JMBAG

IME I PREZIME

Programiranje 1 – ispit, 2. 9. 2024.

Napomene: Svako rješenje napišite isključivo na papir sa zadatkom, jer jedino njega predajete. Pomoćne račune smijete raditi na drugim papirima koje će vam dati dežurni asistent. U svim zadacima **obavezno** pišite postupak!
Dozvoljeno je korištenje isključivo pribora za pisanje i brisanje, te službenog podsjetnika. Kalkulatori, te razne tablice, papiri i sl. nisu dozvoljeni! **Mobitele** isključite i pospremite daleko od sebe! Ako se ustanovi da **kod sebe** imate mobitel za vrijeme ispita, ispit se poništava i pokreće se stegovni postupak protiv vas.

Zadatak 1 (4+3+4+4=15 bodova)

(a) Neka je $b \geq 2$ prirodan broj i neka su x i y nenul znamenke u bazi b. Dokažite da je

$$(xxx)_{(b)} + (xx)_{(b)} \neq (yyy)_{(b)}.$$

(b) Izračunajte bez pretvaranja u bazu 10: $(AAA)_{(12)} + (13)_{(12)} \cdot (11)_{(12)}$.

(c) Neka su a, b i c logičke varijable koje se pojavljuju u logičkom izrazu f. Izraz f je istinit ako i samo ako je binarni broj $(1abc)_{(2)}$ prost (ako je varijabla istinita, pripadna znamenka je 1, a inače je 0). Odredite tablicu istinitosti te konjunktivnu **ili** disjunktivnu normalnu formu izraza f.

(d) U nekom tekstnom dokumentu zapisan je niz rečenica. Zapis je formatiran tako da je unutar zagrada navedeno nekoliko rečenica, a rečenice su međusobno odvojene zarezom i razmakom. Svaka rečenica se sastoji od jedne ili više riječi odvojenih razmakom. Riječi se sastoje od malih slova engleske abecede, s izuzetkom prve riječi u rečenici koja počinje velikim slovom. Na kraju svake rečenice je točka, upitnik ili uskličnik. Zadnja rečenica sastoji se od samo jedne riječi.

Primjer zapisa jednog niza je sljedeći:

```
(Bok␣mama.,␣Volim␣te␣mama!,␣Das␣mi␣pet␣kuna?,␣Pliz.)
```

Napišite regularan izraz koji prepoznaje ovako zapisan niz rečenica.

RJEŠENJA:

a) Primijetimo da je $x(b^2+2b+2) = y(b^2+b+1)$. Primijetimo da je $2x - y$ djeljivo s b, a manje od $2b$ i veće od $-b$. Onda je ili $2x = y$ ili $2x = y + b$.

Ako je $y = 2x$, imamo $x(b^2+2b+2) = x(2b^2+2b+2)$, kontradikcija.

Ako je $y = 2x - b$, imamo $x(b^2+2b+2) = (2x-b)(b^2+b+1)$. To se sredi kao $xb^2 = 2xb^2 - b^3 - b^2 - b$, ali onda je $b^3 + b^2 + b = xb^2 < b^3$, opet kontradikcija.

b) 1065_{13}

c) Tablica istinitosti:

a	b	c	$(1abc)_2$	Je li broj prost?
0	0	0	8	ne
0	0	1	9	ne
0	1	0	10	ne
0	1	1	11	da
1	0	0	12	ne
1	0	1	13	da
1	1	0	14	ne
1	1	1	15	ne

KNF:

$$f = (\neg a \wedge b \wedge c) \vee (a \wedge \neg b \wedge c)$$

d) $[A-Z][a-z]*([a-z]+)*[.?!]\,[A-Z][a-z]*([a-z]+)*[.?!])*\ \ [A-Z][a-z]*[.?!]$

JMBAG

IME I PREZIME

Programiranje 1 – ispit, 2. 9. 2024.

