuje ve značení tlačítek pro výpočet inverzní funkce. Někdy se používá samostatné tlačítko ARC nebo INV, které se kombinuje s tlačítkem příslušné funkce, tedy ARC sin, ARC cos nebo INV tg, INV ln (což je e^x). Často se setkáme se značením $\sin_{-1}$ a s nevhodným $\sin^{-1}$ (nezaměňovat za mocninu). Tyto kalkulačky plně nahradí používané matematické tabulky a jsou již dnes běžné. Lepší typy mají více datových pamětí, bývají vybaveny programy pro statistické výpočty, převody pravoúhlých souřadnic na polární, řešení kvadratických rovnic, kombinatorické výpočty ap. Aritmetika v pamětech umožňuje i násobení a dělení.

c) K současným špičkovým výrobkům patří programovatelné minikalkulátory, které umožňují navíc vložit do zvláštní programové paměti (programových registrů) soubor instrukcí, které pak počítač v určité posloupnosti provádí. Tento program můžeme vkládat buď ručně pomocí příslušných tlačítek nebo jej prostřednictvím vestavěného magnetofonu přehrajeme z hotové magnetické karty. Do nejnovějších typů vkládáme polovodičový modul, ve kterém je již zakódována řada programů o délce až několik set kroků.

Dále se budeme zabývat programovatelnými kalkulátory Texas Instruments **TI-58 a TI-59,** *které jsou v ČSSR nejvíce rozšířeny.* Tyto přístroje mají (údaje v závorkách platí pro typ 59)

a) algebraický operační systém,
b) 60 (100) datových registrů,
c) 480 (960) programových registrů,
d) výměnný polovodičový modul s 25 programy,
e) možnost připojení tiskárny,
f) magnetofon pro nahrání a přehrání programů z magnetického štítku (pouze TI-59),
g) řadu funkcí, se kterými se postupně seznámíme.

Za každou kapitolou tohoto kursu je uvedena řada příkladů, které si podrobně zpracujte a zkontrolujte podle výsledků. Snažte se vytvářet vlastní varianty uvedených příkladů.

2. Algebraický operační systém (AOS)

Algebraický operační systém respektuje pravidla o nadřazenosti aritmetických operací. Pro srovnání sledujeme výpočet na běžné jednoduché kalkulačce při posloupnosti tlačítek

$$5 + 3 \times 4 - 8 : 2 = .$$

Při stisknutí operačního tlačítka (tj. tlačítka k provedení aritmetické operace) uzavře se vždy předešlý naznačený početní výkon, takže počítáme hodnotu výrazu

$$[(5 + 3) \cdot 4 - 8] : 2 = 12 .$$

Při stejném sledu tlačítek na kalkulátoru s AOS bude mít součin přednost před součtem a podíl před rozdílem, takže vypočítáme

$$5 + 3 \cdot 4 - \frac{8}{2} = 13$$

Slučování součinů a podílů zde probíhá přímo, bez nutnosti použití závorek.

Všechny operace můžeme roztřídit podle jejich priority (nadřazenosti) do šesti tříd:

1. Nejprve proběhnou aritmetické operace na paměťových registrech. Jsou to:
   a) **SUM** mn ... číslo na displeji se přičte k obsahu registru číslo mn,
   b) **Prd** mn ... číslem na displeji se vynásobí obsah registru číslo mn,
   c) **INV SUM** mn ... číslo na displeji se odečte od obsahu registru číslo mn,
   d) **INV Prd** mn ... číslem na displeji se dělí obsah registru číslo mn.
   Číslo na displeji zůstává beze změny, započaté aritmetické operace nejsou ovlivněny.
2. Dále mají přednost výpočty hodnoty funkce jedné proměnné. Je-li na displeji hodnota argumentu x, pak se po stisknutí funkčního tlačítka objeví na displeji příslušná funkční hodnota. Započaté aritmetické operace nejsou opět narušeny. Na TI-58/59 máme k dispozici funkční tlačítka **sin, cos, tan, lnx, log, x^2, |x|, $\sqrt{x}$, Int** (tato operace oddělí celou část z čísla na displeji, např. Int 5,65 = 5), kombinací tlačítek můžeme vytvořit funkce další jako **INV sin, INV cos, INV tan, INV ln x** (tj. e^x), **INV log** (tj. 10^x), **INV Int** (INV Int 5,65 = 0,65) a **1/x**.
3. Výpočet funkcí dvou proměnných y^x a **INV** y^x (což je $y^{1/x}$) proběhne tak, že nejprve vložíme na displej hodnotu základu y a po stisku tlačítka y^x, popř. INV y^x vložíme exponent x. Výpočet ukončí každé operační tlačítko s nižší prioritou. Máme-li např. vypočítat $5^6 - 7$, bude pořadí tlačítek 5 y^x 6 − 7 = .
4. Násobení a dělení proběhne vždy před
5. sčítáním a odčítáním.
6. Nejnižší prioritu má operace vyvolaná tlačítkem =, která vyhodnotí všechny započaté operace vzhledem k jejich nadřazenosti a uzavře výpočet.

Pořadí operací stanovené výše uvedeným rozdělením lze podle potřeby měnit vložením levých a pravých závorek na příslušné místo. Můžeme použít až 8 párů závorek a počítat tak velmi složité výrazy. Chceme-li po otevření levé závorky použít jako první člen v závorce číslo z displeje, stiskneme tlačítko CE. Přesuneme tak toto číslo do příslušného operačního registru (CE v tomto případě nenuluje displej). Při výpočtu výrazu 5,678 + (5,678 − 2,3). 5 postupujeme tedy takto:

5 , 6 7 8 + (CE − 2 . 3) × 5 =

Příklad 2.1

Machovo číslo udává poměr rychlosti letadla k rychlosti zvuku. Protože rychlost zvuku závisí na tlaku a teplotě vzduchu, uvádí se pro letadlo letící rychlostí v [km/h] ve výšce h [m] komplikovaný vztah

$$M = \sqrt{5\left[\left(\left\{\frac{\left[\left(1 + 0{,}2 \cdot \left[\frac{v}{1225}\right]^2\right)^{3{,}5} - 1\right]}{\left[1 - 2{,}256 \cdot 10^{-5} \cdot h\right]^{-5{,}2656}}\right\} + 1\right)^{0{,}286} - 1\right]},$$

kde M je Machovo číslo. Vypočítejte M pro rychlost v = 650 km/h a výšku h = 7600 m.

Řešení

Vzhledem k AOS a dostatečnému počtu závorek vkládáme výraz pod odmocninou tak, jak je napsán, tedy

5 × (((((1 + . 2 x (650 : 1225) x^2) y^x 3 . 5 − 1) x (1 − 2.256 EE 5 +/− x 7600) y^x 5.2656 +/−) + 1) y^x . 286 − 1) = $\sqrt{x}$

Na displeji čteme výsledek M = 0,83.

*Při výpočtu M pro jiné hodnoty v a h musíme opakovaně stisknout většinu tlačítek. Při použití programovatelného kalkulátoru se celá posloupnost instrukcí daná tlačítky uloží do programové paměti a při výpočtu vkládáme pouze hodnoty **v** a **h**. Tím se celý výpočet zjednoduší a zkrátí. K příkladu se ještě v dalším vrátíme.*

Poznámka: *Při výpočtu jsme použili tlačítko +/−, které mění znaménko čísla na displeji, dále pak tlačítko EE pro vstup exponentu, zobrazujeme-li číslo v semilogaritmickém tvaru. Výsledek na displeji pak čteme ve tvaru $8{,}3.10^{-1}$, zaokrouhlíme-li číslo na jedno desetinné místo instrukcí Fix 1. Výsledek v pevné řádové čárce vyžaduje zrušení nastaveného tvaru tlačítky INV EE. Na dvě desetinná místa zaokrouhlíme instrukcí Fix 2.*

Cvičení

1. Bez použití paměti a závorek vypočítej následující příklady:

a) $5 - 3 \cdot \sin 40°$ b) $2 + \dfrac{5 \cdot \ln 0{,}87}{\operatorname{tg} 10°}$

c) $3 \cdot e^4 - \ln \operatorname{tg} 50°$

2. Urči zbytek po dělení čísla a číslem b výpočtem podle vztahu

$$z = a - b \cdot \operatorname{Int} \frac{a}{b}$$

pro a) a = 32, b = 7
b) a = 43276, b = 41
c) a = 96647, b = 127

3. Pomocí kosinové věty $c = \sqrt{a^2 + b^2 - 2ab\cos\gamma}$ vypočítej stranu **c** v trojúhelníku, je-li a) a = 33, b = 7, γ = 120°, b) a = 21, b = 5, γ = 60°.
4. Ze vzorce v příkladě 2.1 vypočítej rychlost **v** v závislosti na **M** a **h**. Potom vypočítej rychlost ve výšce h = 10 000 m potřebnou k dosažení M = 1.
5. Ze stejného vzorce vypočítej výšku **h** v závislosti na M a v. Potom vypočítej výšku, ve které při rychlosti v = 460 km/h dosáhneme M = 1.
6. Vzroste-li veličina a_0 *n*-krát pravidelně o p %, dosáhne hodnoty

$$a_n = a_0 \cdot \left(1 + \frac{p}{100}\right)^n .$$

Vypočítej a) a_n, je-li a_0 = 200, p = 5 %, n = 50, b) n, je-li a_n = 430, a_0 = 15, p = 2 %, c) a_0, je-li a_n = 100, p = 3 %, n = 70, d) p, je-li a_n = 300, a_0 = 50, n = 10.

3. Programové instrukce

Programovatelný kalkulátor může pracovat v režimu

1. přímého výpočtu, řízeného ručně tlačítky,
2. výpočtu řízeného programem, který vložíme do programové paměti ručně nebo přehráním z magnetické karty. Výpočet může být též řízen programem z vestavěného výměnného polovodičového modulu.

Při *ručním vkládání programu nejdříve* tlačítkem LRN (learn – učiti se) přepneme kalkulátor do režimu, ve kterém si bude pamatovat sled instrukcí vkládaných tlačítky. Soubor těchto instrukcí pak tvoří program. Používané instrukce můžeme rozdělit do několika skupin:

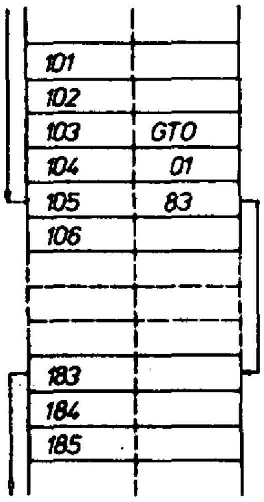
a) Jsou to jednak instrukce k provedení matematické operace. Kromě instrukcí uvedených již v 2. kapitole uvedeme ještě následující:
+/− ... změna znaménka čísla na displeji,
Fix *n* ... zaokrouhlení čísla na displeji na *n* desetinných míst, další výpočet však probíhá dále na plný počet desetinných míst (celkem 13 cifer). Odříznutí skrytých desetinných míst (tedy zaokrouhlení i pro další výpočet) můžeme provést sekvencí EE INV EE,
EE ... (enter exponent) příprava pro vložení exponentu desíti ($2{,}3.10^{-11}$ vložíme jako 2.3 EE 11 +/−),
Eng ... číslo na displeji se zobrazí tak, aby exponent byl násobkem tří,
D.MS ... převod stupňů, minut a vteřin na stupně,
INV D.MS . převod stupňů na stupně, minuty a vteřiny,
Deg ... přístroj nadále počítá goniometrické funkce pro argument ve stupních,
RAD ... přístroj počítá hodnoty goniometrických funkcí pro argument v radiánech.

b) Druhou skupinu tvoří instrukce k přesunu dat (čísel) mezi displejem a datovými registry. Číslo na displeji budeme nadále označovat jako číslo uložené v registru Rx, datové registry budeme adresovat dvouciferným číslem, tedy R34 značí datovou paměť 34. Zvláštní, tzv. testovací registr, budeme označovat Rt. Instrukce
RCL mn ... přesune číslo z Rmn na displej (RCL značí recall – vyvolání),
STO mn ... přesune číslo z displeje do Rmn (STO značí story – paměť),
Exc mn ... vymění čísla mezi Rmn a Rx (Exc značí exchange – výměna),
x ⇄ t ... vymění čísla mezi Rx a Rt.
Po instrukci RCL mn původní číslo v Rmn zůstává, STO mn také neovlivní číslo v Rx.

