

Osnovi programiranja

I kolokvijum

9.11.2018.

I grupa

Na **Desktop**-u u direktorijumu **Rad** kreirati direktorijum **ImePrezime_BrIndeksa** i unutar njega sačuvati programe koji sadrže rešenja datih zadataka. Rešenje 1. zadatka **mora** da se nalazi u fajlu **Zadatak1.c**, rešenje 2. zadatka **mora** da se nalazi u fajlu **Zadatak2.c**. Od dva ponuđena zadatka birate **jedan** koji ćete raditi.

Napisati C program koji:

1. U učitanoj broju **n** nalazi se rešenje dugo tražene mape skrivenog blaga. Otkrivanje mape ima dva moguća načina. Ukoliko učitani broj ima jednu ili dve cifre brojana vrednost koja otvara tajni sef u kome se nalazi mapa skrivenog blaga se dobija kao rezultat date sume:

$$f(n) = \sum_{i=1}^n (1 + 2 + \dots + i) \cdot \frac{|n^i - i^i|}{(2 + 4 + \dots + 2n)}$$

S druge strane, ukoliko učitani broj ima više od 2 cifre iza odgovarajuće cigle se nalazi mapa skrivenog blaga. Odabir cigle se vrši na sledeći način: ako učitani broj **n** ima neparan broj cifara, iza cigle na kojoj je broj cifara broja **n** nalazi se mapa skrivenog blaga. Ukoliko učitani broj ima paran broj cifara, onda se mapa skrivenog blaga nalazi iza cigle sa brojem koji predstavlja sumu prve i poslednje cifre broja **n** ($n=1234 \rightarrow$ cigla broj 5; $n=56781 \rightarrow$ cigla broj 5).

Ulaz sadrži samo jedan ceo broj **n**, izlaz sadrži samo jedan broj, ili realan broj koji otvara tajni sef u formatu **na deset mesta sa pet decimale**, ili predstavlja ceo broj na traženoj cigli i to u formatu **na 3 mesta**.

NAPOMENA: Funkcije **abs/fabs**, **pow**, kao i **svoje funkcije** ne smete koristiti ni kod uslova definisanosti, ni kod računanja vrednosti. Takođe, tekući član sume **OBAVEZNO** računati na osnovu prethodnog člana, ukoliko je to moguće. Zadatak se boduje i delimično, ukoliko se reši samo jedan način otkrivanja mape skrivenog blaga.

Ulaz	Izlaz
1234	5
56781	5
1	0.00000
3	1.41667

2. Klovni Logi ima **n** kutija koje treba redom spakovati jednu u drugu u potpunosti na najbolji mogući način, ako je to moguće. Kutija se može dodati u spakovane na dva načina:

- Da se ubaci u poslednje spakovanu kutiju
- Da se sve spakovane kutije ubace u nju.

Ako kutija ne može da se spakuje sa prethodno spakovanim, uklanja se. Pomozite klovnu Logiju da sazna koliko kutija neće uspeti da spakuje. Program učitava broj kutija **n**, i dimenzije (dužina, širina, visina) svih **n** kutija koje redom pokušava da spakuje (kutije mogu da se okreću i otvore na bilo koju stranu). Odštampati koliko kutija je ostalo nespakovano.

Ulaz	Izlaz
6 100 100 100 90 95 80 120 110 130 110 120 120 50 80 75 60 30 60	1

Broj poena: 1. zadatak – 13 poena, 2. zadatak – 16 poena

Vreme izrade: 75 minuta

Osnovi programiranja

I kolokvijum

9.11.2018.

II grupa

Na **Desktop**-u u direktorijumu **Rad** kreirati direktorijum **ImePrezime_BrIndeksa** i unutar njega sačuvati programe koji sadrže rešenja datih zadataka. Rešenje 1. zadatka **mora** da se nalazi u fajlu **Zadatak1.c**, rešenje 2. zadatka **mora** da se nalazi u fajlu **Zadatak2.c**. Od dva ponuđena zadatka birate **jedan** koji ćete raditi.

Napisati C program koji:

3. Rimljanima su formacije za napad predstavljali brojevima sastavljeni samo od cifara 0 i 1. Kada su slali formaciju za napad na bojno polje, šifrovali su je u obliku jednog celog broja **n**. Poslat broj **n** se tumači na sledeći način: na mestima u broju gde su parne cifre, smatra se da je 1, a gde su neparne da je 0, tako da dešifrovanjem dobiju ceo broj **m** ($n=1524 \rightarrow m=11$; $n=32281 \rightarrow m=1110$). Međutim oni ne znaju šta da rade ako dobiju da je **m** nula, u toj situaciji crtež sa formacijom napada se nalazi u kutiji koja se otvara brojem koji se dobija kao rezultat sume:

$$f(n) = \sum_{i=1}^n (3 + 6 + \dots + 3i) \cdot \frac{(-1)^i \cdot i^i + n!}{(5 + 5^2 + \dots + 5^i)}$$

Ulaz sadrži samo jedanceo broj **n**, izlaz sadrži **samo jedan** ceo broj **m** koji predstavlja nov broj i to u formatu **na deset mesta** (ukoliko je dobijen broj **m** različit od nule) ili realan broj koji otvara kutiju u formatu **na deset mesta sa pet decimala**.

NAPOMENA: Funkcije **abs/fabs, pow, kao i svoje funkcije** ne smete koristiti ni kod uslova definisanosti, ni kod računanja vrednosti. Takođe, tekući član sume **OBAVEZNO** računati na osnovu prethodnog člana, ukoliko je to moguće. Zadatak se boduje i delimično, ukoliko se reši samo jedan način okivanja formacije.

Ulaz	Izlaz
1524	11
32281	1110
1	0.00000
3	3.56130

4. Mika i Žika igraju igru pokrivanja. Svaki od njih ima po **n** poteza u kojima biraju karte sa po tri broja. Igru započinje računar tako što izabere tri karte sa brojevima **a** u svakom potezu igru prvi igra Mika. Pokrivanje karata se može izvršiti na dva načina:

- Izabrani brojevi na kartama su veći od brojeva na vrhu gomila sa kartama (svaka od izabranih karata je veća od karte na vrhu gomile na koju se spušta i tom prilikom se vodi računa o idealnom rasporedu karata na gomile)
- Izabrani brojevi na kartama su manji od brojeva na dnu gomila sa kartama (svaka od izabranih karata je manja od karte na dnu gomile ispod koje se stavlja i tom prilikom se vodi računa o idealnom rasporedu karata na gomile)

Ukoliko pokrivanje ne može da se izvrši smatra se da je potez propuštenih. Odredite pobednika u igri a pobednik je onaj igrač koji ima manje propalih poteza i štampati broj tih poteza. Program učitava broj poteza **n** po igraču, vrednosti na kartama **x, y i z** koje postavlja računar i svih **n** poteza i za Miku i za Žiku (u svakom potezu, za svakog od njih po tri broja). Odštampati ko je pobedio i sa koliko propuštenih poteza. Ako imaju isto propalih poteza štampa se **NERESEN** i broj propuštenih poteza.

Primer:

Ulaz	Izlaz
4	Mika
5 4 9	1
3 4 5	
5 1 2	
11 6 10	
8 16 9	
12 11 8	
3 2 4	
2 3 4	
20 12 15	

3	2	4
4	3	5
5	4	9
10	6	11
11	8	12
15	12	20

Broj poena: 1. zadatak – 13 poena, 2. zadatak – 16 poena
Vreme izrade: 75 minuta