Banka typových úloh pro maturitu z Programování

Gymnázium Brno, Vídeňská

David Martinek

2017/2018

1 Společné pro všechny úlohy

Odlaď program s podprogramy, který…

- Očekává se, že každá úloha bude vypracována pomocí podprogramů. To se týká i úloh, které se vám zdají být triviální a krátké.
- Podprogramy budou mít parametry a budou si přes ně vyměňovat data. Nebudou výsledky tisknout na výstup, pokud to po nich nebude explicitně požadováno. Pro tisk výsledků slouží funkce main (z níž lze volat pomocné funkce pro tisk výsledků).
- Podprogramy budou dostatečně obecné a znovupoužitelné, tj. věci, které by přirozeně měly být řešeny jako parametry, budou zadávány jako parametry a nebudou uvnitř podprogramu definovány jako konstanty. To se týká zejména rozměrů, souborů, názvů souborů atd. Takové konstanty lze použít ve funkci main, nikoli v podprogramech.

Rozdělením problému na podprogramy dáváte najevo, že

- umíte vytvářet znovupoužitelné, obecné podprogramy,
- umíte je volat,
- umíte pracovat s parametry podprogramů,
- znáte rozdíl mezi návratovou hodnotou funkce a parametry předávanými odkazem a umíte to používat,
- umíte pracovat i na větších programech, které se bez podprogramů a vhodných programátorských návyků neobejdou.

2 Úlohy

Pomocí podprogramů

- načti m×n prvků do 2D pole ze standardního vstupu, ze souboru
- vypiš výsledné pole na standardní výstup, do souboru
- zjisti počet sudých prvků v každém sloupci a zapiš jej do posledního řádku v tomto sloupci
- zjisti součet všech prvků na obvodu matice a pak všech prvků kromě obvodových
- zjisti součet prvků ve čtvrtinách matice, když každý rozměr rozdělíš celočíselným dělením 2
- zjisti, zda je matice vertikálně, či horizontálně symetrická
- vytvoř transponovanou matici

- vytvoř matici otočenou o 90, 180, 270 stupňů (zadej počet otočení o 90 stupňů parametrem)

Pomocí podprogramů

- načti ze vstupu stupeň a koeficienty polynomu, dále načti hodnotu x a vypočítej Hornerovým schématem hodnotu polynomu v daném bodě
- čti ze vstupu řadu číselných hodnot předem neznámé délky (až do konce vstupu – EOF) a vypiš posledních N hodnot, součet posledních N hodnot, vypiš posledních N hodnot v opačném pořadí
- čti vstup neznámé délky po znacích a vrať pole reprezentující histogram, poté z něj vypiš absolutní četnosti malých znaků anglické abecedy (počty znaků na vstupu)
- Vytvoř funkci realizující textovou nabídku, která bude vracet znaky A, B, K, X s významem A – Volba A, B – Volba B, K – Konec, X – Neznámá volba (bude to vracet při neočekávané volbě). Použij ji pro komunikaci s uživatelem dokud nezvolí ukončení programu.
    - Definuj pole dvojic (znak, řetězec). Definuj takové pole obsahující textovou nabídku pro uživatele s významem A – Volba A, B – Volba B, K – Konec, X – Neznámá volba. Vytvoř funkci, která podle takto zadaného pole zobrazí textovou nabídku pro uživatele a vrátí písmeno zvolené volby (musí fungovat i s pozměněným polem s nabídkou). Použij ji pro komunikaci s uživatelem dokud nezvolí ukončení programu.

Vytvoř podprogramy pro zpracování textového řetězce zadaného jako parametr

- načti řetězec ze vstupu (maximálně N znaků)
- vrať počet číslic v řetězci
- vrať indexy první a poslední nalezené číslice v řetězci (přes parametry)
- nahraď v řetězci každý výskyt znaku x znakem y (x a y jsou parametry funkce)
- skryj v řetězci každé číslo přepsáním pomocí znaku #
- vrať počet slov v řetězci
- nahraď v řetězci malá písmena za velká, nebo velká za malá
- proveď reverzi řetězce jednou pomocí rekurze a jednou pomocí cyklu s co nejmenším počtem iterací

Pomocí podprogramů zpracuj soubor se strukturovanými daty, kde na každém řádku je záznam o sopkách – název (20 znaků), výška (v metrech), zda je aktivní (ano/ne). Počet uložených záznamů není předem znám. Názvy souborů musí zadávat uživatel (dotazem nebo přes parametry příkazového řádku).