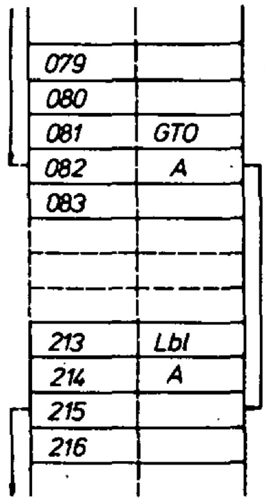
c) Zvláštní postavení mají instrukce k označení začátku a konce programu nebo k označení jeho částí. Začátek programu vyznačujeme tlačítkem **Lbl** (label – značka) s připojením některého z písmen A, B, C, D, E, A', B', C', D', E'. Místo tlačítka s písmenem můžeme použít i libovolného jiného tlačítka, které pak ztrácí svůj význam k provedení operace. Spojení Lbl A, Lbl E', Lbl x^2, Lbl sin apod. pak nazýváme návěštím. Na konci programu uvádíme **R/S** (run/stop – výpočet/zastavení výpočtu) nebo **INV SBR** (zastavení výpočtu nebo návrat z podprogramu).

d) Jednotlivé instrukce zapsané v programu provádí počítač v pořadí, jak za sebou následují, pokud není v programu zařazena instrukce ke skoku na jinou část programu.

1. **Nepodmíněný skok** programujeme instrukcí **GTO** (go to – jdi na) s uvedením absolutní nebo symbolické adresy. Na obr. 1 je zobrazen nepodmíněný skok na symbolickou adresu, představovanou návěštím Lbl A. Pro instrukci GTO A program pokračuje instrukcí za návěštím Lbl A. Trojciferná čísla v levé části obdélníčku udávají absolutní adresy, pod kterými jsou jednotlivé instrukce zapsány v programové paměti. Na obr. 2 je zobrazen nepodmíněný skok na absolutní adresu. Program pokračuje instrukcí na adrese 183.
2. **Skok na podprogram** následuje za instrukcí **SBR** s uvedením absolutní nebo symbolické adresy (subroutine – podprogram). Podprogram proběhne až k místu označenému instrukcí **INV SBR** (návrat z podprogramu), zde se však výpočet nezastaví, ale vrátí se na krok následující za příkazem ke skoku na podprogram.



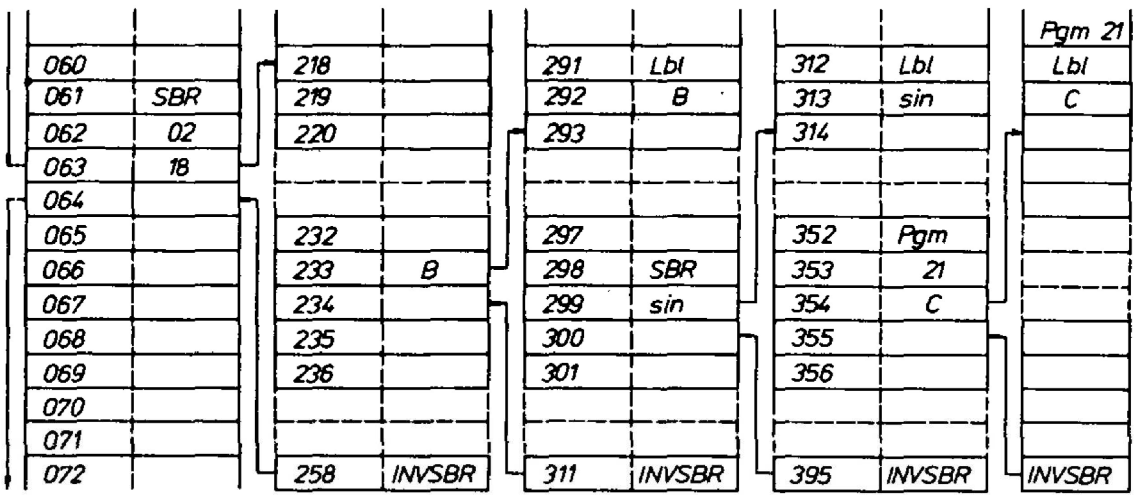
Obr. 1



Obr. 2

Podprogram označený návěštím s písmenem lze adresovat zkráceně bez instrukce SBR, tedy místo SBR B stačí pouhé B. Podprogram z polovodičového modulu voláme instrukcí **Pgm** mn (mn udává číslo podprogramu) opět s uvedením absolutní nebo symbolické adresy. Podprogramy lze řadit až do šesti úrovní, tzn., že podprogram může být podprogramem jiného podprogramu. Obr. 3 ukazuje strukturu tří úrovní podprogramů a užití podprogramu z polovodičového modulu.

Po instrukci SBR 218 běží podprogram od kroku 218. 233. krok obsahuje zkrácený příkaz ke skoku za návěští Lbl B. Tato část od kroku 291 až do kroku 311 tvoří druhou úroveň podprogramu vzhledem



Obr. 3

k programu hlavnímu. Třetí podprogram následuje za instrukcí SBR sin (což je příkaz ke skoku za návěští Lbl sin, se sinem jako funkcí nemá tato část programu nic společného). Tento program využívá ještě programu číslo 21 programového modulu od návěští Lbl C. Pak již začínají po instrukci INV SBR návraty do podprogramů nižších úrovní, dále pak hlavní program pokračuje krokem 064.

3. **Podmíněný skok** proběhne tehdy, je-li splněna podmínka tzv. rozhodovací operace. Adresa skoku následuje za rozhodovací instrukcí. Pokud podmínka v této operaci splněna není, příkaz ke skoku se vynechá a program pokračuje následující instrukcí. TI-58/59 používá tyto rozhodovací operace:

x=t ... je číslo v registru Rx (na displeji) rovno číslu v registru Rt?
INV x=t ... je číslo v Rx různé od čísla v Rt?
x≥t ... je číslo v Rx větší nebo rovno číslu v Rt?
INV x≥t ... je číslo v Rx menší než číslo v Rt?
Dsz n ... je číslo v Rn různé od nuly? Tato instrukce ještě před rozhodováním odečte 1 v Rn.
INV Dsz n . odečtení 1 v Rn a test „je číslo v Rn rovno nule?"
If flg m . je příznak (flag, přepínač) číslo m nastaven? Příznak se nastavuje v programu instrukcí **Stflg** m, ruší se instrukcí **INV Stflg** m.

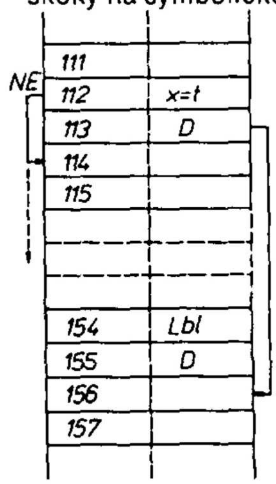
Na obr. 4 až 6 jsou zobrazeny podmíněné skoky na symbolickou i absolutní adresu.

e) Instrukce k přerušení výpočtu používáme tehdy, jestliže chceme zjistit mezivýsledek nebo kontrolovat postup výpočtu. Tlačítkem **Pause** přerušíme výpočet asi na 1 s, program pak dále pokračuje. Instrukce **R/S** zastaví probíhající výpočet nebo naopak program spouští.

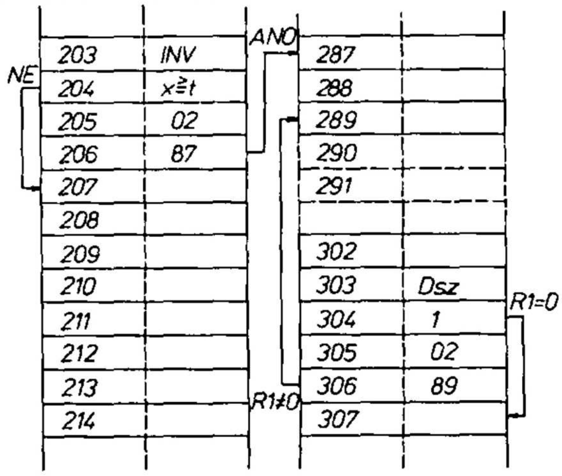
f) Zvláštní skupinu tvoří instrukce
CE ... mazání čísla na displeji, pokud bylo vloženo ručně tlačítky,
CLR ... nulování displeje a zrušení všech započatých operací,
CMs ... nulování všech datových registrů,
CP ... nulování registru Rt, v režimu přímého výpočtu se navíc maže vložený program,
RST ... čítač instrukcí se vrací na adresu 000, ruší se návraty z podprogramů a nastavené příznaky.

g) K vložení programu, k jeho opravě a orientaci v něm slouží instrukce
LRN ... přepnutí režimu výpočtového do režimu pro vkládání programu tlačítky,
SST ... (single step – jeden krok) posuv programu o jeden krok vpřed,
BST ... (back step – krok zpět) posuv programu o jeden krok zpět,
Ins ... uvolnění jednoho programového kroku pro vložení dodatečné instrukce,
Del ... vymázání kroku z programové paměti.

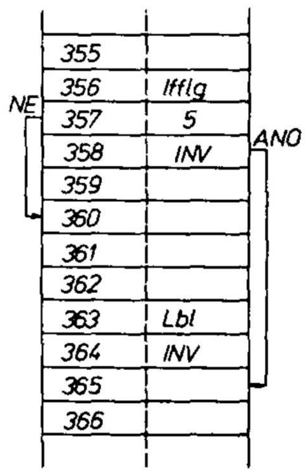
h) Instrukce k ovládání tiskárny:
Prt (print – tiskni) ... vytiskne se obsah displeje,
Adv ... posuv papíru v tiskárně o jeden řádek.



Obr. 4.



Obr. 5.



Obr. 6.

4. Programování podle vzorce

Pokud máme několikrát opakovaně dosazovat dané veličiny do stejného vzorce, můžeme použitý sled operací uložit do programové paměti a vlastní výpočet pak řídit programem. Postup si ukažme na následujícím příkladu:

Příklad 4. 1

Sestav program pro výpočet obvodu pravidelného n-úhelníka vepsaného do kružnice o poloměru r.

Řešení

Rozborem vzorce pro výpočet obvodu

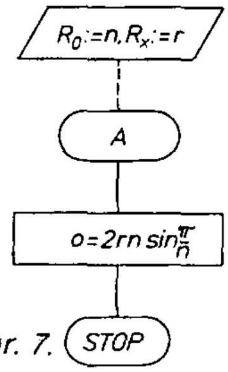
$$o = 2 \cdot r \cdot n \cdot \sin \frac{\pi}{n}$$

zjistíme, že číslo n se při výpočtu opakuje, a proto je před výpočtem vložíme do paměti R0 instrukcí STO 00. Poloměr r předpokládáme v registru Rx – tedy na displeji. Tyto dvě počáteční operace zapíšeme

$$R0: = n, Rx: \ = r$$

a čteme „do registru R0 dosadíme n, do registru Rx dosadíme r“. Při sestavování programu pomáhá nakreslení tzv. ***vývojového diagramu,*** *ve kterém graficky znázorníme logický postup výpočtu.*

Činnost týkající se vstupních hodnot zapisujeme do značky č. 12 normy ČSN 36 9030 – kosodélníku. Pro označení začátku a konce programu slouží mezní značka č. 29 – ovál. Do značky č. 1 – obdélníku zapisujeme sled použitých operací.



Obr. 7.

Počátek programu z příkladu 4.1 označíme návěštím Lbl A. Do oválu ve vývojovém diagramu proto zapíšeme písmeno A. Vztah pro výpočet obvodu zapíšeme do obdélníku a diagram zakončíme opět mezní značkou, do které zapíšeme STOP. Svislou čarou spojující jednotlivé značky vyznačíme jejich vzájemný vztah.

Podle diagramu (obr. 7) pak již snadno zapíšeme celý program:

Lbl A × 2 × RCL 00 × (π : RCL 00) Rad sin = R/S.

Výpočet začíná návěštím Lbl A za předpokladu, že poloměr r je na displeji, proto hned následuje instrukce krát a násobíme číslem 2, pak násobíme číslem n, které instrukcí RCL 00 přesuneme z paměti na displej. Závorkami vyznačíme, že nejdříve nutno vypočítat podíl a pak teprve jeho sinus (jinak by měl výpočet sinu přednost před dělením). Protože příslušný úhel je dán v radiánech, přepneme před výpočtem sinu kalkulátor do příslušného režimu instrukcí Rad. Ukončení výpočtu zajistíme instrukcí R/S.

