Počítač nejenom počítá, ale taky se opakuje, rozhoduje …

Použití základních prostředků pro řízení běhu programu. Schematický zápis sekvenčního postupu řešení úlohy.

Úlohy

Schémata

1. Zapište grafické schéma v podobě vývojového diagramu pro řešení (některých z) níže uvedených úloh.

Rozhodování, větvení

2. Napište program, který pro dvě zadaná celá čísla určí a vypíše, zda první číslo je dělitelné druhým zadaným číslem.
3. Napište program, který načte souřadnice bodu a parametry kružnice (souřadnice středu a poloměr) a určí vzájemnou pozici bodu a kružnice.
4. Napište program, který načte délky tří úseček a zjistí a vypíše informaci o tom, zda lze sestrojit trojúhelník s odpovídajícími délkami stran. Pomůcka: „trojúhelníkové nerovnosti“: $(a + b > c)$ a zároveň $(|a – b| < c)$.
5. Napište program, který bude řešit polynomiální rovnice maximálně druhého řádu $ax^2 + bx + c$ (tj. rovnice lineární, kvadratické v reálném i komplexním oboru).
6. Zapište program, který na vstupu načte tři číselné hodnoty a vypíše je v sestupném pořadí, popřípadě v sestupném či vzestupném pořadí dle požadavku uživatele.
7. Napište program, který načte souřadnice bodu roviny a určí a vypíše, ve kterém kvadrantu zadaný bod leží.
8. Napište program, který načte délku tří úseček jako potenciálních délek stran trojúhelníka. V případě, že lze zkonstruovat trojúhelník se stranami zadaných délek, nechť program vypíše, o jaký typ trojúhelníka se jedná (rovnostranný, rovnoramenný, rovnoramenný pravoúhlý, pravoúhlý, obecný). Případně doplňte program o výpočet parametrů trojúhelníka – obvodu, plochy, velikosti úhlů, délek výšek a těžnic.
9. Varianta předchozí úlohy. Napište program, který načte souřadnice tří bodů v rovině a určí typ trojúhelníka, jehož vrcholy jsou tvořeny zadanými body.
10. Napište program, který načte souřadnice dvou bodů a vypíše rovnici odpovídající přímky v parametrickém, obecném a směrnicovém tvaru. Při testování pak zadejte a vyzkoušejte i přímky rovnoběžné s jednotlivými osami.
11. Zapište program, který bude určovat, zda zadaný bod leží v trojúhelníku zadaném souřadnicemi vrcholů. Úlohu řešte ve 2D.
12. Napište program, který bude řešit zadanou soustavu dvou rovnic o dvou neznámých. Pomocí podmínky je nutno vyloučit případ lineární závislosti rovnic (nejednoznačné řešení). Proveďte rozbor úlohy, stanovte základní výrazy pro existenci řešení a výpočet *x* a *y*, poté zapište kód programu.
13. Zapište program, který načte datum narození osoby a informaci o jejím pohlaví a na základě toho sestaví první část rodného čísla příslušné osoby.
14. Zapište program, který načte celé číslo představující rok, a určí, zda příslušný rok byl/je/bude přestupný a vypíše k tomu i počet dnů v daném roce. Přestupný rok je dělitelný 4 s tou výjimkou, že roky dělitelné 100 jsou přestupné pouze tehdy, pokud jsou zároveň dělitelné i 400. Přestupné tak byly například i roky 1600 a 2000. Přestupné naopak nejsou například 1700, 1800, 1900, 2100, 2200 atd.
15. Zapište program, který načte dvě celá čísla (do proměnných celočíselného typu) představující první a druhou část rodného čísla, a otestuje, zda se jedná o platné rodné číslo. Jednoduchý test na platnost rodného čísla je dělitelnost 11. Rodná čísla jsou zpravidla dělitelná číslem 11. Nicméně, toto neplatí pro všechna přidělená rodná