- Vytvoř funkci vracející průměrnou výšku všech ne/aktivních sopek (vybere se parametrem).
- Vypiš názvy všech sopek vyšších než hodnota zadaná parametrem. U každého názvu vypiš, zda je sopka aktivní, či ne.
- Zapiš do výstupního souboru všechny záznamy pouze o ne/aktivních sopkách (zadej parametrem). Vytvoř jeden soubor se záznamy o aktivních a druhý se záznamy o neaktivních sopkách. Výstupní soubory musí zachovávat formát vstupního souboru.

Pomocí podprogramů…

- Načítej ze vstupu hodnoty zapsané v šestnáctkové soustavě a oddělené bílými znaky a vracej jejich vyčíslenou hodnotu. V případě chyby vrať zápornou hodnotu a ukonči program. Vyčíslené hodnoty zapisuj na výstup, každou na jeden řádek.
- Vytvoř podprogram, který zapíše zadanou hodnotu na výstup ve zvolené číselné soustavě. Základ číselné soustavy a hodnota budou parametry podprogramu a zjisti je od uživatele. Počítej se základy od 2 do 36.
- Na vstupu jsou dvě celá čísla oddělená písmenem E, např. 124E-10. Načti je jako celá čísla a vrať desetinné číslo, které představují (předveď, že víš, co ten zápis představuje). V případě chyby vrať hodnotu NAN (Not A Number = 0.0/0.0). Nesmíš je načítat jako desetinné číslo. Získanou hodnotu vytiskni s přesností na 3 desetinná místa v semilogaritmickém tvaru.

Nakresli na papír syntaktický diagram

- příkazu if-else v jazyce C
- příkazu switch v jazyce C
- cyklů while, for, do-while v jazyce C
- prototypu funkce v jazyce C
- deklarace typu struktura v jazyce C

Pomocí podprogramů vytvoř program

- Je dána množina znaků Σ (zadá uživatel). Napiš funkci, která ověří, zda se na vstupu/v textovém řetězci nachází pouze věta jazyka Σ^* (případně Σ^+) a vrátí pozici prvního místa, které již do zadaného jazyka nepatří. Pozice vstupu počítej od hodnoty 1 a v případě, že celý vstup patří do tohoto jazyka, vrať nulu.
- Vytvoř logickou funkci, která zjistí, zda v zadaném textovém řetězci/na vstupu odpovídá počet levých a pravých hranatých závorek []. Funkce musí zároveň ověřit, zda jsou závorky správně zanořeny. Například toto je správně „[a]b[ ] [c[]d]e“, ale toto „a[b]c]d[e[f]g]“ správně není. Význam ostatních znaků neřeš.
- Vytvoř logickou funkci jako v minulé úloze, ale nevíc ber v úvahu i jiné typy závorek: kulaté (), složené {}, úhlové <>.

Napiš na papír

- EBNF příkazu if-else v jazyce C
- EBNF cyklů while, for, do-while v jazyce C
- gramatiku generující věty „b“, „bab“, „baab“, „baaab“, „baaaab“ atd. až do nekonečna. Do které kategorie v Chomského hierarchii patří? Proč?

Pomocí podprogramů…

- Je dáno Σ={a}, N={S, A, B, C}. Pravidlo gramatiky má tvar „řetězec->řetězec“. Napiš funkci, která ověří, zda zadaný řetězec je
  - pravidlem gramatiky typu 3
  - pravidlem gramatiky typu 2

- Napiš funkci, která bude systematicky generovat prvních N vět jazyka zadaného gramatickými pravidly S->aA, A->bB, B->cA, B->c. Zapiš tyto věty na samostatné řádky výstupu.

Pomocí podprogramů…

- Napiš funkce, které zpracují dvě 16 bitová celá dvojková čísla reprezentovaná v textových řetězcích znaky '0' a '1'. Cílem je demonstrovat, znalost toho, jak fungují logické a bitové operace, proto funkce nad textovými řetězci provedou
  - bitový součin
  - bitový součet
  - bitový exkluzivní součet (XOR)
  - bitový posun o N bitů (kladné N vpravo, záporné vlevo), vypadlé bity zahoď, nové nahrazuj nulami
  - bitovou rotaci o N bitů (kladné N vpravo, záporné vlevo)
  - aritmetický součet (přenos do vyššího řádu zahoď)
- Vstupní hodnoty pro předchozí funkce načítej ze standardního vstupu nebo příkazového řádku a do funkcí je předávej pomocí parametrů.