Hotový program nyní vložíme do programové paměti kalkulátoru. Postupuj následujícím způsobem:

a) Zapni kalkulátor.

b) Stiskni tlačítko LRN. Na displeji se objeví 000 00. První trojčíslí udává pořadové číslo vkládané instrukce, význam druhého dvojčíslí si vysvětlíme později.

c) Vlož sestavený program tím způsobem, že tiskneš odpovídající tlačítka. Před instrukcemi Lbl, Rad a sin stiskni tlačítko 2nd.

d) Stiskni opět tlačítko LRN. Tím je kalkulátor připraven k výpočtu.

Nejdříve si ale vložený program překontrolujeme. Tlačítkem RST vrátíme čítač instrukcí na adresu 000 a tlačítkem LRN si zobrazíme na displeji program. Objeví se 000 76. Dvouciferné číslo udává kód vložené instrukce. Většinou značí souřadnice příslušného tlačítka. Např. 34 je kód 4. tlačítka ve 3. řadě. Pokud bylo použito funkčního tlačítka 2nd, zvětšuje se druhá číslice o 5. Tedy 76 udává 1. tlačítko v 7. řadě – 2nd Lbl.

Zkontrolujeme si tedy správnost programu podle těchto čísel. Tlačítkem SST si vždy zobrazíme na displeji následující programový krok (BST vrací o krok zpět). Tedy

000	76	**Lbl**	16,8	poloměr kružnice ***r***
001	11	**A**	16,8	
002	65	**×**	16,8	
003	02	**2**	2	
004	65	**×**	33,6	součin ***2r***
005	43	**RCL**	33,6	
006	00	**0**	52	počet stran ***n***
007	65	**×**	1747,2	součin ***2rn***
008	53	**(**	1747,2	
009	89	π	3,141592654	
010	55	**:**	3,141592654	
011	43	**RCL**	3,141592654	
012	00	**0**	52	***n***
013	54	**)**	.0604152433	podíl π/***n***
014	70	**Rad**	.0604152433	výpočet proběhne v obloukové míře

015	38	sin	.0603784974	hodnota sin π/n
016	95	=	105,4933107	výsledek – obvod mnohoúhelníka
017	91	R/S	105,4933107 zaokrouhlení na 1 desetinné místo Fix 1 105,5	

V třetím sloupci je uveden význam kódových čísel. Pokud některá instrukce nesouhlasí, stiskněte správné tlačítko, chybná instrukce se přepíše správnou.

Tlačítkem LRN přepněte kalkulátor zpět do výpočetního režimu a zkuste projít program krok po kroku pro hodnoty n = 52, r = 16,8. Postupujeme takto:

52 STO 00 16,8 RST

a tlačítkem SST prověříme jednotlivé programové kroky. Na displeji se budou objevovat dílčí výpočty podle čtvrtého sloupce tabulky. Výsledky zaokrouhlíme na 1 desetinné místo instrukcí Fix 1.

Tím jsme podrobně rozebrali postup při vkládání programu a můžeme přistoupit k vlastnímu výpočtu. Jestliže stiskneme tlačítko A, proběhne rychle výpočet od návěští Lbl A až k instrukci R/S. Doplňte sami následující tabulku:

n	r	o	postup	na displeji
52	16,8	105,5	52 STO 00 16,8 A	105,5
52	180		180 A	1130,3
10	20		10 STO 00 20 A	
1000	1		1000 STO 00 A Fix 5	
1000000	0,5		1000000 STO 00 .5 A	

*Vkládání čísel bychom mohli zjednodušit užitím registru Rt. Jestliže dané veličiny vložíme postupem **r x≷t n A**, bude program*

Lbl A x x≷t x 2 x (π : x≷t)Rad sin = R/S

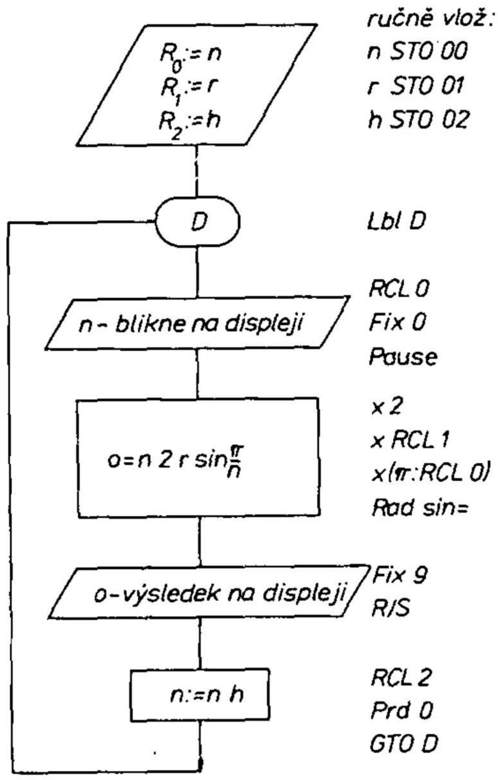
Vyzkoušejte tento program

Příklad 4.2

*Doplňme předešlý příklad podmínkou, aby se výpočet obvodu mnohoúhelníka opakoval pro **n.h** úhelník, kde **h** je daná konstanta. Poloměr kružnice předpokládáme před zahájením výpočtu v R1 a krok **h** v R2. Počáteční hodnota **n** bude v registru R0.*

Řešení

*Vývojový diagram na obr. 8 názorně ukazuje stavbu celého programu. Počátek označíme návěštím **Lbl D** a na displeji nejdříve na okamžik zobrazíme počet stran **n** instrukcí **Pause**. Protože n je číslo přirozené, zajistíme tvar bez desetinných míst instrukcí **Fix0**. Následuje výpočet obvodu, po zaokrouhlení výsledku na 9 desetinných míst výpočet zastavíme instrukcí **R/S**. Pokračování programu předpokládá stisknutí tlačítka **R/S**, číslo v R0 vynásobíme krokem h, takže v R0 je nová hodnota n. Příkazem ke skoku **GTO D** se program vrátí za návěští **Lbl D** a program probíhá znovu pro nové **n**. Vložte nyní program do programové paměti známým postupem **LRN Lbl D . . . GTO D LRN**. Vyzkoušejte program pro hodnoty r = 0,5, h = 2, počáteční hodnota n = 10.*



Obr. 8.

Postup:

10 STO 00 . 5 STO 01 2 STO 02 D

*Zastavený výpočet vždy spustíme tlačítkem **R/S**. Nahradíme-li tuto instrukci instrukcí **Pause**, výpočet pokračuje asi po 1 s. Pokud připojíme ke kalkulátoru tiskárnu, nahradíme obě Pause instrukcí **Print**, n i o se pak vytisknou. Pro n= 5242880 se obvod shoduje s Ludolfovým číslem na 9 desetinných míst. Tato shoda platí už od n = 3141593, jak se můžete přesvědčit.*

Příklad 4.3

Vyzkoušejte si různé způsoby zadání vstupních veličin. V příkladu 4.2 jsme dané veličiny předem vložili do registrů R0, R1, R2 ručně *tlačítkem STO. I tato činnost se dá nahradit programem. Uveďme si dva způsoby:*

1. ***Lbl A STO 0 R/S STO 1 R/S STO 2 R/S*** *– nebo místo poslední instrukce Lbl D . . .*
   Při použití tohoto programu zadáme vstupní veličiny takto:
   n A r R/S h R/S
   přesun veličin do registrů zajistí sám program.
2. ***Lbl A STO 0 R/S Lbl B STO 1 R/S Lbl C STO 2 R/S***

Výhodou tohoto programu je možnost zadávat veličiny v libovolném pořadí, např.
n A r B h C *nebo r B n A h C ap.*
Vyzkoušejte oba způsoby.

Příklad 4.4

Indukčnost cívky s činným odporem R lze vypočítat ze vztahu

$$L = \frac{1}{2\pi f} \cdot \sqrt{\left(\frac{U}{I}\right)^2 - R^2}\ ,$$

kde U [V] je napětí, I [A] proud a f [Hz] kmitočet střídavého proudu.

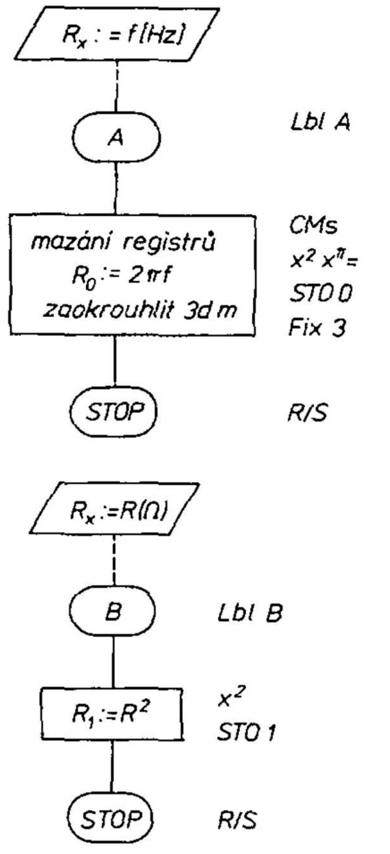
Při měření indukčnosti cívky se vzduchovým jádrem Ohmovou metodou jsme naměřili hodnoty uvedené v následující tabulce. Sestavte program pro zpracování měření.

číslo měř.	I [A]	U [V]	L [H]
1	0,10	14,6	
2	0,15	21,9	
3	0,20	29,1	
4	0,25	36,5	
5	0,30	43,7	
6	0,35	51,0	
f = 49,5 Hz, R = 52 Ω			

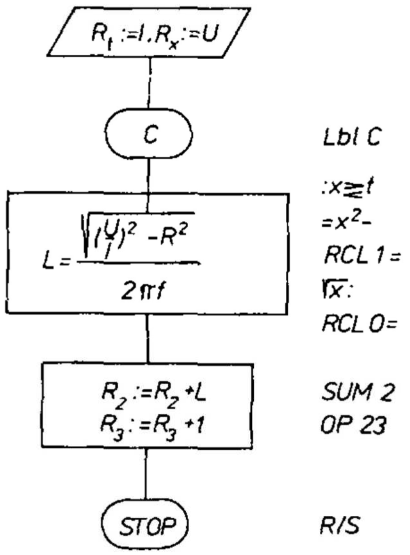
Řešení:

*Kmitočet a odpor jsou veličiny, které se během měření nemění. Při výpočtu indukčnosti použijeme součin 2πf, který uložíme do registru R0 programem **A**. Sledujte na vývojovém diagramu (obr. 9, 10), kde je výpočet doplněn počátečním mazáním datových registrů a zaokrouhlením na 3 desetinná místa. Programem **B** uchováme R^2 do R2. Pro každou dvojici naměřených hodnot I a U použijeme program **C**. Vypočítaná hodnota indukčnosti se přičte do R2, abychom na závěr mohli vypočítat průměrnou hodnotu indukčnosti. Počet měření registrujeme v R3. Operace **Op 23** totiž zvětší*

Obr. 9.



Obr. 10.



*číslo v R3 o jedničku. Průměrnou hodnotu indukčnosti spočítáme programem **Lbl D.** Stačí vydělit součet indukčností v R2 počtem měření v R3.*

I když jsou programové kroky uvedeny vedle vývojových diagramů, uveďme si celý program znovu:

Lbl A CMs × 2 × π = STO 0 Fix 3 R/S Lbl B x² STO 1 R/S Lbl C : x≷t = x² − RCL 1 = √x : RCL 0 = SUM 2 Op 23 R/S Lbl D RCL 2 : RCL 3 = R/S

Nyní zpracuj výsledky měření z výše uvedené tabulky. Postup:

a) ***LRN*** *– vlož program –* ***LRN****,*
b) vlož kmitočet f na displej – stiskni ***A*** *– na displeji 2πf,*
c) vlož odpor R na displej – stiskni ***B*** *– na displeji R², *
d) vlož proud a napětí: ***I x≷t U C*** *– na displeji vypočítaná indukčnost L,*
e) krok d) opakuj pro ostatní měření,
f) po skončení všech výpočtů stiskni ***D*** *– na displeji průměrná hodnota L.*

Prostuduj si podrobně tento příklad a pak vypracuj následující cvičení.