Zadatak 2 (5+5=10 bodova)

(a) Objasnite što je enumeracija. Napišite definiciju enumeracije koja reprezentira dane u tjednu. Napišite deklaraciju varijable tipa definirane enumeracije i postavite joj vrijednost na konstantu PONEDJELJAK.

(b) Napišite C-program koji implementira jednostavan kalkulator koristeći `if-else` naredbu. Kalkulator prima dva realna broja, te znak koji reprezentira operaciju ('+' - plus, '-' - minus, '*' - puta i '/' - podijeljeno). Kalkulator računa rezultat operacije nad ulaznim brojevima te ga ispisuje u komandni prozor.

RJEŠENJE:

a) Enumeracije su vrste konstanti u C-u. Enumeracijske konstante su simbolička imena za cjelobrojne konstante, pišu se kao identifikatori. Enumeracija je tip koji: a) počinje ključnom riječi enum, b) sadrži popis imena za cjelobrojne konstante unutar vitičastih zagrada. Prvom identifikatoru se pridružuje konstanta 0, drugom konstanta 1 i tako redom. (predavanje 6, slajd 39).

```
 #include <stdio.h>

enum dani {PONEDJELJAK, UTORAK, SRIJEDA, CETVRTAK, PETAK, SUBOTA, NEDJELJA};

int main(void){

  enum dani x;
  x = PONEDJELJAK;

 return 0;
}
```

b)

```
#include <stdio.h>

int main(void){

  double a,b, rezultat = 0.0;
  char c;
  int ok = 1;

  printf("Ucitajte dva realna broja i operaciju (znak)\n");
  scanf("%lg %lg",&a,&b);
  scanf(" %c",&c);

  if(c == '+') rezultat = a+b;
  else if(c == '-') rezultat = a-b;
  else if(c == '*') rezultat = a*b;
  else if(c == '/') rezultat = a/b;
  else{
    ok = 0;
    printf("Unesen je krivi znak!");
  }

  if(ok == 1) printf("Rezultat je: %g",rezultat);

   return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 1 – ispit, 2. 9. 2024.

Zadatak 3 (6+19=25 bodova)

Neka je $p_0(x) = a_0$ konstantni polinom. Rekurzivno definiramo niz polinoma $p_1(x), p_2(x), p_3(x), \ldots$ tako da je

$$p_n(x) = \sum_{k=0}^{n} p_{n-1}(k)^2 x^k \text{ za } n \geq 1.$$

(a) Napišite pomoćnu funkciju `double eval(int k, double a[], double y)` koja prima nenegativan cijeli broj k, niz $a_0, a_1, \ldots, a_k$ i realan broj y te vraća $\sum_{j=0}^{k} a_j y^j$ koristeći Hornerov algoritam. Ako ne koristite Hornerov algoritam, možete dobiti maksimalno 1 bod na ovom podzadatku.

(b) Napišite funkciju `double mega_eval(double a_0, double y)` koja prima realne brojeve a_0 i y. Funkcija treba vratiti vrijednost $p_{20}(y)$. Kad god evaluirate bilo koji polinom u nekoj točki, treba koristiti funkciju `eval` iz (a) podzadatka. Smijete koristiti tu funkciju čak i ako je niste napisali. Smijete koristiti pomoćna polja.

NAPOMENA: U ovom zadatku smijete koristiti **najviše** tri polja duljine 21

RJEŠENJE:

```
#include <stdio.h>

// Funkcija za evaluaciju polinoma pomoću Hornerovog algoritma (iz (a) dijela zadatka)
double eval(int k, double a[], double y) {
    double result = a[k];  // Početni koeficijent a_k
    for (int i = k - 1; i >= 0; i--) {
        result = result * y + a[i];  // Hornerova metoda
    }
    return result;
}

// Funkcija za izračunavanje p_{20}(y) prema zadanoj rekurzivnoj formuli
double mega_eval(double a_0, double y) {
    double prev[21];  // Polinom za p_{n-1}
    double curr[21];  // Polinom za p_{n}