Pomocí podprogramů…

- Vytvoř dynamický datový typ zásobník. Datovou složkou bude jeden znak. Vytvoř funkce Push a Pop.
  - Otestuj pomocí nich, zda ve vstupním textu jsou správné počty správně uzavřených a vnořených závorek. Uvažuj kulaté (), hranaté [] a složené závorky {}. Například tohle „{([])}“ je správně, tohle „{[([])}“ je špatně.
- Vytvoř dynamický datový typ fronta. Datovou složkou bude jeden znak. Vytvoř operace Vlož a Vyber.
  - Použij je pro vypsání posledních N znaků vstupu předem neznámé délky.

Pomocí podprogramů vyřeš… Názvy souborů a dalších parametrů musí hlavní program získat od uživatele (ze vstupu nebo z parametrů příkazového řádku).

- Vytvoř podprogram, který zpracuje vstupní soubor tak, že každý výskyt znaku # nahradí zadaným textovým řetězcem a výsledek zapíše do výstupního souboru.
- Vytvoř podprogram, který zkombinuje dva vstupní soubory tak, že bude do výstupního souboru zapisovat střídavě vždy jeden řádek z prvního a pak ze druhého souboru.
- Vytvoř podprogram, který zjistí počet slov na každém řádku vstupního souboru a tyto počty zapíše na odpovídající řádky výstupního souboru.
- Vstupní soubor obsahuje textově zapsaná celá čísla větší než nula. Na začátku jsou data seřazená vzestupně na konci je oblast přeplnění. Seřazenou oblast a oblast přeplnění odděluje řádek s hodnotou 0. Vytvoř funkci, která v tomto souboru efektivně vyhledá zadanou hodnotu a bude brát v úvahu jak seřazenou oblast, tak oblast přeplnění.

Pomocí podprogramů vyřeš… Názvy souborů a dalších parametrů musí hlavní program získat od uživatele (ze vstupu nebo z parametrů příkazového řádku).

- V textových souborech jsou textově uloženy dvě databázové tabulky. V první tabulce (souboru) jsou uloženy záznamy s atributy ID, jméno (10 znaků). Ve druhé tabulce (souboru) jsou uloženy záznamy s atributy ID_uživatele, e-mail (20 znaků). V první tabulce má ID charakter primárního klíče a ID_uživatele je cizí klíč odkazující do první tabulky. Napiš podprogram, který vypíše na řádky všechny emailové adresy uživatelů se zadaným jménem (parametr podprogramu). Na každém řádku výstupu budou tyto údaje ID, jméno, e-mail.
- V tabulce (souboru) jsou záznamy o sopkách s atributy název (20 znaků), výška (v metrech), zda je aktivní (ano/ne). Vytvoř podprogram pro výpis všech záznamů o nad/podprůměrně (parametr) vysokých ne/aktivních (parametr) sopkách.

Doplň program (bude k dispozici kostra programu) pro řešení soustav lineárních rovnic pomocí Gaussovy / Gauss-Jordanovy eliminační metod. Napiš podprogram pro

- přímý chod Gaussovy metody
- zjištění počtu řešení po přímém chodu Gaussovy metody
- kontrolu, zda je matice koeficientů ve tvaru po provedení přímého chodu Gaussovy metody
- provedení zpětného chodu na i-tém řádku (předpokládá se, že na dalších řádcích již výpočet proběhl)
- kontrolu, zda je matice koeficientů po přímém chodu Gauss-Jordanovy metody
- provedení zpětného chodu Gauss-Jordanovy metody
- atd.

Pomocí podprogramů…

- Načti ze vstupu matici koeficientů o rozměrech m×n.
- Ověř, zda načtená matice může být maticí koeficientů soustavy lineárních rovnic (stačí rozměry).
- Ověř, zda je matice ostře diagonálně dominantní.
- Ověř, zda řešení pomocí Jacobiho nebo Gauss-Seidelovy iterační metody bude nad touto maticí koeficientů konvergovat.

Pomocí podprogramů…

- Vytvoř funkci pro vyčíslení zadaného polynomu v daném bodě x pomocí Hornerova schématu.
- Vytvoř funkci pro nalezení kořene nelineární funkce (parametr) se zadanou přesností (parametr) na zadaném intervalu (parametry) pomocí
  - metody půlení intervalu
  - metody regula falsi
  - Newtonovy metody (bude zadána derivace vyšetřované funkce)
- Ověř funkčnost na polynomu <bude zadán> na intervalu <bude zadán>.