Cvičení

7. Sestav program pro výpočet obvodu kružnice o poloměru **r**. Program označ jako **A**, poloměr r vkládáme před zahájením výpočtu na displej.
8. Doplň předešlý příklad programem **B** pro výpočet obsahu kruhu.
9. Uprav poslední dva programy tak, aby po stisku tlačítka **C** se na displeji objevil nejdříve obvod a pak obsah.
10. Sestav program, který k danému napětí *U* (V) a proudu *I* (A) spočítá činný odpor *R* (Ω) a výkon P (W). Návod:
    Lbl A . . . přesun *U* z displeje do R1,
    Lbl B . . . přesun *I* z displeje do R2,
    Lbl C . . . výpočet *R = U/I*,
    Lbl D . . . výpočet *P = U.I.*
11. V obvodu střídavého proudu měříme *U* (V), proud *I* (A) a činný výkon *P* (W). Sestav následující program pro zpracování naměřených hodnot:
    Lbl A . . . přesun *U* z displeje do R1,
    Lbl B . . . přesun *I* z displeje do R2,
    Lbl C . . . přesun *P* z displeje do R3,
    Lbl A' . . . výpočet $\cos\varphi = P/(U.I)$,
    Lbl B' výpočet zdánlivého výkonu P_s = U.I,
    Lbl C' výpočet jalového výkonu $P_q = U.I \sin\varphi$ (sinus urči z kosinu instrukcemi INV cos sin)
    Lbl D' výpočet činného odporu proudu $I_w = I.\cos\varphi$,
    Lbl E' výpočet jalového proudu $I_j = I.\sin\varphi$.

Program zkontroluj podle následující tabulky (1. měření) a tabulku doplň.

č. m	*U*(V)	*I*(A)	*P*(W)	$\cos\varphi$	P_s (VA)	P_q (VAr)	I_w (A)	I_j (A)
1	220	1,99	400	0,91	438	178	1,82	0,81
2	165	1,51	222					
3	110	1,10	95,6					

12. Podle příkladu 3 vytvoř program pro výpočet strany c v trojúhelníku, je-li dáno: strana **a**: **Lbl A** . . . do R1, strana **b**: **Lbl B** . . . do R2, úhel **γ**: **Lbl C'** . . . do R3. Stranu **c** spočítej programem **Lbl C**.
13. Podle vztahu v příkladu 2.1 utvoř program pro výpočet Machova čísla. Vložení rychlosti **v** a výšky **h** a zahájení výpočtu proveď ručně takto: **v x≷t h A**. Doplň tabulku:

v (km/h)	650	650	800	800	800	800	200	200	1225
h (m)	7600	12 000	3000	6000	12 000	20 000	1000	5000	0
M	0,83								

14. Vytvoř program, který k dané hodnotě x vypočítá funkční hodnotu y v následujících případech:
    a) $y = 4x - 5$ b) $y = 3x^2 - 5x + 1$
    c) $y = -x^7 + 3x^6 - 4x^3 + 5x^2 - x + 7$
    d) $y = x - \sin x - 1{,}9$ (v radiánech)
    e) $y = \mathrm{tg}\,x - x - 0{,}01$ (v radiánech)
15. Sestav program, který k daným stranám v trojúhelníku **a, b, c** vypočítá po řadě obsah **S**, poloměr kružnice vepsané **ρ**, opsané r a vnitřní úhly **α, β, γ**. Dané strany vlož po řadě do registrů R1, R2, R3 programy **A, B, C**, pokračování proveď instrukcí **R/S**.

● ● ●

5. Programové vybavení TI-58/59 – softwarový integrovaný modul

V minulé kapitole jsme poznali, jak správně sestavený program urychluje opakované výpočty. Příprava programu a jeho ladění (tzn. zjišťování jeho správnosti, úplnosti, upravování, zjednodušování ap.) je však pracná a vyžaduje kromě potřebných znalostí i praxi. Proto výrobci počítačů dodávají hotové programy pro často se opakující úlohy jako součást tzv. programového vybavení – software. Zdokonalující se polovodičová technika umožnila výrobci TI vložit do integrovaného polovodičového modulu 25 různých programů o celkové délce 5000 kroků. S některými se nyní seznámíme.

Další programy jsou uvedené v příručce Standard Software Modul. Pro lepší orientaci lze zasunout do výřezu pod displejem štítek se stručným návodem k použití programu.

Tabulka některých funkcí polovodičového programového modulu:

Příklad 5.1: řešení trojúhelníka sss	vstupní hodnota	tlačítko	displej	
	–	**Pgm 11**	–	Vyzkoušej pro
vlož délku strany **a**	a	**A**	a	a = 5
vlož délku strany **b**	b	**B**	b	b = 12
vlož délku strany **c**	c	**C**	c	c = 13
výpočet úhlu α	–	**A'**	α	α =
výpočet úhlu β	–	**B'**	β	β =
výpočet úhlu γ	–	**C'**	γ	γ =
Příklad 5.2: řešení trojúhelníka sus				
	–	**Pgm 11**	–	Vyzkoušej pro
vlož délku strany **a**	a	**A**	a	a = 21
vlož délku strany **b**	b	**B**	b	b = 5
vlož velikost úhlu γ	γ	**C**	γ	γ = 60°
výpočet délky strany **c**	–	**E**	c	c =
výpočet úhlu α	–	**B'**	α	α =
výpočet úhlu β	–	**C'**	β	β =
Příklad 5.3: kruhová výseč a úseč				
	–	**Pgm 13**	–	Vyzkoušej pro
vlož poloměr kružnice **r**	r	**B**	r	r = 8
vlož délku tětivy **t**	t	**D**	t	t = 12
výpočet středového úhlu (rad)	–	**A'**	α	α =
výpočet obsahu kruhové výseče	–	**E**	Sv	Sv =
výpočet obsahu kruhové úseče	–	**E'**	Su	Su =
Příklad 5.4: výpočet hodnoty polynomu				
$f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$	–	**Pgm 07**	–	Vyzkoušej pro
vlož stupeň polynomu **n**	n	**A**	n	$n = 5, a_0 = 5$
	0	**B**	0	$a_1 = -3, a_2 = 0$
vlož koeficient **a0**	a0	**R/S**	a0	
vlož koeficient **a1**	a1	**R/S**	a1	$a_3 = 4, a_4 = -2$
vlož koeficient **a2**	a2	**R/S**	a2	
.	.	.	.	$a_5 = 1$
.	.	.	.	
.	.	.	.	x = 2, f(x)=
vlož hodnotu proměnné **x**	x	**C**	f(x)	x = −6, f(x)=
vlož další hodnotu **x**	x	**C**	f(x)	
Příklad 5.5: počet dní mezi dvěma daty				
	–	**Pgm 20**	–	vypočítej,
vlož prvé datum **d1** ve tvaru				kolik dní,
mmdd . rrrr				uplynulo od
(např. datum narození 5. září 1963)				Tvého narození
ve tvaru 905.1963)	d1	**A**	0	
vlož druhé datum **d2**	d2	**B**	0	
výpočet počtu dnů mezi d1 a d2	–	**C**	d	

6. Podmíněný skok, skok na podprogram, nepřímé adresování, operace Dsz

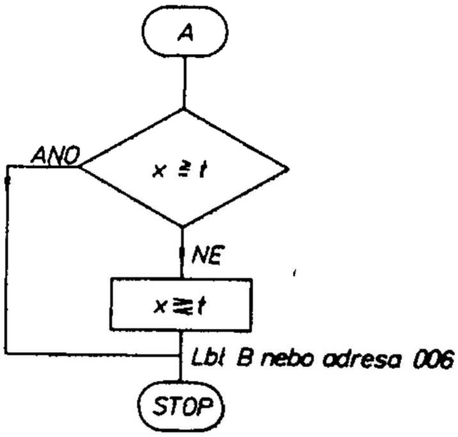
O podmíněném skoku následujícím za rozhodovací (logickou) operací jsme se zmínili na str. 5, použití si nyní ukážeme na několika příkladech.

Příklad 6.1

V registrech Rx a Rt máme dvě libovolná čísla. Sestav program, který vybere větší z těchto čísel a přesune je na displej (do registru Rx).

Řešení

Ve vývojovém diagramu úlohy (obr. 11) je použita schematická značka č. 2 pro rozhodovací operaci. Tvoří ji kosodélník s jedním vstupem a se dvěma výstupy. Je-li podmínka ***x ≧ t*** *splněna, uskutečníme skok (podmíněný) na jinou část programu. V opačném případě program pokračuje instrukcí x≷t, takže na displeji bude opět větší z daných čísel.*



Obr. 11

Použijeme-li skok na symbolickou adresu Lbl B, bude program vypadat takto: ***Lbl A x≧t B x≷t Lbl B R/S.*** *Prostuduj si znovu obrázky 4 až 6.*

Při skoku na absolutní adresu nahradíme příkaz ke skoku B adresou instrukce R/S. Tím se program zkrátí o 1 krok:

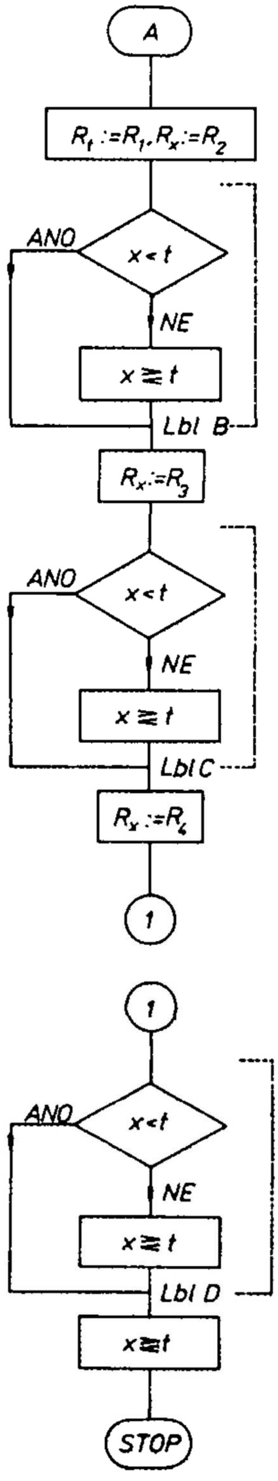
Lbl A x ≧ t 0 06 x≷t R/S

Příklad 6.2

V registrech R1 až R4 máme čtyři libovolná čísla. Sestav program, který vybere z těchto čísel číslo největší a přesune je na displej.

Řešení

Uvedený postup je modifikací předcházejícího příkladu, větší číslo však necháváme v Rt. Postupně přesouváme na displej čísla z R2,



Obr. 12

R3, R4 a testujeme, zda je x <t. Není-li tato podmínka splněna, zaměníme obsahy registrů Rx a Rt. Závěrem přesuneme největší číslo z Rt na displej.

Podle vývojového diagramu (obr. 12) pak snadno sestavíme program:

Lbl A RCL 1 x≷t RCL 2 INV x ≥ t B x≷t
Lbl B RCL 3 INV x ≥ t C x≷t
Lbl C RCL 4 INV x ≥ t D x≷t Lbl D x≷t R/S

Na vývojovém diagramu jsou tečkovaně vyznačeny tři části, které se v programu opakují. Přepišme proto tuto část za návěští Lbl E a ukončeme ji instrukcí INV SBR – návrat z podprogramu. Ve vlastním programu pak tyto části nahraďme instrukcí ke skoku na podprogram E. Dostaneme pak program

Lbl E INV x ≥ t 007 x≷t INVSBR Lbl A RCL 1 x≷t RCL 2 E RCL 3 E RCL 4 E x≷t R/S.

Ušetřili jsme 7 programových kroků. Vyzkoušejte.

Příklad 6.3.