čísla. Ve skutečnosti je algoritmus přidělování rodných čísel (čerpáno z https://phpfashion.com/jak-overit-platne-ic-a-rodne-cislo) následující: (1) vypočti zbytek po dělení prvních devíti číslic a čísla 11; (2) je-li zbytek 10, poslední číslice musí být 0; (3) jinak je poslední číslice rovna zbytku. 780123/3540 je tak platné rodné číslo, přestože není dělitelné 11. Pravidla pro rodná čísla též například na http://lorenc.info/3MA381/overeni-spravnosti-rodneho-cisla.htm. V kontextu této skupiny úloh realizujte program s využitím základních prostředků pro rozhodování (if) a výpočtu výrazů bez využití cyklů, byť by se použití cyklu v tomto případě nabízelo.

16. Zapište program, který načte celé číslo a ověří, zda se jedná o platné IČ. Maximální délka IČ je 8, pokud je kratší, úvodní nuly není nutno zadávat. Pro ověření platnosti proveďte: (1) první až sedmou číslici vynásobte postupně čísly 8, 7, 6, 5, 4, 3, 2 a součiny sečtěte; (2) zjistěte zbytek po vydělení vzniklého součtu jedenácti; (3) pro poslední číslici musí platit: je-li zbytek 0 poslední číslice je 1, je-li zbytek 1 poslední číslice je 0, v ostatních případech je poslední číslice rovna 11-zbytek. Zadané IČ načtěte do proměnné celočíselného typu. V kontextu této skupiny úloh realizujte požadovaný program bez využití cyklů, byť by se použití cyklu v tomto případě nabízelo.

Opakování, cykly – `while`, `do-while`, `(for)`

17. Vyberte jednu z dříve zadaných úloh (například určení dělitelnosti nebo převod sekund na hodiny, minuty, sekundy) a zapište program, který bude danou úlohu řešit opakovaně pro více vstupních hodnot při jednom spuštění programu. Ukončení programu bude specifikováno jistou kombinací uživatelem zadávaných vstupních hodnot nebo jako odpověď a/n na „dotaz položený programem“.
18. Zapište program, který bude zpracovávat sadu zadaných kladných čísel. Program má postupně zjistit počet a součet a následně i průměr zadaných čísel. Zadávání bude ukončeno uživatelem zadáním nulové nebo záporné hodnoty. Jaké jsou rozdíly při výpočtu průměru z celých nebo reálných čísel?
19. Zapište program pro výpočet hodnoty faktoriálu čísla. Pro jaké rozsahy čísel můžeme vyčíslit faktoriál v případě, že výsledek budeme uchovávat jako hodnotu typu `short`, `int`, `long`? Je vhodné použít některý z reálných typů?
20. Zapište program pro výpočet celočíselné mocniny zadaného čísla (celého, reálného) pomocí opakovaného součinu.
21. Zapište program, který vypíše všechny kladné dělitele zadaného čísla (které potenciální dělitele musíme testovat – 2 .. n-1, 2 .. n/2). Dále upravte program tak, aby pro každé zadané číslo vypsal jeho dělitele – program má potom ukončit činnost v případě, že na vstupu ke zpracování obdrží číslo 0 nebo. Doplňte určení počtu dělitelů.
22. Zapište program, který bude určovat, zda zadané číslo je prvočíslo. Porovnání případu, kdy budeme testovat všechny potenciální dělitele nebo ukončíme cyklus při nalezení prvního dělitele (předčasné ukončení cyklu zajistit podmínkou v záhlaví příkazu `while` a nikoli příkazem `break`). Modifikace pro opakované zadávání čísel.
23. Zapište program, který každé zadané číslo vypíše jako součin prvočísel. Program nechť skončí po zadání hodnoty 0 nebo záporného čísla na vstupu.
24. Zapište program pro výpočet největšího společného dělitele *nsd* dvou zadaných čísel – porovnat různé varianty řešení – 1) prosté testování všech potenciálních hodnot (procházení zdola nebo shora, 2) s využitím skutečnosti, že rozdíl obou čísel a libovolné z nich mají stejného největšího společného dělitele jako původní dvě čísla, 3) využití *nsd* menšího z čísel a zbytku po celočíselném dělení obou čísel (Eukleidův algoritmus). Doplňte program o výpočet nejmenšího společného násobku *nsn* obou čísel – využijte vypočtený *nsd*.
25. Zapište program, který pro každé zadané kladné číslo vypíše jeho ciferný součet a ciferný součin. Program nechť skončí po zadání záporné hodnoty nebo 0 na vstupu.
26. Zapište program, který bude převádět čísla mezi různých číselnými soustavami.
27. Zapište program, který načte osmimístné celé číslo a ověří, zda se jedná o platné IČ. Pro ověření platnosti proveďte: (1) první až sedmou číslici vynásobte postupně čísly 8, 7, 6, 5, 4, 3, 2 a součiny sečtěte; (2) zjistěte

