3. veljače 2021.

2021 Natjecanje iz informatike

Školska razina 2021/ Osnovna škola (5. i 6. i 7. i 8. razred)

Primjena algoritama OŠ

OPISI ALGORITAMA

Agencija za odgoj i obrazovanje
Education and Teacher Training Agency

HRVATSKI SAVEZ INFORMATIČARA

Ministarstvo znanosti, obrazovanja i sporta

5.1. Zadatak: Pahuljice

Autor: Nikola Dmitrović

Kako u posudi trebaju biti 73 pahuljice, provjerit ćemo je li zadani broj N manji ili veći od 73 te ovisno o tome odrediti koliko pahuljica treba dodati ili oduzeti.

Programski kod (pisan u Python 3)

```
N = int(input())
if N < 73:
    print('DODAJ')
    print(73 - N)
else:
    print('IZVADI')
    print(N - 73)
```

Potrebno znanje: naredba odlučivanja

Kategorija: ad hoc

5.2. Zadatak: Igra

Autor: Adrian Satja Kurdija
Prilagodio: Pavel Kliska

Potrebno je unijeti brojeve A, B, C i D te ispisati sljedeće:

- A (broj bodova na prvoj razini),
- A + B (broj bodova na prvim dvjema razinama),
- A + B + C (broj bodova na prvim trima razinama),
- A + B + C + D (broj bodova na svim četirima razinama).

Potom za broj A + B + C + D (ukupan broj bodova) treba s pomoću if-naredbe provjeriti je li manji od 100. Ako jest, ispisujemo NE, a inače ispisujemo DA.

Zadatak je originalno objavljen na školskom natjecanju za 5. razred 2015. godine. Za ovogodišnje je natjecanje prilagođen.

Potrebno znanje: osnovne operacije, naredba odlučivanja

Kategorija: ad hoc

5.3. Zadatak: Točkice

Autor: Gabrijel Jambrošić

Imamo šest brojeva, a za svaki po tri načina brojanja točkica, dakle ukupno 18 slučajeva. Naredbom grananja lako provjerimo o kojem slučaju se radi te ispišemo odgovarajuće rješenje.

Potrebno znanje: naredba odlučivanja

Kategorija: ad hoc

6.1. Zadatak: Triple — Autor: Nikola Dmitrović

Trebamo provjeriti jesu li sva tri zadana broja dvoznamenkasta te ako jesu treba ispisati riječ DA i najveću od tri vrijednosti. Ako nisu, tada treba ispisati riječ NE i najmanju od te tri vrijednosti.

Kao posebnost treba istaknuti da broj 100, koji može doći u ulaznim podacima, nije dvoznamenkast broj kako je traženo u zadatku.

Programski kod (pisan u Python 3)

```
P = int(input())
S = int(input())
A = int(input())
if 9 < P < 100 and 9 < S < 100 and 9 < A < 100:
    print('DA')
    print(max(P, S, A))
else:
    print('NE')
    print(min(P,S,A))
```

Potrebno znanje: naredna odlučivanja

Kategorija: ad hoc

6.2. Zadatak: Gusari — Autor: Adrian Satja Kurdija, Prilagodio: Pavel Kliska

Četiri puta treba izvršiti sljedeći postupak:

- ispiši četvrtinu trenutnog broja dragulja,
- taj broj oduzmi od trenutnog broja dragulja.

Zadatak je originalno objavljen na školskom natjecanju za 6. razred 2014. godine. Za ovogodišnje natjecanje je prilagođen.

Potrebno znanje: naredba učitavanja i naredba ispisa, naredba ponavljanja, operator cjelobrojnog dijeljenja i oduzimanja

Kategorija: ad hoc

6.3. Zadatak: Čekaonica — Autor: Josip Klepec

Zadatak možemo riješiti na mnogo načina.

Opisat ćemo jedan od njih. Za svako slobodno sjedeće mjesto pronalazimo najbliže zauzeto sjedeće mjesto. Primijetite da nas ne zanima koje je to mjesto nego samo njegova udaljenost pa ovaj dio problema odgovara poznatom problemu traženja minimuma.

Sada je preostalo provjeriti je li ovo sjedeće mjesto najbolje koje smo do sada probali. To možemo znati ako održavamo sljedeće tri informacije (varijable) o dosad najboljem sjedećem mjestu: *udaljenost do najbližeg zauzetog, udaljenost do ruba, indeks*.

Drugi način kako riješiti zadatak temelji se na činjenici da će najbolje sjedeće mjesto uvijek biti na "sredini" između neka 2 zauzeta.

Potrebno znanje: naredba odlučivanja, naredba ponavljanja

Kategorija: ad hoc

7.1. Zadatak: Finska — Autor: Nikola Dmitrović

Dok god u stringu postoje dva uzastopna slova, treba ih zamijeniti jednim takvim slovom. Ovo treba ponoviti za svako slovo abecede.

