Структуре података и алгоритми 1 - II тест

Институт за математику и информатику, ПМФ Крагујевац

16. мај 2024.

Одговоре написати у празном простору одговарајућег задатка. Кодови се односе на *GCC 14 x86_64* компајлер. Сматрати да је `stdio.h` увек укључен. На свакој страни написати име, презиме, индекс и годину. Време за израду теста је 60 минута.

1. (1 поен) Допунити функцију за учитавање, тако да одговара датом испису:

```
struct film {
  int trajanje;
  float budzet;
  char naziv[30];
  union {
    int godine;
    char ime[30];
    char prezime[30];
  } reziser;
};

void ucitaj(struct film *film) {
  scanf("%s %s %f", film->naziv, film->reziser.ime, &film->budzet);
}

int main() {
  struct film film;
  ucitaj(&film);
  printf("Naziv filma:%s\nIme rezisera:%s\nBudzet:%f\n", film.naziv, film.
      reziser.ime, film.budzet);
}
```

2. (1 поен) Дата је променљива `int x[3][2][1]`. Приступити елементу `x[1][1][0]` ако је дозвољено само коришћење симбола `x`, `(`, `)`, `+`, `-`, `>`, `3`, `2`, `1` и `*`.

```
**(*(x+1)+1)
```

3. (2 поена) Шта ће бити испис следећег програма:

```
struct s {
  char c;
};

int main() {
  struct s p1[] = {'I', 'M', 'I'};
  struct s* ptr1 = p1;
  printf("%ld\n", sizeof(p1));                    3
  printf("%ld\n", sizeof(ptr1));                  8
}
```

4. (2 поена) Написати програм који при покретању из командне линије прима произвољни број (бар један) целобројних аргумената и враћа највећи јединствени.

```
$ ./najveci 1 -2 3 0 4 1 4
$ 3
```

```
#include <stdlib.h>
int compare(const void* a, const void* b) {
  return (*(int*)a - *(int*)b);
}

int main(int argc, char* argv[]) {
  int a[argc - 1], i;
  for (i = 1; i < argc; i++) a[i - 1] = atoi(argv[i]);

  qsort(a, argc - 1, sizeof(int), compare);
  for (i = argc - 2; i > 0; i--) {
    if (a[i] != a[i - 1]) break;
    while (i && a[i] == a[i - 1]) i--;
  }

  if (i > -1) printf("%d\n", a[i]);
}
```

Напомена: Било који сорт је прихватљив, као и решења која не укључују сортирање.

5. (2 поена) Написати функцију која за дати број алоцира и попуњава "полуматрицу", а као резултат враћа показивач на њу, као у следећем примеру:

```
5
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

```
int** polumatrica(int n) {
  int** m = (int**)malloc(n * sizeof(int*));

  for (int i = 0; i < n; i++) {
    m[i] = (int*)malloc((n - i) * sizeof(int));

    for (int j = 0; j < n - i; j++)
      m[i][j] = j + 1;
  }

  return m;
}
```

6. (2 поена) Дат је показивач `p` на први елемент једноструко повезане листе:

```
struct element {
  int broj;
  struct element *sledeci;
};
```

Написати функцију која убацује нови елемент вредности 5 после другог места у листи.

```
void ubaci_posle_drugog(struct element* p) {
  if (!p || !p->sledeci) {
    printf("Lista nema dovoljno elemenata.\n");
    return;
  }

  struct element* drugi = p->sledeci;

  struct element* novi = (struct element*)malloc(sizeof(struct element));
  novi->broj = 5;

  novi->sledeci = drugi->sledeci;
  drugi->sledeci = novi;
}
```

7. (2 поена) Написати функцију која без употребе додатних структура података избацује дупликате из једноструко повезане листе. (нпр. 1 2 3 1 1 2 трансформише у 1 2 3)

```
void izbaci_duplikate(struct element* p) {
  struct element* trenutni = p;

  while (trenutni && trenutni->sledeci) {
    struct element* prethodni = trenutni;
    struct element* proveri = trenutni->sledeci;

    while (proveri)
      if (trenutni->broj == proveri->broj) {
        prethodni->sledeci = proveri->sledeci;
        free(proveri);
        proveri = prethodni->sledeci;
      } else {
        prethodni = proveri;
        proveri = proveri->sledeci;
      }

    trenutni = trenutni->sledeci;
  }
}
```

8. (1 поен) Колико пута ће доћи до замене места елемената низа 5 1 7 2 4 2 приликом његовог сортирања у неопадајући поредак, употребом Bubble Sort алгоритма?

4

Напомена: Признаје се и решење 8 у случају верзије алгоритма без оптимизације.