    // Inicijaliziramo p_0(x) = a_0
    prev[0] = a_0;

    // Rekurzivno definiranje polinoma p_n(x) za n >= 1
    for (int n = 1; n <= 20; n++) {
        // Inicijaliziramo polinom p_n(x) s 0
        for (int k = 0; k <= n; k++) {
            curr[k] = 0;
            // Sumiramo kvadrate prethodnog polinoma evaluiranih na točkama k
            for (int j = 0; j < n; j++) {
                double val = eval(j, prev, k);  // Evaluiramo p_{n-1}(k)
                curr[k] += val * val;  // Dodajemo kvadrat vrijednosti
            }
        }

        // Ažuriramo prev s vrijednostima iz curr za sljedeću iteraciju
        for (int k = 0; k <= n; k++) {
            prev[k] = curr[k];
        }
    }

    // Vraćamo vrijednost p_20(y) koristeći eval funkciju
    return eval(20, curr, y);
}
```

```
int main() {
    double a_0 = 1.0;  // Vrijednost za p_0(x) = a_0
    double y = 2.0;    // Vrijednost za evaluaciju

    // Izračunaj p_20(y)
    double rezultat = mega_eval(a_0, y);

    // Ispis rezultata
    printf("Vrijednost p_20(%f) je: %f\n", y, rezultat);

    return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 1 – ispit, 2. 9. 2024.

Zadatak 4 (25 bodova)

Za prirodan broj d veći od 1 i prirodan broj a, definiramo $v_d(a)$ kao najveći nenegativan cijeli broj m takav da je d^m djelitelj od a.

Napišite funkciju `void divisible_sort(int n, int a[], int d, double *avg)` koja prima prirodan broj n, niz različitih prirodnih brojeva $a_0, a_1, \ldots, a_{n-1}$ i prirodan broj d veći od 1.

Funkcija treba sortirati niz $a_0, a_1, \ldots, a_{n-1}$ tako da za $i < j$ vrijedi da je $v_d(a_i) < v_d(a_j)$ ili $v_d(a_i) = v_d(a_j)$ i $a_i < a_j$. Preko varijabilnog argumenta `avg` treba vratiti prosječnu vrijednost od $v_d(a_i)$ u nizu.

Treba napisati i program koji korisniku omogućava da učita niz brojeva $a_0, a_1, \ldots, a_{n-1}$, sortira ga koristeći napisanu funkciju za $d = 5$, te ispisuje sortirani niz i prosječnu vrijednost od $v_5(a_i)$.

