OPISI ALGORITAMA

Županijska razina 2020/ Osnovna škola (5. i 6. i 7. i 8. razred)

Primjena algoritama OŠ

Agencija za odgoj i obrazovanje
Education and Teacher Training Agency

HRVATSKI SAVEZ INFORMATIČARA

Ministarstvo znanosti, obrazovanja i sporta

5.1. Zadatak: Tagus Autor: Nikola Dmitrović

U ovom zadatku starost svakog predmeta pretvorit ćemo u starost u danima prema formuli dani=G*4*7*12+M*4*7+T*7. Kada odredimo starost predmeta X ispišemo ju. Također u neku varijablu, npr. zbroj pamtimo ukupan zbroj starosti u danima svih predmeta.

5.2. Zadatak: Dorino Autor: Nikola Dmitrović

Na obije domine izračunamo koji je veći od dva dvoznamenkasta broja. Zapamtimo to u varijable *prvi* i *drugi*. Za odgovor na prvo pitanje ispišemo veći od ta dva broja. Za odgovor na drugo pitanje ispišemo veći od ta dva broja pomnožen sa 100 i tom umnošku pribrojimo manji broj.

5.3. Zadatak: Odluka Autor: Nikola Dmitrović

Pretvorimo dan u tjednu u odgovarajući broj između 0 i 6. Rješenje je sada ispisati (dan+(n-1)*x)%7+1.

6.1. Zadatak: Posao Autor: Nikola Dmitrović

Zadatak je najlakše riješiti ako dane u tjednu zamijenimo brojevima od 1 do 7. Tada odgovor na prva dva pitanja možemo dobiti jednostavnim matematičkim operacijama i usporedbama, dok za odgovor na treće pitanje trebamo za svaki dan u tjednu izračunati koliko je taj dan radilo programera i tada pronaći dan s najviše radnika.

6.2. Zadatak: TZK Autor: Nikola Dmitrović

Odgovor na prvo pitanje možemo dobiti da u neku varijablu zbroj izračunamo zbroj upaljenih lampica za sve znamenke u M i sve znamenke u S. Odgovor na drugo pitanje dobivamo uz pomoć dvije for petlje, jedna koja će nam simulirati minute, druga sekunde, obje idu od 0 do 59. Kada smo s petljama na nekom vremenu, npr., a:b, rastavimo a na znamenke, b na znamenke, izračunamo koliko lampica treba biti upaljeno za to vrijeme a:b te ako je taj broj upaljenih lampica jednak K povećamo neki brojač za rješenje za 1.

6.3. i 8.1. Zadatak: Navijač Autor: Nikola Dmitrović

Zadatak rješavamo simulacijom uvijeta iz teksta zadatka. Trebamo pomoćne varijable u koje ćemo pamtiti koliko je golova postigla svaka reprezentacija, koliko je sekundi hrvatski navijač bio sretan te jednu koja će nam govoriti kada je navijač postao sretan ako njegova sreća traje i u ovom trenutku. Uz pomoć ove zadnje možemo izračunati odgovor na treće pitanje.

7.1. Zadatak: Kava Autor: Nikola Dmitrović

Prvo što ćemo učiniti je proći kroz string S kako bismo shvatili koliko žličica automat ubacuje u svaku od pripremljenih kava. Budući da znamo da automat ubacuje sve dosada nepostavljene žličice kada učini

dodatak, koristit ćemo brojač neubacene_žličice koji će povećavati za 1 svaki put kada naiđemo na nulu (0) i pohraniti ukupno ubačene žličice kada naiđemo na jedinicu (1).

1. Inicijalizirajte brojač neubacene_žličice na nulu.
2. Inicijalizirajte listu ubacene_žličice_po_kavi na praznu listu.
3. Prođite kroz svaki znak u stringu S:
    a. Ako je znak nula (0), povećajte neubacene_žličice za 1.
    b. Ako je karakter jedinica (1), povećajte neubacene_žličice za 1, dodajte taj broj u listu ubacene_žličice_po_kavi i resetirajte brojač neubacene_žličice natrag na nulu.
4. Sada, kada imamo sve ubačene žličice po kavama, možemo izračunati sve tražene informacije:
    a. Ukupan broj žličica koje su ubačene je zbroj svih vrijednosti u listi ubacene_žličice_po_kavi.
    b. Broj žličica ubačenih u K-tu kavu je ubacene_žličice_po_kavi[K-1] (s obzirom da indeksiranje u listama počinje od 0).
    c. Najveći broj žličica ubačenih u neku od kava je maksimalna vrijednost u listi ubacene_žličice_po_kavi.

Na kraju, ispišite odgovore u traženom formatu.

7.2. Zadatak: Žirafa — Autor: Josip Klepec

Rješenje ovog zadatka je simulacija Maksimilijanovog postupka. Pseudo kod rješenja je:

1. Radi tako dugo dok nisi nazvao sve prijatelje
    a. Ako postoji prijatelj s trenutnim brojem, nazovi ga, odnosno povećaj rješenje za 5 i označi (u nekoj pomoćnoj listi) da si nazvao tog prijatelja
    b. Povećaj trenutni broj za jedan, odredi koliko to traje sekundi te povećaj rješenje za taj broj sekundi

Na kraju ispiši rješenje.