Programski kod (pisan u Python 3)

```
N = int(input())
s = input()
alfabet = []
for i in range(26):
    alfabet += [chr(ord('a') + i)]

for slovo in alfabet:
    while s.count(slovo+slovo) > 0:
        s = s.replace(slovo+slovo, slovo)
print(s)
```

Potrebno znanje: string

Kategorija: ad hoc

7.2. Zadatak: Kocka — Autor: Gabrijel Jambrošić

Pogledamo li sve slučajeve broja na kocki te načina brojanja točkica, vidjet ćemo da postoji 18 mogućih brojeva A i B (ulazni podaci). Za svaki od tih slučajeva možemo jednostavno ispisati točno rješenje.

Drugi i elegantniji način bio bi da za svaki broj na kocki odredimo koliko se točkica nalazi na kojem položaju te za svaki od osam mogućih načina brojanja točkica zbrojimo točkice s odgovarajućih položaja i odredimo daje li taj način brojanja broj B ili ne.

Potrebno znanje: naredba grananja

Kategorija: ad hoc

7.3. Zadatak: Luka

Autor: Nikola Dmitrović
Prilagodio: Pavel Kliska

Jedno od rješenja je da između svih knjiga prvo potražimo najtanju knjigu crvenih korica te je stavimo na policu na prvo mjesto. Zatim potražimo najtanju knjigu bijelih korica te nju postavimo na drugo mjesto, a zatim pronađemo najtanju knjigu plavih koricu te je stavimo na treće mjesto. Ovih nekoliko koraka ponavljamo sve dok sve knjige ne složimo na policu.

Zadatak je originalno objavljen na školskom natjecanju za 7. razred 2013. godine. Za ovogodišnje natjecanje je prilagođen.

Potrebno znanje: naredba učitavanja, naredba ispisa, naredba ponavljanja, algoritam traženja minimuma

Kategorija: simulacija

8.1. Zadatak: Online

Autor: Nikola Dmitrović

Jedno od mogućih rješenja je da prvo generiramo niz od R * (R + 1) // 2 brojeva koji počinje s X, a svaki sljedeći broj je za D veći od prethodnog. Nakon toga odredimo zbroj zadnjih R brojeva u nizu.

Programski kod (pisan u Python 3)

```
X = int(input())
D = int(input())
R = int(input())
L = [X]
N = R * (R + 1) // 2
for i in range(1, N):
    L += [L[i - 1] + D]
print(sum(L[len(L) - R:]))
```

Potrebno znanje: nizovi, naredba ponavljanja

Kategorija: ad hoc

8.2. Zadatak: Picard

Autor: Nikola Dmitrović
Prilagodio: Pavel Kliska

Zadanu tablicu sa slovima možemo promatrati kao dvodimenzionalni niz znakova. Za svaki redak tablice trebamo provjeriti može li se u njega upisati zadana riječ. Kako bi to otkrili, simuliramo upisivanje te riječi u redak. Riječ počinjemo upisivati prvo od prve pozicije u retku, pa od druge i tako sve do broj_stupaca-duljina(rijeci)+1 pozicije. Slovo iz riječi možemo upisati u polje retka ako je to polje prazno ('*') ili na tom polju već piše to slovo (ovu situaciju moramo prepoznati i pamtiti koliko se takvih podudaranja dogodilo pri simulaciji upisa riječi u redak od te pozicije te istovremeno pratiti poziciju na kojoj se dogodio maksimalan broj podudaranja). Kada uspoređujemo dvije različite pozicije na kojima možemo upisati riječ prvo uspoređujemo broj podudaranja na te dvije pozicije. Ako je jednak, promatramo brojeve redaka, a ako su i oni jednaki brojeve stupaca.

Bilješka: Sličan zadatak je originalno objavljen na županijskom natjecanju za 7. razred 2012. godine. Za ovogodišnje natjecanje je prilagođen.

Potrebno znanje: dvodimenzionalno polje znakova (ili polje stringova), naredba ponavljanja, naredba odlučivanja, algoritam traženja minimuma i maksimuma

Kategorija: ad hoc, simulacija

8.3. Zadatak: Dostava Autor: Stjepan Požgaj

Na početku možemo primijetiti da će nam za optimalni raspored dostava uvijek biti dovoljan samo jedan dostavljač. Ograničenja u zadatku omogućuju dobivanje svih bodova i bez te opservacije. U tom slučaju moramo isprobati sve moguće kombinacije podjela narudžbi među dostavljačima. Ako je K broj narudžbi postoji ukupno 2K raspodjela koje možemo generirati pomoću binarnih brojeva duljine K uz dozvoljene vodeće nule. Svaki takav broj predstavlja nam jednu raspodjelu: prvoj narudžbi pridružimo prvu znamenku binarnog broja, drugoj drugu i sve tako do K-te narudžbe. Ako je znamenka koja pripada nekoj narudžbi jedinica ćemo tu narudžbu dodijeliti Slavku, a u suprotnom će ju obavljati Mirko. Pretpostavit ćemo da sve dostave obavlja jedan dostavljač, općenitiji slučaj rješava se analogno. Potrebno je odrediti kojim redom će dostave biti obrađene. Najlakši način za to je pomoću funkcije next_permutation u C++-u ili pomoću sličnih funkcija u drugim jezicima.

Potrebno znanje: generiranje svih permutacija skupa

Kategorija: ad hoc