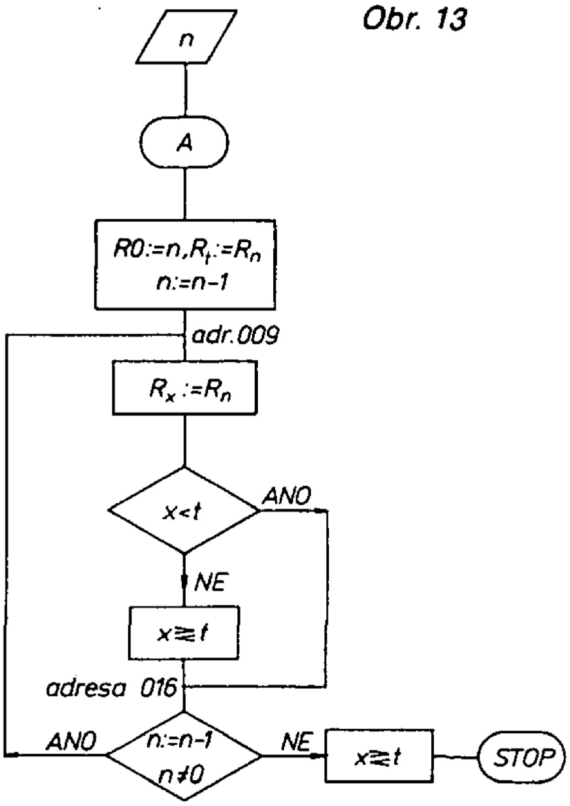
V registrech R1 až Rn máme n čísel. Sestav program, který vybere z těchto čísel číslo největší a přesune je na displej.

Řešení:

*Úloha je zobecněním postupu použitého v příkladu 6.2 (obr. 13). Počet čísel n uložíme nejdříve do registru R0, který použijeme jako indexregistr pro nepřímé adresování a jako řídicí registr smyčky mezi 9. a 19. krokem. Při nepřímém adresování instrukcí **RCL Ind 00** se přesune na displej číslo z registru, jehož adresu najdeme v R0. Je-li tedy např. v R0 číslo n=8, bude na displeji číslo z R8. Zmenšení čísla v R0 o jedničku provedeme instrukcí **Op 30** (v úvodu programu), dále pak použijeme instrukce **Dsz 0**, která jednak zmenší číslo v R0 o jedničku, ale která také testuje, zda v R0 není nula. Pokud R0 ≠ 0, uskuteční se skok na adresu následující za instrukcí Dsz 0 (v našem případě na adresu 009) a část programu (tzv. smyčka) se bude opakovat tak dlouho, dokud nebude R0 = 0. Pak se příkaz ke skoku na adresu 009 přeskočí a program pokračuje, z Rt se přesune na displej nalezené číslo a výpočet se ukončí. Smyčka a nepřímé adresování umožňují jinak celkem rozsáhlý program zkrátit na 22 kroky:*

Lbl A STO 0 RCL Ind 0 x≷t Op 30 RCL Ind 0 INV x ≥ t 016 x≷t Dsz 0 009 x≷t R/S

Obr. 13



Příklad 6.4.

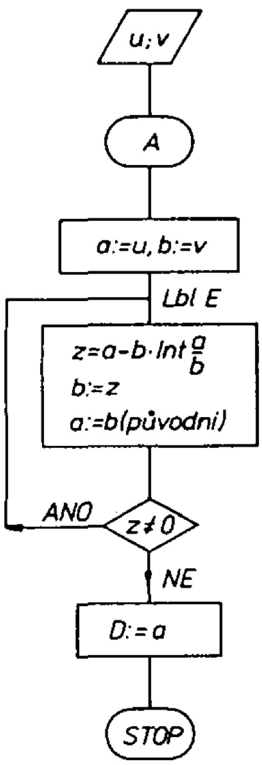
*Sestav program pro určení největšího společného dělitele čísel **u** a **v**.*

Řešení

Pro určení D (u; v) použijeme Eukleidův algoritmus, podle kterého nejprve dělíme větší číslo menším, potom menší číslo dělíme prvním zbytkem, dále první zbytek dělíme druhým zbytkem a podobně pokračujeme tak dlouho, až dostaneme podíl beze zbytku (obr. 14). První dělitel, při kterém vyjde dělení beze zbytku, je hledaný D. Např. hledáme-li tímto způsobem D (660; 168), postupujeme takto: 660 = 3.168 + 156, 168 = 1.156 + 12, 156 = 13.12 + 0, poslední nenulový zbytek je 12, proto D (660; 168) = 12.

Zbytek po dělení čísla a číslem b určíme podle cvičení 2 jako z = a – b. Int $\frac{a}{b}$. Za a pak dosadíme b a za b dosadíme z. Pokud je z ≠ 0, výpočet opakujeme, jinak je D (u, v) = a. Před testováním zbytku z vynulujeme registr Rt instrukcí CP. Daná čísla u a v předpokládejme před počátkem výpočtu v Rx a Rt. Zvolíme-li obsazení registrů R1 = a, R2 = b, R3 = u, R4 = v, můžeme podle vývojového diagramu sestavit program:

Lbl A STO 2 STO 4 x≷t STO 1 STO 3 Lbl E RCL 1 – RC 2 x (RCL 1 : RCL 2) Int = Exc 2 STO 1 CP RCL 2 INV x = t E RCL 1 INVSBR.



Obr. 14

V tomto programu jsme použili instrukci Exc 2, která vyměňuje obsah displeje (zbytek z) a obsah registru R2 (b). Uschování u a v v R3 a R4 je zde zbytečné, ale využijeme je ve cvičení v příkladu na nejmenší společný násobek dvou čísel.

Urči podle tohoto programu:

D (18; 72) =

D (24 335; 413478) =

D (484909; 216775) =

Postup:

u x⇄t v A

Příklad 6.5

Sestav program pro výpočet jednotlivých členů rekurentní posloupnosti dané vztahem $a_n = a_{n-1} + a_{n-2}$ (Fibonacciho posloupnost) za předpokladu, že jsou dány členy a_1, a_2.

Řešení

Budeme-li ukládat člen a_{n-1} do registru R1 a člen a_{n-2} do registru R2, dostaneme sečtením obsahů těchto registrů člen a_n, který zobrazíme na displeji instrukcí Pause. Pak přesuneme a_n do R_2 a a_{n-2} do R1 a výpočet budeme opakovat. Jednoduchý program

Lbl A RCL 1 + RCL 2 = Pause Exc 2 STO 1 GTO A

vyžaduje na počátku programu vložení a_1, a_2 do R1, R2. Např. pro $a_1 = a_2 = 1$ vložíme 1 STO 1 STO 2 A, na displeji se postupně objevují čísla 2, 3, 5, 8, 13, 21, 34, 55....... atd.

Chceme-li, aby se výpočet po zjištění k-tého členu zastavil, vložíme do R0 číslo k-2 a instrukci GTO A nahradíme instrukcí Dsz 0 A RCL 2 R/S. Registr R0 pak řídí výpočet ve smyčce tak dlouho, pokud R0 ≠ 0. Potom se výpočet zastaví, na displeji bude a_k.

Cvičení

16. V registrech R1 až R3 máme tři libovolná čísla. Sestav program, který je uspořádá podle velikosti tak, aby v R1 bylo číslo nejmenší a v R3 největší. Pokus se zobecnit program pro *n* čísel.
17. Program podle příkladu 6.4 doplň o program Lbl B pro výpočet nejmenšího společného násobku dvou čísel *u, v*. Program A pro výpočet D ponech jako podprogram a násobek vypočítej podle vztahu $n\,(u;\,v) = \dfrac{u \cdot v}{D\,(u;v)}$. Urči nejmenší společné násobky čísel z příkladu 6.4.
18. Sestav program pro výpočet největšího společného dělitele tří čísel *u, v, w*. Návod: nejdříve určíme největšího společného dělitele dvou čísel D_1 a pak dělitele zbývajícího čísla a D_1. Program z příkladu 6.4 nechte jako podprogram.
19. Sestav program pro výpočet jednotlivých členů geometrické posloupnosti dané rekurentním vzorcem $a_n = a_{n-1} \cdot q$. Předpokládej R1 = q, Rx = a_1.
20. Sestav program pro výpočet členů Fibonacciho posloupnosti zlomků
$$\frac{1}{1}, \frac{1}{2}, \frac{2}{3}, \frac{3}{5}, \frac{5}{8}, \frac{8}{13}, \frac{13}{21}, \frac{21}{34}, \frac{34}{55}, \frac{55}{89}, \ldots$$
Podíly vyjádři jako desetinné číslo. Tato posloupnost má limitu $\dfrac{\sqrt{5}-1}{2} \doteq 0{,}618033989$.
21. Sestav program pro výpočet druhých odmocnin přirozených čísel od 1 do 100. Na displeji zobraz nejdříve přirozené číslo a pak jeho odmocninu.

7. Řešení rovnic

Příklad 7.1

Sestav program pro řešení kvadratické rovnice s reálnými koeficienty

$$ax^2 + bx + c = 0.$$

Koeficienty a, b, c vkládej po řadě tlačítky A, B, C, pro výpočet použij tlačítka D.

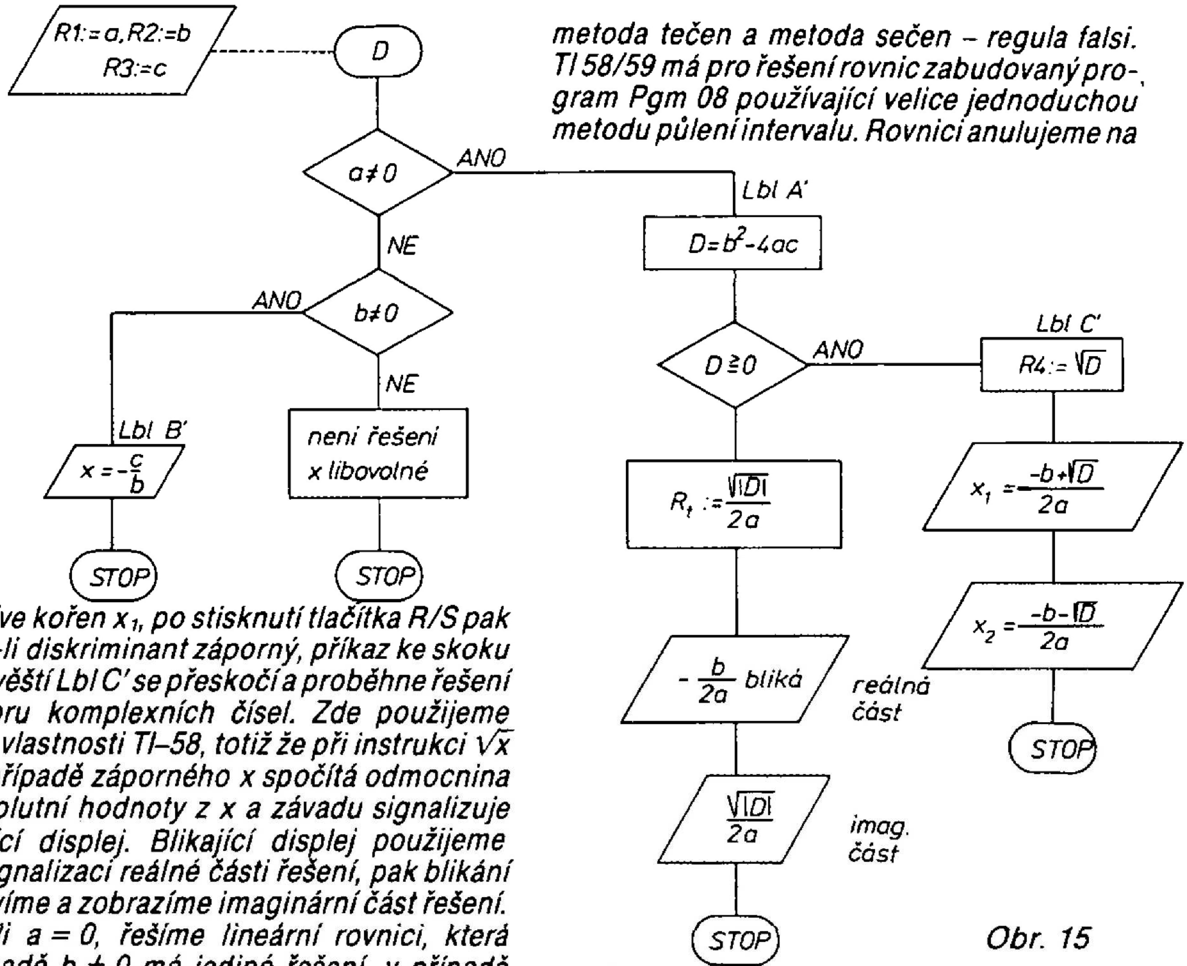
Řešení

Koeficienty a, b, c nejdříve uložíme po řadě do registrů R1, R2, R3 programy

Lbl A × 2 = STO 1 R/S

Lbl B STO 2 R/S Lbl C STO 3 R/S.