zbytek po vydělení vzniklého součtu jedenácti; (3) pro poslední číslici musí platit: je-li zbytek 0 poslední číslice je 1, je-li zbytek 1 poslední číslice je 0, v ostatních případech je poslední číslice rovna 11-zbytek.

28. Napište program, který pro každé zadané kladné číslo vypíše informaci o tom, zda se jedná o číslo dokonalé. *Dokonalé číslo* je takové přirozené číslo, které je rovné součtu všech svých kladných dělitelů (včetně 1, kromě čísla samého). Dokonalá čísla jsou například 6 a 28: 6 = 1 + 2 + 3; 28 = 1 + 2 + 4 + 7 + 14. Další dokonalá čísla jsou i: 496, 8128, 33550336, 8589869056, 137438691328.

29. Zapište program, který pro dvě zadaná přirozená čísla rozhodne, zda se jedná o čísla spřátelená. Přirozená čísla a, b nazveme *spřátelená*, jestliže součet kladných dělitelů (včetně 1, kromě čísla samotného) každého z nich je roven druhému z čísel.

    První a nejmenší dvojici spřátelených čísel tvoří čísla 220 a 284.

    Vlastní dělitele čísla 220 jsou: 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110.

    Vlastní dělitele čísla 284 jsou: 1, 2, 4, 71 a 142

    Přitom platí: 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284; 1 + 2 + 4 + 71 + 142 = 220

    Další dvojice spřátelených čísel jsou např.: 1184 a 1210, 17296 a 18416, 9363584 a 9437056

30. Zapište program pro opakovaný výpočet úsekové rychlosti (viz zadání jedné z úloh na výpočty reálných výrazů) vozidel s vyhodnocením závažnosti případného přestupku a stanovením výše případné sankce.

31. Zapište program pro ověření platnosti rodného čísla nebo IČ s využitím cyklu.

Opakování, cykly – for

32. Zapište cyklus, který zajistí výpis N hvězdiček na jednom řádku výstupu (konzole), jednotlivé * mají být od sebe odděleny právě jednou mezerou. Použijte cyklus `for`

33. Zapište program, který načte celé číslo N a program vypíše „čtverec z '*' " velikosti N (případně obdélník velikosti NxM), ve tvaru (pro N = 3):

```
* * *
* * *
* * *
```

    Realizace vede na použití cyklů. Použijte dva vnořené cykly `for` – vnějším cyklem iterujeme přes všechny řádky, vnitřním cyklem iterujeme přes vypisované '*' na daném řádku.

34. Zapište program, kterému zadáte celé číslo N a program vypíše na konzolovou obrazovku „trojúhelník z '*' " výšky N, ve tvaru (pro N = 3):

```
*
* *
* * *
```

    Pomocí parametru N vyjádřete, kolik řádků má program vypsat, z pořadí řádku vyjádřete počet'*' na řádku.