Připrav si na papír

- odvození lineární regrese metodou nejmenších čtverců

- odvození konstantní regrese metodou nejmenších čtverců

Pomocí podprogramů…

- Vytvoř funkci, která z polí naměřených hodnot x a y vypočte aproximaci pomocí
  - konstantní funkce
  - lineární funkce (bude k dispozici funkce pro řešení soustavy 2 rovnic o 2 neznámých)
- Načti ze vstupu čtvercovou matici. Ověř, zda je matice souměrná podle
  - hlavní/vedlejší diagonály
  - vodorovné/svislé osy
- V polích x a y délky N jsou naměřené hodnoty. Je dána lineární funkce s koeficienty a, b. Vytvoř v C funkci, která vrátí součet čtverců vertikálních vzdáleností naměřených hodnot od dané lineární funkce.

Pomocí podprogramů… Vytvoř program pro seřazení pole n hodnot načtených ze vstupu. Vytvoř podprogram realizující algoritmus

- Insertion Sort / se zarážkou
- Selection Sort
- Bubble Sort s pamětí poslední výměny
- Merge Sort – slévání dvou úseků (bude k dispozici kostra se zbytkem algoritmu)

Pomocí podprogramů zpracuj tři soubory (zadá uživatel) po znacích.

- Dva vstupní soubory obsahují seřazené posloupnosti znaků. Sluč je do třetího výstupního souboru tak, aby v něm byly všechny znaky také seřazeny.
- Dva vstupní soubory obsahují znaky. Sluč je do třetího výstupního souboru tak, aby v něm byly znaky seřazeny po dvojicích.
- Vstupní soubor obsahuje posloupnost znaků. Rozděl je po N-ticích do dvou výstupních souborů (N je parametr).

Načti ze vstupu pole N celých čísel a hledanou hodnotu (ze vstupu nebo přes parametr příkazového řádku.

- Vytvoř podprogram, který v tomto poli vyhledá zadanou hodnotu algoritmem
  - sekvenčního hledání bez zarážky / se zarážkou v neseřazeném poli
- Ověř pomocí své funkce, zda je vstupní pole seřazeno. Vytvoř podprogram pro vyhledání v tomto poli algoritmem
  - sekvenčního hledání bez zarážky / se zarážkou v seřazeném poli
  - binárního vyhledávání

Doplň program (bude k dispozici kostra programu) pro práci s binárním vyhledávacím stromem obsahujícím celočíselné klíče. Napiš podprogram pro

- nalezení minimálního / maximálního prvku ve stromu
- výpis klíčů seřazených vzestupně / sestupně
- přidání nového klíče do stromu

- nalezení zadaného klíče
- výpis všech klíčů menších / větších než zadaný klíč
- zrušení celého stromu
- zrušení listu
- výpočet váhy stromu (počtu prvků ve stromu)
- výpočet výšky stromu (největší vzdálenost kořen – list)

Uprav a odlaď předložený program v C++ tak, aby se uplatnil
- polymorfismus
- dědičnost
- zapouzdření

Pomocí podprogramů…
- Vytvoř podprogram, který ze vstupního řetězce (parametr) odstraní všechny znaky, které se nemohou vyskytovat v binárním vyjádření adresy. Výsledkem bude řetězec tvořený nesmazanými znaky.
- Vytvoř podprogram, který bude mít ve vstupním textovém řetězci (parametr) zapsanou hodnotu v šestnáctkové soustavě. Ověř, zda tato hodnota může být adresou v 8, 16, 32 nebo 64 bitovém operačním systému. Funkce bude vracet hodnoty 0, 8, 16, 32 nebo 64. Nulu vracej v případě chyby, nebo příliš velké hodnoty.
- Nebo nějaká jiná stringologie…

Pomocí podprogramů…
- Ověř zda je zadaný textový řetězec validní MAC, IPv4, IPv6, e-mailovou adresou. Kontroluj oddělovače, přebytečné nuly, mezery atd.

Na papír…
- Na papír proveď minimalizaci logické funkce <bude zadána> o třech či čtyřech proměnných pomocí Karnaughovy mapy.
- Napiš logickou funkci, která ověří, zda je zadaný rok přestupným rokem. Rok je přestupný, když je dělitelný 4, ale není dělitelný 100, pokud zároveň není dělitelný 400. Napiš tento výraz co nejúsporněji.
- Napiš co nejúsporněji logickou funkci, která vrátí true, je-li číslo dělitelné 2, 3 nebo 4, ale zároveň vrátí false, je-li číslo dělitelné 6 nebo 8.