Koeficient a je vynásoben 2, takže v R1 máme k dispozici 2a. Vlastní výpočet začíná návěštím Lbl D. V případě, že a ≠ 0, proběhne skok na návěští Lbl A' a spočítá se diskriminant kvadratické rovnice D, který testujeme, zda je nezáporný. V kladném případě se na displeji objeví



Obr. 15

nejdříve kořen x_1, po stisknutí tlačítka R/S pak x_2. Je-li diskriminant záporný, příkaz ke skoku na návěští Lbl C' se přeskočí a proběhne řešení v oboru komplexních čísel. Zde použijeme jedné vlastnosti TI–58, totiž že při instrukci $\sqrt{x}$ se v případě záporného x spočítá odmocnina z absolutní hodnoty z x a závadu signalizuje blikající displej. Blikající displej použijeme pro signalizaci reálné části řešení, pak blikání zastavíme a zobrazíme imaginární část řešení.

Je-li a = 0, řešíme lineární rovnici, která v případě b ≠ 0 má jediné řešení, v případě b = 0 signalizujeme blikajícím displejem (samé devítky) zbývající případy (není řešení nebo je x libovolné). Podle vývojového diagramu (obr. 15) pak sestavíme program:

Lbl D CP RCL 1 INV x = t A' RCL 2 INV x = t B' 1/x R/S Lbl B' 1/x × RCL 3 = +/− R/S Lbl A' RCL 2 x² − 2 × RCL 1 × RCL 3 = x ≧ t C' √x : RCL 1 = x⇄t RCL 2 : RCL 1 = +/− R/S CLR x⇄t R/S Lbl C' √x STO 4 − RCL 2 = :RCL 1 = R/S RCL 2 + RCL 4 = +/− :RCL 1 = R/S

Vyzkoušejte si program na následujících rovnicích:

a) $2x^2 - 15x - 8 = 0$ ... 2 A 15 +/− B 8 +/− C D ... 8 (x_1) R/S −0,5 (x_2)

b) $2x^2 - 6x + 17 = 0$ ... 2 A 6 +/− B 17 C D ... 1,5 bliká-reálná část R/S 2,5 imaginární část

Příklad 7.2

V množině reálných čísel řeš rovnice

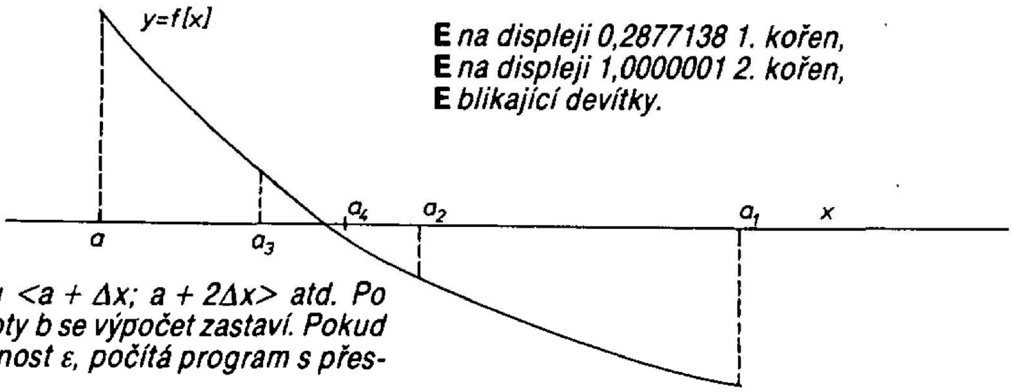
a) $x^2 - 2^x = 0$

b) $x^3 + 2x^2 - 4,52x + 0,816 = 0$

c) $\sin x - \cos x - e^x + \ln x + 1,3365088 = 0$

Pro numerické řešení rovnic existuje řada metod, z nichž nejznámější jsou Newtonova metoda tečen a metoda sečen – regula falsi. TI 58/59 má pro řešení rovnic zabudovaný program Pgm 08 používající velice jednoduchou metodu půlení intervalu. Rovnici anulujeme na tvar f (x) = 0 a hledáme nulové body funkce y = f (x). Jestliže je funkce f (x) spojitá v intervalu $<a; a_1>$ a jestliže platí $f(a) \cdot f(a_1) < 0$, bude hledané řešení ležet uvnitř intervalu $<a; a_1>$. Určíme tedy střed intervalu $a_2 = \frac{a + a_1}{2}$ a podle znaménka hodnoty $f(a_2)$ zúžíme interval obsahující hledaný kořen na interval $<a; a_2>$ (na obr. 16) popř. $<a_2; a_1>$. Celý postup pak opakujeme pro tento nový interval. Na obr. tak dostáváme intervaly $<a; a_2>$, $<a_3; a_2>$, $<a_3; a_4>$ atd. Předem si obvykle stanovíme přesnost vypočítaného kořenu tím způsobem, že zvolíme dostatečně malé kladné číslo ε a intervaly půlíme tak dlouho, až je $|a_{n-1} - a_n| < \varepsilon$, tzn. že šířka intervalu obsahujícího hledaný kořen nepřesáhne číslo ε. Za řešení rovnice pak pokládáme hodnotu a_n.

*Program **Pgm 08** vyžaduje především volbu intervalu <a; b>, ve kterém předpokládáme výskyt kořenu, a dále volíme krok Δx, kterým bude program daný interval prohledávat. Nejdříve totiž zjistí kořen v intervalu <a; a+ Δx>,* Obr. 16



dále v intervalu <a + Δx; a + 2Δx> atd. Po dosažení hodnoty b se výpočet zastaví. Pokud nezadáme přesnost ε, počítá program s přesností 0,01.

Do programové operační paměti vložíme za návěští Lbl A' program pro výpočet f(x) a zakončíme jej instrukcí INV SBR. Hodnotu x předpokládáme na displeji a můžeme si ji uschovat do paměti R0. V programu užíváme pouze závorky, nesmíme použít =.

Postup výpočtu vypadá pak takto:

RST LRN LBL A'... program pro výpočet f(x)...INV SBR LRN Pgm 08

vlož hranice intervalu, ve kterém hledáš řešení: a ... A, b ... B; vlož zvolený krok: x ... C; vlož požadovanou přesnost: ε ... D; řešení rovnice: ... E na displeji se zobrazí vypočítaný kořen, další kořen se počítá po stisknutí tlačítka E.

Jestliže v daném intervalu již není žádný další kořen, bliká na displeji 9,9999999 99.

Dané rovnice podle tohoto vzoru vyřešíme následujícím způsobem:

a) $x^2 - 2^x = 0$

RST LRN LBL A' (STO 0 x² − 2 yˣ RCL 0) INV SBR LRN Pgm 08 −1,5 A 4,5 B 1 C 0,001 D

E *na displeji −0,767 1. kořen,*
E *na displeji 2,000 2. kořen,*
E *na displeji 4,000 3. kořen.*

b) $x^3 + 2x^2 - 4{,}52x + 0{,}816 = 0$

RST LRN LBL A' (STO 0 × x² + 2 × RCL 0 x² − 4,52 × RCL 0 + 0,816) INV SBR LRN Pgm 08
−5 A 3 B 1 C 0,001 D

E *na displeji −3,400 1. kořen,*
E *na displeji 0,200 2. kořen,*
E *na displeji 1,200 3. kořen,*
E *blikající devítky, v intervalu není další řešení.*

Vzhledem k danému ε = 0,001 zaokrouhlíme opět na 3 desetinná místa instrukcí ***Fix 3.***

c) $\sin x + \cos x - e^x + \ln x + 1{,}3365088 = 0$

RST LRN Lbl A' (STO 0 sin + RCL 0 cos − RCL 0 INV lnx + RCL 0 lnx + 1,3365088) INV SBR LRN Pgm 08 Rad
0,0001 A 10 B 0,5 C 0,0000001 D

E *na displeji 0,2877138 1. kořen,*
E *na displeji 1,0000001 2. kořen,*
E *blikající devítky.*

Množina kořenů nemusí být v tomto případě úplná. Pokud za návěští Lbl A' zařadíme instrukci Pause, můžeme na displeji pozorovat postupná přiblížení k hledanému kořenu.

Příklad 7.3

Řeš soustavu lineárních rovnic o 5 neznámých u, v, x, y, z

$u + 2v - 2x + y - z = -2$
$5u - 4v + 3y + 4z = -46$
$2u + 6v + 2x - 7y + 2z = 37$
$u + v + 2x + y + 4z = 2$
$0{,}52u + 1{,}6v - x + 2y - 1{,}04z = 4{,}7$

Pro řešení soustavy lineárních rovnic je určen program ***Pgm 02.*** *Předpokládá vložení matice koeficientů soustavy A a vložení vektoru koeficientů pravé strany soustavy rovnic* $\vec{b}$*. Program pak spočítá determinant soustavy, inverzní matici* A^{-1} *a vektor řešení* $\vec{x}$ *jako* $\vec{x} = \vec{b} \cdot A^{-1}$*.*

Vypišme tedy matici soustavy (opíšeme koeficienty u neznámých) A a vektor $\vec{B}$*.*

$$A = \begin{vmatrix} 1 & 2 & -2 & 1 & -1 \\ 5 & -4 & 0 & 3 & 4 \\ 2 & 6 & 2 & -7 & 2 \\ 1 & 1 & 2 & 1 & 4 \\ 0{,}52 & 1{,}6 & -1 & 2 & -1{,}04 \end{vmatrix} \quad \vec{b} = \begin{vmatrix} -2 \\ -46 \\ 37 \\ 2 \\ 4{,}7 \end{vmatrix}$$

Předpokládáme, že pro n neznámých máme n lineárních rovnic, rozměry matice jsou tedy n x n, v našem případě tedy 5 × 5. Postup výpočtu: 50 datových pamětí potřebných k výpočtu

... ***5 Op 17*** *na displeji 79,49, tj. 79 programových kroků, 49 + 1 datová paměť*

Zvol program ***Pgm 02***

Vlož počet neznámých ***5 A***

Stiskni postupně ***1 B***

Vkládej prvky matice soustavy po sloupcích:

1 R/S 5 R/S 2 R/S 1 R/S 0,52 R/S 2 R/S až 4 RS 1,04 +/− RS

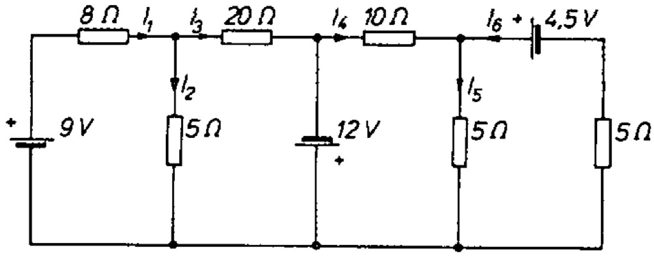
Stiskni tlačítko C ... spočítá se determinant soustavy 618,6,

stiskni postupně ***1 D***

Vkládej prvky vektoru $\vec{b}$: ***2 +/− R/S 46 +/− R/S 37 R/S 2 R/S 4,7 R/S***
Stiskni ***CLR E****... probíhá řešení soustavy... na displeji 1,*
stiskni A' a čti výsledné řešení: ***R/S*** *$u = -1,2$* ***R/S*** *$v = 5,4$* ***R/S*** *$x = 8,1$* ***R/S*** *$y = 0$* ***R/S*** *$z = -4,6$*
Soustava má tedy řešení $\vec{x} \doteq [-1,2; 5,4; 8,1; 0; -4,6]$.
Pro y čteme na displeji $5,4117647.10^{-12}$, zaokrouhlujeme na 0.

Cvičení

22. Podle programu z příkladu 7.1 řeš následující kvadratické rovnice:
    a) $x^2 - x - 12 = 0$
    b) $25x^2 + 50x - 100,44 = 0$
    c) $x^2 - 3,2x + 8,81 = 0$
23. Podle příkladu 7.2 řeš rovnice:
    a) $\sin x - x + 5,95892 = 0$
    b) $x - \operatorname{tg} x + 1,76 = 0$
    c) $x^{\sin x} + \ln^2 x - 2\sqrt{x} + \cos x + 0,793 = 0$
24. Podle příkladu 7.3 řeš soustavy lineárních rovnic:
    a) $2,167x - 5,234y = 1,456$
       $5,324x + 8,324y = -0,234$
    b) $2x - 3y + z = 5$
       $7x + y - z = 2$
       $x + 5y - 2z = 1$
25. Vypočítej proudy I_1, I_2, I_3, I_4, I_5, I_6 obvodu na obrázku 17. Návod: Podle Kirchhoffových zákonů sestav soustavu 6 rovnic o 6 neznámých.