7.3. Zadatak: Megić — Autor: Stjepan Požgaj

Kada unosimo podatke u neku listu odmah trebamo spremati parove (kada će ta osoba naručiti, koja je to osoba). Primjetimo da trebamo imati neku varijablu koja prati koliko je dugačak red na blagajni, odnosno kada najprije osoba koja želi naručiti na blagajni, može naručiti na blagajni.

Kada smo napravili listu tih parova, sortiramo ju i prolazimo element po element. Nakon sortiranja znamo da ćemo obilaziti ljude točno onim redom kojim je kuhinja zaprimala njihove narudžbe pa opet možemo imati neku varijablu koja prati kada će kuhinja završiti sa svim dosad zaprimljenim narudžbama.

8.2. Zadatak: Biljar — Autor: Marin Kišić

10 bodova na ovome zadatku moglo se je osvojiti tako da isprobamo sve moguće poteze i provjerimo koji od njih će posložiti kuglice u dobar raspored.

Dodatnih 20 bodova se je moglo osvojiti tako da isprobamo sve moguće kombinacije neka 2 poteza.

Opišimo sada algoritam koji nosi barem 35 bodova, odnosno točan je, ali ne u najmanjem broju poteza.

1. Fiksiraj u koji od dva dobra rasporeda ćemo posložiti kuglice, na kraju ćemo uzeti samo rješenje za onaj do kojeg možemo doći u manjem broju poteza
2. Pronađi kuglicu koja je na krivoj poziciji, npr. "puna" na poziciji gdje treba biti "prazna" ili "crna". Ako su sve na dobrim pozicijama, onda smo gotovi.
3. Pronađi kuglicu koja je na krivoj poziciji i u našem rasporedu dobra pozicija za nju je ona koju smo pronašli u prethodnom koraku
4. Zamjeni kuglice iz koraka 2 i 3 i vrati se na korak 2

Lako se dokaže da ako smo našli neku kuglicu u koraku 2, da ćemo onda moći pronaći i neku u koraku 3 te pošto nakon svake zamjene broj kuglica na krivim pozicijama smanjimo barem za 1, onda znamo da će algoritam završiti pa s tim možemo zaključiti da je algoritam točan.

Za sve bodove bilo je potrebno uočiti da ako bismo na prvom potezu "crnu" doveli na svoje mjesto i onda izvršili gornji algoritam, tada bismo napravili najmanji mogući broj koraka. Dokaz ove tvrdnje ostavljam čitateljima za vježbu.

Potrebno znanje: for petlja, logičko razmišljanje

Kategorija: ad hoc

8.3. Zadatak: DJ — Autor: Karlo Franić

Za prvi podzadatak bilo je dovoljno isprobati sve poretke pjesama u playlisti.

Za drugi podzadatak trebamo primjetiti da je nebitno kojim redoslijedom slažemo pjesme u sredini playliste. Onda možemo isprobati sve kombinacije za prvu i zadnju pjesmu u playlisti te proći po svima i uzeti onaj njihov dio koji će se pustiti.

Za daljnje podzadatke trebamo primijetiti da svaku pjesmu možemo podijeliti na tri dijela: početak, sredinu i kraj. Početak *i*-te pjesme traje od prve do $(P_i - 1)$-te sekunde, sredina od P_i-te do K_i-te sekunde, a kraj od $(K_i + 1)$-te do T_i-te sekunde. Tada ćemo od svih pjesama sigurno pustiti njihovu sredinu, a od prve pjesme ćemo još dodatno pustiti njen početak, a od zadnje kraj.

Za treći podzadatak onda možemo prvo dodati trajanja svih sredina u neku varijablu. Zatim isprobamo sve kombinacije za prvu i zadnju pjesmu te samo dodamo njihov početak odnosno kraj.

Za sve bodove trebamo primijetiti da nam je kao prvu pjesmu u playlisti najbolje uzeti pjesmu s najduljim početkom, a kao zadnju onu s najduljim krajem. Jedino na što moramo pripaziti je da ne uzmemo istu pjesmu za prvu i zadnju pa moramo pamtiti koja pjesma ima najdulji, ali i drugi najdulji početak odnosno kraj. Ako ista pjesma ima najdulji početak i najdulji kraj onda ćemo uzeti pjesmu s drugim najduljim početkom kao prvu ili s drugim najduljim krajem kao zadnju, ovisi što je bolje.

Programski kod (pisan u Python 3)

```
n = int(input())
max_poc1 = -1
max_poc2 = -1
max_kraj1 = -1
```

```
max_kraj2 = -1
ukupno = 0
ind_poc1 = 0
ind_kraj1 = 0
for i in range(n):
      p, k, t = map(int, input().split())
      ukupno += k - p + 1
      if p - 1 >= max_poc1:
      max_poc2 = max_poc1
      max_poc1 = p - 1
      ind_poc1 = i
      elif p - 1 > max_poc2:
      max_poc2 = p - 1
      if t - k >= max_kraj1:
      max_kraj2 = max_kraj1
      max_kraj1 = t - k
      ind_kraj1 = i
      elif t - k > max_kraj2:
      max_kraj2 = t - k
if ind_poc1 == ind_kraj1:
      ukupno += max(max_kraj1 + max_poc2, max_poc1 + max_kraj2)
else:
      ukupno += max_kraj1 + max_poc1
print(ukupno)
```

Potrebno znanje: for petlja, logičko razmišljanje

Kategorija: ad hoc