RJEŠENJE:

```
#include <stdio.h>

// Funkcija za računanje v_d(a)
int v_d(int a, int d) {
    int count = 0;
    while (a % d == 0) {
        a /= d;
        count++;
    }
    return count;
}

// Pomoćna funkcija za zamjenu elemenata u nizu
void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

// Funkcija za sortiranje niza prema uvjetima iz zadatka
void divisible_sort(int n, int a[], int d, double *avg) {
    int i, j;
    int total_vd = 0;

    // Sortiranje niza koristeći bubble sort prema uvjetima
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - 1 - i; j++) {
            int vd1 = v_d(a[j], d);
            int vd2 = v_d(a[j+1], d);

            // Ako je v_d(a[j]) > v_d(a[j+1]) ili ako su jednaki, a a[j] > a[j+1], zamijeni ih
            if (vd1 > vd2 || (vd1 == vd2 && a[j] > a[j+1])) {
                swap(&a[j], &a[j+1]);
            }
        }
    }

    // Izračun prosjeka v_d(a_i)
    for (i = 0; i < n; i++) {
        total_vd += v_d(a[i], d);
    }
    *avg = (double) total_vd / n;
}
```

```c
// Glavni program
int main() {
    int n, i;
    double avg;

    printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);

    int a[n];

    printf("Unesite %d prirodnih brojeva: ", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &a[i]);
    }

    divisible_sort(n, a, 5, &avg);

    printf("Sortirani niz: ");
    for (i = 0; i < n; i++) {
        printf("%d ", a[i]);
    }

    printf("\nProsječna vrijednost v_5(a_i): %.2f\n", avg);

    return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 1 – ispit, 2. 9. 2024.

Zadatak 5 (5+ 20 + [5] = 25 + [5] bodova)

(a) Napišite funkciju `void kanta(int a[], int n, int b[])` koja prima niz $(a[0], a[1], \ldots, a[n-1])$ koji se sastoji od nenegativnih cijelih brojeva manjih od 10 i njegovu duljinu n, te niz cijelih brojeva $(b[0], b[1], \ldots, b[9])$. Funkcija treba u $b[i]$ upisati broj pojava broja i u nizu $(a[0], a[1], \ldots, a[n-1])$.

(b) Napišite funkciju `void histogram(int a[], int n, int *h, int *argh)` koja prima niz $(a[0], a[1], \ldots, a[n-1])$ koji se sastoji od nenegativnih cijelih brojeva manjih od 10 i njegovu duljinu n. Funkcija treba ispisati sveukupno 10 stupaca sastavljenih od simbola '*', tako da se i-ti stupac sastoji od onoliko zvjezdica koliko puta se broj $i-1$ pojavljuje u zadanom nizu. Stupci međusobno trebaju biti odvojeni razmakom. Možete pretpostaviti da će se svaki broj od 0 do 9 pojaviti barem jednom u nizu.

Na primjer, za niz $(1, 1, 2, 3, 2, 9, 0, 9, 1, 4, 4, 5, 6, 7, 7, 7, 8)$, ispis treba izgledati ovako:

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

Nadalje, funkcija treba u varijabilne argumente `h` i `argh` zapisati visinu najvišeg stupca histograma i neki broj koji se pojavljuje najviše puta u nizu.

Smijete koristiti funkciju iz (a) podzadatka.

(c) (Dodatnih 5 bodova koji se ne ubrajaju u uvjet od 80 posto) Napišite funkciju `void sortiranje_kantom(int a[], int n)` koja prima niz $(a[0], a[1], \ldots, a[n-1])$ koji se sastoji od nenegativnih cijelih brojeva manjih od 10 i njegovu duljinu n, te sortira niz. Smijete koristiti funkciju iz (a) podzadatka. Podzadatak treba riješiti u linearnoj složenosti $\mathcal{O}(n)$, u protivnom nećete dobiti bodove na njemu.

RJEŠENJE:

```
#include <stdio.h>

void kanta(int a[], int n, int b[]){
    for(int i=0 ; i<10 ; i++){
        b[i]=0;
        for(int j=0 ; j<n ; j++)
            if(a[j]==i) b[i]++;
    }
}

void histogram(int a[], int n, int* h, int* argh){
    int b[10];
    kanta(a, n, b);
    int max=0;
    for(int i=0 ; i<10 ; i++){
        if(b[i]>max){
            max=b[i];
            *argh=i;
        }
    }
    *h=max;
    while(max){
        for(int i=0 ; i<10 ; i++){
            if(b[i]>=max) printf("* ");
            else printf("  ");
    }
        printf("\n");
        max--;
    }
}
```

```
void sortiranje_kantom(int a[], int n){
    int b[10];
    kanta(a, 17, b);
    int j=0;
    for(int i=0 ; i<10 ; i++){
        int k=b[i];
        while(k){
            a[j]=i;
            j++;
            k--;
        }
    }
}

int main()
{
    int n=17;
    int a[17]={1, 1, 2, 3, 2, 9, 0, 9, 1, 4, 4, 5, 6, 7, 7, 7, 8};
    int b[10];
    int h, argh;
    histogram(a, n, &h, &argh);
    printf("%d\n", h);
    printf("%d\n", argh);
    sortiranje_kantom(a, n);
    for(int i=0 ; i<n ; i++){
        printf("%d ", a[i]);
    }
    return 0;
}
```