Obr. 17

8. Hry s kalkulátorem

V příkladech této kapitoly několikrát využijeme tzv. generátoru náhodných čísel. Vestavěný program **Pgm 15** generuje po instrukci **SBR D.MS** určité číslo z intervalu (0; 1). Aby se tato čísla neopakovala při novém použití tohoto programu, vložíme na počátku výpočtu do registru R9 zdrojové číslo Z, které pak slouží jako výchozí při určování náhodného čísla. Vyzkoušejme si tento program:
Vol program tlačítky **Pgm 15.**
Vlož zdrojové číslo 234567
STO 09 . . . SBR D.MS na displeji 0,8045
SBR D.MS na displeji 0,46599
SBR D.MS na displeji 0,22463 atd.

Příklad 8.1.

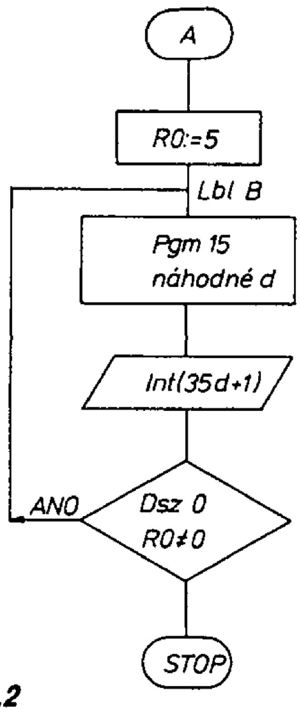
Sestav program nahrazující hrací kostku, tj. program, který bude náhodně vytvářet čísla 1, 2, 3, 4, 5, 6.

Řešení

Z náhodně utvořeného desetinného čísla d utvořme číslo k = Int (6d + 1). Toto číslo nabývá pouze hodnot 1, 2, 3, 4, 5, 6, můžeme je tedy považovat za výsledek házení kostkou. „Házení" nahradíme stlačením tlačítka A:

Lbl A Pgm 15 SBR D.MS × 6 + 1 = Int R/S

Vložme zdrojové číslo např. 5555 STO 9 a házejme: A na displeji 5, A ... 4, A ... 5, A ... 3, A ... 2, A ... 2, A ... 5, A ... 3, A ... 1, A ... 2, A ... 6, atd.



Obr. 18

Příklad 8.2

Sestav program nahrazující losování 5 čísel MATESA.

Řešení

Protože v MATESU losujeme čísla od 1 do 35, utvoříme z náhodného desetinného čísla číslo m = Int (35d + 1). Toto „tažené" číslo zobrazíme na displeji instrukcí Pause a losování opakujeme celkem pětkrát ve smyčce řízení instrukcí ***Dsz 0*** *(obr. 18). Před začátkem prvého losování opět vložíme zdrojové číslo do R9, abychom dostali jinou pětici čísel.*

Lbl A 5 STO 0 Lbl B Pgm 15 SBR D.MS × 35 + 1 = Int Pause Dsz 0 B R/S

Do osudí vložíme zdrojové číslo, např. 978654 STO 09 a losujeme:

***A** . . . na displeji se postupně objeví čísla*
9 21 7 31 14

Druhý tah:

***A** . . . 19 7 32 34 4.*

Při úpravě programu pro SPORTKU nahradíme číslo 5 číslem 6 a číslo 35 číslem 49. Jinou možností použití by bylo např. losování lístků v tombole.

Příklad 8.3

*Hádání tajného čísla HI-LO (High-Low . . . vysoko-nízko) je hra zabudovaná jako program **Pgm 21**. Může se hrát ve dvou variantách:*

I. Hádá operátor, číslo si myslí kalkulátor přirozené z intervalu <1;1023>)

Postup

***Pgm 21** . . . zdrojové číslo . . . **A B** na displeji 0, vlož číslo, které hádáš **C** . . . na displeji*
−1 . . . příliš nízké,
1 . . . příliš vysoké,
bliká 0 . . . uhádl jsi,
v případě neuhádnutí poslední krok opakuj.

*Po ukončení hry . . . **D** . . . na displeji počet pokusů.*

II. Hádá kalkulátor

Mysli si číslo z intervalu <1; 1023>.

***A'** . . . na displeji odhad kalkulátoru:*
*je-li nízký . . . **B'**,*
*je-li vysoký . . . **C'**.*
kalkulátor hádá znovu.

*Po ukončení hry . . . **D'** . . . na displeji počet pokusů kalkulátorů. Vypiš si z programové paměti program této hry **(Pgm 15 Op 09 LRN)** a podle výpisu sestroj vývojový diagram. Poznáš tak strategii kalkulátoru.*

Příklad 8.4 – POSLEDNÍ VYHRÁVÁ

*Jsou dána dvě přirozená čísla **a** a **d<a**. Od čísla **a** dva hráči střídavě odčítají číslo z intervalu <1; **d**> a oznámí výsledek. Vyhrává ten, kdo dříve dosáhne nuly.*

Program (bez odvození a odůvodnění): **Lbl D CLR RCL 0 STO 1 GTO STO Lbl C CLR RCL 0 STO 1 Lbl E' RCL 1 – R/S STO 7 CP x = t lnx = x⇄t x ≧ t E' RCL 2 INV x ≧ t E' 0 x⇄t INV Int INV x = t E' RCL 7 STO 1 Lbl STO RCL 1: (RCL 2 + 1) = STO 7 INV Int x = t C' RCL 2 + 1 = × RCL 7 Int = STO 1 INV x = t E' Op 23 R/S Lbl C' RCL 1 – (1 + (Pgm 15 SBR D.MS x RCL 2) Int = STO 1 GTO E' Lbl lnx RCL 1 INV x ≧ t E' Op24 0 R/S Lbl A STO 0 R/S Lbl B STO 2 R/S**

Postup:

*Vlož počáteční hodnotu **a** (např. 100) . . . **A***
*Vlož maximální číslo, které budeš odečítat **d** (např. 12) . . . **B***
*Chceš začínat sám? . . . **C***
*Chceš, aby začínal kalkulátor? . . . **D***

*Odečti (zpaměti) od čísla na displeji číslo z intervalu <1; d>, vlož, stiskni **R/S**.*

*Poslední krok opakuj tak dlouho, pokud nevyhraješ (nevložíš na displej 0) nebo pokud nevyhraje kalkulátor. Pak můžeš znovu volit variantu **B** nebo **C**.*

Pokud vložíš číslo, které nevyhovuje podmínkám úlohy, zadání se automaticky opakuje.

Výsledky cvičení

1. a) 5 − 3 x 40 sin = . . . 3,071637171
   b) 2 + 5 x 0,87 lnx : 10 tg = −1,948972153
   c) 3 x 4 INV lnx − 50 tg ln = . . . 163,6190243
2. a) a − b x (a : b) Int = . . . 4
   b) . . . 21
   c) . . 0
3. a) $ax^2 + bx^2 - 2xax\,\gamma\cos = \sqrt{x}$ . . . 37
   b) . . . 19
4.

$$v = 1225 \cdot \sqrt{5 \cdot \{\{[(0{,}2 \cdot M^2 + 1)^{\frac{1}{0{,}286}} - 1] \cdot [1 - 2{,}256 \cdot 10^{-5} \cdot h]^{5{,}2656} + 1\}^{\frac{1}{3{,}5}} - 1]}$$

výpočet
1225 x (5 x ((((. 2 x M x^2 + 1) INV y^x . 286 − 1) x (1 − 2.256 EE 5 +/− x h) y^x 5.2656 + 1) INV y^x 3.5 − 1)) $\sqrt{x}$ = výsledek 679 km/hod

5.

$$h = [1 - \{[(0{,}2 \cdot M^2 + 1)^{\frac{1}{0{,}286}} - 1] : [(1 + 0{,}2\left[\frac{v}{1225}\right]^2)^{3{,}5} - 1]\}^{\frac{1}{-5{,}2656}}] : 2{,}256 \cdot 10^{-5}$$

výpočet:
(1 − (((2 x M x^2 + 1) INV y^x . 286 − 1) : ((1 + . 2 x (v : 1225) x^2) y^x 3.5 − 1)) INV y^x 5.2656 +/−) : 2.256 EE 5 +/− = výsledek 14 950 m

6. a) a_0x (1 + p : 100) y^xn = výsledek 2293,48

b) $$n = \frac{\ln \frac{a_n}{a_o}}{\ln\left(1 + \frac{p}{100}\right)}$$

výpočet:
$a_n : a_o$ = lnx : (1 + p : 100) lnx = výsledek 169

c) $a_0 = \dfrac{a_n}{(1 + \frac{p}{100})^n}$

výpočet:
a_n : (1 + p : 100) y^x n = výsledek 12,63

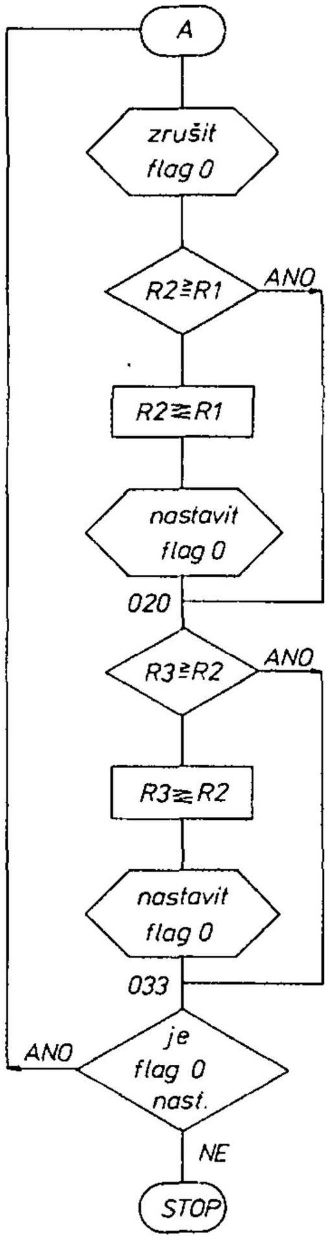
d) $p = 100 \cdot [(\frac{a_n}{a})^{\frac{1}{n}} - 1]$

výpočet:
100 x ((a_n : a_0) INV y^x n − 1) = výsledek 19,62

7. **Lbl A x 2 x π = R/S**
8. **Lbl B x^2 x π = R/S**
9. **Lbl C STO 0 x 2 x π = Pause RCL 0 x^2 xπ R/S**
10. **Lbl A STO 1 R/S Lbl B STO 2 R/S Lbl C RCL 1 : RCL 2 = R/S Lbl D RCL 1 x RCL 2 = R/S**
11. **Lbl A STO 1 R/S Lbl B STO 2 R/S Lbl C STO 3 R/S Lbl A' RCL 3 : RCL 1 : RCL 2 = Fix 2 R/S Lbl B' RCL 1 x RCL 2 = R/S Lbl C' RCL 3 : RCL 1 : RCL 2 = INV cos sin x RCL 1 x RCL 2 = R/S Lbl D' RCL 3 : RCL 1 = R/S Lbl E' RCL 3 : RCL 1 : RCL 2 = INV cos sin x RCL 2 = R/S**
12. **Lbl A STO 1 R/S Lbl B STO 2 R/S Lbl C' STO 3 R/S Lbl C RCL 1 x^2 + RCL 2 x^2 − 2 x RCL 1 x RCL 2 x RCL 3 cos = √x R/S**
13. **Lbl A STO 2 x⇄t STO 1 5 x ((((1 + .2 x (RCL 1 : 1225) x^2) y^x 3.5 − 1) x (1 − 2.256 EE 5 + / − x RCL 2) y^x 5.2656 +/− + 1) y^x .286 − 1) = √x R/S**
Vypočítané Machovo číslo v posledním řádku tabulky:
0,83 1,09 0,77 0,91 1,29 2,08 0,17 0,2 2 1,00
Obdobně bychom mohli utvořit program podle cvičení 4 a 5.
14. Hodnotu x předpokládáme na displeji:
Lbl A x 4 − 5 = R/S Lbl B STO 0 x^2 x 3 − 5 x RCL 0 + 1 = R/S Lbl C STO 00 x x^2 x^2 x RCL 0 x^2 + / − + 3 x (RCL 0 x x^2) x^2 − 4 x (RCL 0 x x^2) + 5 x RCL 0 x^2 − RCL 0 + 7 = R/S Lbl D − Rad sin − 1.9 = R/S Lbl E − Rad tg + .01 = + / − R/S
15. $s = \dfrac{a + b + c}{2}$, $S = \sqrt{s \cdot (s-a) \cdot (s-b) \cdot (s-c)}$,
$\varrho = P/s$, $r = abc/4P$, $\operatorname{tg} \dfrac{\alpha}{2} = \dfrac{\varrho}{s-a}$, atd.
Strany a, b, c uložíme programy A, B, C do registrů R1, R2, R3. Program D uloží s do R0 a S do R4. ϱ spočítáme programem E a uložíme do R5. Pro výpočet r zvolíme program E, úhly α, β, γ spočítáme programy A', B', C':