35. Úloha je variantou předešlé. Zapište program, kterému zadáte celé číslo N a program vypíše na konzolovou obrazovku „trojúhelník z '*' " výšky N ve tvaru:

```
   *
  * *
 * * *
```

    Zde pomocí indexu vypisovaného řádku obecně vyjádřete, kolik mezer má být vypsáno před první '*' na daném řádku. Vnějšímu příkazu cyklu budou vnořené dva sekvenčně zapsané příkazy cyklu – první zajistí výpis požadovaného počtu mezer na řádku, druhý zajistí výpis požadovaného počtu sekvencí '* '. Poté lze ukončit aktuální řádek výstupu.

36. Předchozí úlohy lze variovat – výpis „kosočtverce", opakovaný výpis zadaného tvaru apod.

37. Zapište program pro „textový výpis vlajky CR" zadané velikosti.

Opakování, rozhodování, vícenásobné větvení

38. Asi znáte hru „Hádání čísla“. Jedná se o hru dvou hráčů. Jeden z hráčů si myslí číslo v zadaném rozsahu (rozsah může být pevně stanovený, nebo to může být hodnota zadávaná uživatelem po spuštění programu). Druhý hráč se opakovaně snaží uhodnout myšlené číslo. Na každý jeho tip první hráč odpovídá jedním z následujících způsobů: „Uhodl jsi“, „Myšlené číslo je menší“, „Myšlené číslo je větší“. Hra končí v případě, že druhý hráč uhodne myšlené číslo.

    Vytvořte program, který bude realizovat hru „Hádání čísla“ a to tak, aby si uživatel mohl zvolit rozsah „myšlených/hádaných čísel“, roli ve které se bude hry účastnit. Při jednom spuštění programu umožněte hraní libovolného počtu her (dle volby uživatele). Program má při jednotlivé hře počítat počet pokusů. Program má s uživatelem komunikovat formou řádkového menu. Program by mohl svoji činnost vykonávat následujícím způsobem – vypíše jednoduché menu, uživatel si zvolí akci, nastavení rozsahu čísel, hraní hry v roli prvního hráče, hraní hry v roli druhého hráče nebo ukončení hry, pokud uživatel zvolil akci, akce se provede (např. je zahájena a realizována příslušná hra do ukončení), poté program opět vypíše menu – atd.

    Pokud počítač bude zastávat roli hráče, který hádá číslo, potom algoritmus musí zvolit vhodnou strategii, podle které bude program dávat jednotlivé tipy (náhodné generování tipů asi předem zavrhneme) – ne příliš vhodnou strategií je postupné procházení všech hodnot v zadaném rozsahu, na místě je asi použití metody půlení intervalu.

39. Odlaďte následující program, pro výpočet zadané aritmetické operace mezi dvěma čísly:

```
float a, b, c;
char zn;
a = sc.nextFloat();
zn = sc.next().charAt(0);
b = sc.nextFloat();
switch (zn) {
  case '+' : c = a + b; break;
  case '-' : c = a - b; break;
  case '*' : c = a * b; break;
  case '/' : c = a / b; break;
  default : c = Float.NaN;
}
System.out.println(c);
```

    Program v uvedeném schématu odlaďte a vyzkoušejte.

    Proveďte obměnu předešlého programu tak, aby fungoval jako jednoduchý kalkulátor – program má opakovaně načítat střídavě číselné hodnoty a aritmetické operace tak dlouho, dokud načtenou aritmetickou operací není znak =, poté se má vypsat výsledek. V první verzi realizujte primitivní kalkulátor, který nerespektuje prioritu operací.

    V další verzi implementujte primitivní kalkulátor z předchozí úlohy tak, aby respektoval prioritu operací, popřípadě, aby respektovat zadání celého výrazu na jediném řádku vstupu s tím, že jednotlivé zadávané/načítané hodnoty (čísla, operátory) budou navzájem odděleny minimálně jednou mezerou.