LBL A STO 1 R/S Lbl B STO 2 R/S Lbl C STO 3 R/S Lbl D RCL 1 + RCL 2 + RCL 3 = : 2 = STO 0 x (CE − RCL 1) x (RCL 0 − RCL 2) x (RCL 0 − RCL 3) = √x STO 4 R/S Lbl E RCL 4 : RCL 0 = STO 5 R/S Lbl E' RCL 1 x RCL 2 x RCL 3 : 4 : RCL 4 = R/S Lbl A' RCL 1 GTO D' Lbl B' RCL 2 GTO D' Lbl C' RCL 3 Lbl D' STO 6 RCL 5 : (RCL 0 − RCL 6) = INV tg x 2 = INV D.MS Fix 4 R/S

Vyzkoušejte pro a = 41, b = 50, c = 89:
41 A 50 B 89 C
D na displeji S = 420
E na displeji ϱ = 4,67
E' na displeji r = 108.6
A' α = 10° 52′ 50″,
B' β = 13° 18′32″,
C γ = 155° 48′ 38″



Obr. 19

16. Program spočívá v srovnání velikosti 2 čísel v sousedních registrech. Zjišťujeme, zda platí R2 ≧ R1 a R3 ≧ R2. Pokud některý z těchto vztahů neplatí, zaměníme obsahy příslušných dvou registrů. Proberme si podrobněji jednotlivé případy, máme-li např. uspořádat čísla 3, 4, 5. Jsou-li v R1, R2, R3 čísla
1. 3, 4, 5 . . ., zjistíme, že R2 ≧ R1 i R3 ≧ R2 a výpočet skončí;
2. 3, 5, 4 . . ., přemístíme na 3, 4, 5 a výpočet skončí;
3. 4, 3, 5 . . ., *přemístíme na* 3, 4, 5 *a výpočet* skončí;
4. 4, 5, 3 . . ., přemístíme na 4, 3, 5 a znovu na 3, 4, 5 a výpočet skončí;
5. 5, 3, 4 . . ., přemístíme na 3, 5, 4 a znovu na 3, 4, 5 a výpočet skončí;
6. 5, 4, 3 . . ., přemístíme na 4, 5, 3, pak na 4, 3, 5 a 3, 4, 5.

Aby program obsáhl všech 6 možností, nastavíme v případě přemísťování vždy příznak „flag 0" instrukcí St flg 0, po návěští Lbl A jej vždy zrušíme instrukcí INV St flg 0. Poslední rozhodovací operace „je příznak flag 0 nastaven" – instrukce If flg 0 zajistí, zda má program znovu proběhnout od návěští Lbl A nebo zda je uspořádání u konce. V případě 1. (3, 4, 5) proběhne program od Lbl A pouze jednou – příznak nebude nastaven. V případech 2, 3 a 5 proběhne program dvakrát ve zbývajících možnostech třikrát. Podle vývojového diagramu (obr. 19) *můžeme tedy sestavit* následující program:

Lbl A INV St flg 0 RCL1 x⇄t RCL2 x≧t 020 STO 1 x⇄t STO 2 St flg 0 x⇄t RCL 3 x≧t 033 STO 2 x⇄t STO 3 St flg 0 If flg 0 A CLR R/S.

Vyzkoušej tento program. Do registrů 1 až 3 vlož různá čísla, stiskni tlačítko A, po chvíli se na displeji objeví 0 a zkontroluj uspořádání čísel v R1 až R3.

Program pro uspořádání *n* čísel uložených v registrech R1 až Rn si uvedeme bez vysvětlení. Počet čísel *n* je třeba nejdříve uložit do R49, výpočet pak proběhne od návěští A:

Lbl A INV St flg 0 RCL 49 STO 0 RCL Ind 0 x⇄t Dsz 0 020 If St flg 0 A R/S RCL Ind 0 x⇄t x≧t 012 STO Ind 0 x⇄t Op 20 STO Ind 0 Op 30 St flg 0 GTO 012 Lbl B RCL 49 STO C RCL Ind 0 Pause Dsz 0 046 R/S

V programu je použito nepřímého adresování řízeného registrem R0. Programem B si postupně zobrazíme na displeji čísla z Rn až R1.

17. Lbl B A RCL 3 x RCL 4 : RCL 1 = R/S Lbl A . .
stejný jako v př. 6.4. n (18: 72) = 72, n (24335; 41378) = 324580230n (484909: 216775) = 157595425

19. Lbl A x RCL 1 = Pause GTO A
Chceme-li pouze prvých n-členů, vložíme n-1 do R0 a program upravíme na
Lbl A x RCL 1 = Pause Dsz 0 A R/S.

20. Lbl A 1 STO 1 STO 2 Lbl B RCL 1 : RCL 2 = Pause RCL 1 Exc 2 STO 1 SUM 2 GTO B

21. Lbl A 101 x⇄t 1 STO 1 Lbl B Fix 0 Pause √x Fix 8 Pause Op 21 RCL 1 INV x≧t B CLR R/S

22. a/ $x_1 = 4$, $x_2 = -3$ b) $x_1 = 1{,}24$, $x_2 = -3{,}24$
c) $x_1 = 1{,}6 + 2{,}5i$, $x_2 = 1{,}6 - 2{,}5i$

23. a) Lbl A' (CE – Rad sin – 5,95892) +/– INV SBR
Pgm 08 0 A 10 B 1 C 0,00001 D E . . . x = 5,00000
jediný kořen v <0; 10>
b) Lbl A' (CE – Rad tg + 1.76) INV SBR
Pgm 08 0 A 1,5 B 0,5 C 0,001 D E . . . x = 1,25
c) Lbl A' (STO 0 y^x Rad sin + RCL 0 lnx x^2 – 2 x RCL 0 √x + RCL 0 cos + 0,793) INV SBR
Pgm 08 0,01 A 20 B 0,5 C 0,0001 D E $x_1 = 1{,}802$
$x_2 = 5{,}485$
$x_3 = 9{,}298$
$x_4 = 10{,}996$
$x_5 = 15{,}969$
$x_6 = 16{,}567$
Vypočítané kořeny jsou z intervalu (0;20).

24. a) Pgm 02 2 A 1 B 2,167 R/S, 5,324 R/S 5,234 +/– R/S 8,324 R/S C 1 D 1,456 R/S 0,234 +/– R/S CLR E A'
R/S x = 0,2373
R/S y = –0,1799

b) Pgm 02 3 A 1 B 2 R/S 7 R/S 1 R/S 3 +/– R/S 1 R/S 5 R/S 1 R/S 1 +/– R/S 2 +/– R/S 1 D 5 R/S 2 R/S 1 R/S CLR E A' R/S x = 15
R/S y = 64
R/S z = 167

25. Jedno z možných řešení:
$8I_1 + 5I_2 = 9$
$-5I_2 + 20I_3 = 12$
$-10I_4 - 5I_5 = 12$
$5I_5 + 5I_6 - 4{,}5$
$I_1 - I_2 - I_3 = 0$
$I_4 - I_5 + I_6 = 0$

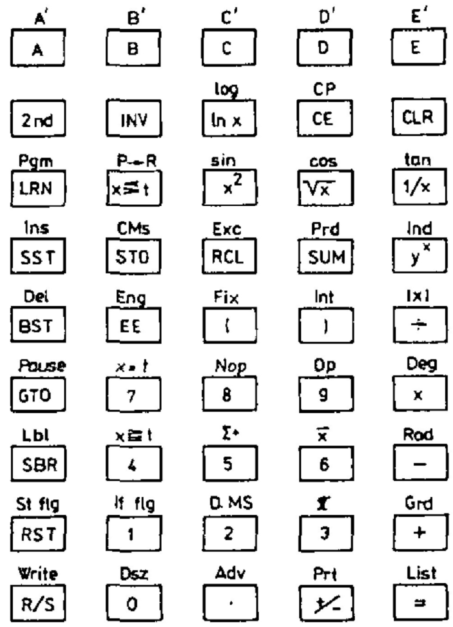
Matice soustavy:

$$M = \begin{vmatrix} 8 & 5 & 0 & 0 & 0 & 0 \\ 0 & -5 & 20 & 0 & 0 & 0 \\ 0 & 0 & 0 & -10 & -5 & 0 \\ 0 & 0 & 0 & 0 & 5 & 5 \\ 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 1 \end{vmatrix} \qquad b = \begin{vmatrix} 9 \\ 12 \\ 12 \\ 4{,}5 \\ 0 \\ 0 \end{vmatrix}$$

6 Op 17 . . . na displeji 0,59 . . . k dispozici je 60 datových pamětí
Pgm 02 6 A 1 B 8 R/S 0 R/S . . . vkládáme po sloupcích prvky matice M . . . 0 R/S

C . . . na displeji determinant soustavy
1 D 9 R/S 12 R/S . . . vkládáme prvky vektoru b . . . 0 R/S
CLR E . . . probíhá výpočet
A'R/S I_1 = 0,95A R/S I_2 = 0,28 A R/S I_3 = 0,67A R/S I_4 = –1,14A R/S I_5 = –0,12A R/S I_6 = 1,02A

Rozmístění tlačítek na kalkulátoru TI 58/59



Na závěr uvádíme tabulku některých matematických a organizačních částí programu, které můžeme „beztrestně" použít ve svém programu, aniž bychom se vůbec starali o to, co vyvolaný program jako celek dělá. V rubrice „funkce" je instrukce, kterou požadovaný úkon vyvoláme. V ostatních rubrikách je stav po proběhnutí příslušné funkce. Předpokládá se, že před začátkem je na displeji X, v R1 je číslo R1, v R2 číslo R2 atd. Rubrika „d = X" značí stav na displeji. Je-li v tabulce prázdné políčko, znamená to, že po proběhnutí vyvolané funkce zůstal zapsán původní výraz. V rubrice „zvláštní funkce" jsou některé další stavy a příkazy, které příslušná funkce vyvolá. V neoznačené rubrice je vysvětlení matematických výrazů, které jsou naznačeny pomocí a, b.

Tabulku nám zaslal Jiří Pobříslo, autor článku „Optimalizace programu nejsou také žádné čáry" z AR12/82.

Funkce	d = x	R1	R2	R3	R4	t	Zvláštní funkce		
Pgm 11 E'	x						INV STFLG 0, 1, 2, 3		
Pgm 19 E'	0					0	INV STFLG 0, 1, 2, 3, 4 R5 = 0 INV FIX		
Pgm 1 SBR CLR	0	0	0	0	0	0	R5 = 0 R6 = 0		
Pgm 1 SBR CE	0	0	0	0	0	0	R5 = 0 R6 = 0 R9 = 0 INV FIX DEG		
Pgm 4 A	x	R2	x			0	RAD CE		
Pgm 4 A'	x			R4	x		CE		
Pgm 4 E'	0	R3	R4	R1	R2				
Pgm 20 C	a							a = R5 – R4	
Pgm 13 C'	a							a = R1 . R2	
Pgm 9 E	a							a = R1 + (R3 . R5)	
Pgm 9 SBR 0 38	a			a				$a = \frac{R2 - R1}{R5}$	
Pgm 4 C	a	a	b			b		a = R1 . R3 – R2 . R4 b = R1 . R4 + R2 . R3	
Pgm 5 B	a					b	RAD	$a = \sqrt{(R1)^2 + (R2)^2}$ $b = \text{arctg}\,\frac{R2}{R1}$	
Pgm 18 SBR CE	a				a			$a = R3 \cdot (R9)^{R1}$	
Pgm 6 A	a						RAD	$a = \sqrt{(R1 + 1)^2 + (R2)^2}$	
Pgm 6 E'	a							$a = \sqrt{(R1 - 1)^2 + (R2)^2}$	