21. siječnja 2014.

Infokup 2013

Školsko natjecanje / Osnovna škola (5. i 6. i 7. i 8. razred)

Algoritmi (Basic/Python/Pascal/C/C++)

OBJAŠNJENJA ZADATAKA

Agencija za odgoj i obrazovanje
Education and Teacher Training Agency

MINISTARSTVO ZNANOSTI, OBRAZOVANJA I ŠPORTA REPUBLIKE HRVATSKE

5.1. Zadatak: Niko

Autor: Nikola Dmitrović[1]

Ukupnu količinu poklonjene čokolade u gramimima dobit ćemo ako zbrojimo sve dobivene količine čokolada. Znači, ukupno_čokolade = M grama mliječne čokolade + K grama čokolade s kokosom + L grama čokolade s lješnjacima.

Programski kod (pisan u psudojeziku)

```
ulaz(M);
ulaz(K);
ulaz(L);
izlaz(M + K + L);
```

Potrebno znanje: naredba učitavanja i naredba ispisa, poznavanje zbrajanja

Kategorija: ad hoc

5.2. Zadatak: Wonka

Autor: Nikola Dmitrović

Promotrimo što se dogodi kada Charlie otvori jednu kutiju.

Od B čokolada s bananom, koliko ih ima u jednoj kutiji, Chalie je putem pojeo njih X. Znači, ostalo je B-X čokolada s bananom. Od M čokolada s mangom, koliko ih ima u kutiji, Charlie je putem pojeo njih Y te je u kutiji ostalo M-Y čokolada s mangom. Na kraju priče, u jednoj kutiji preostalo je (B-X)+(M-Y) čokolada.

Na kraju treba uočiti da je Charlie na poklon dobio dva paketa čokolada te da opisani izraz (B-X)+(M-Y) treba pomnožiti s dva.

Programski kod (pisan u psudojeziku)

```
ulaz(B);
ulaz(M);
ulaz(X);
ulaz(Y);
izlaz(2 * (B - X) + (M - Y));
```

Potrebno znanje: naredba učitavanja i naredba ispisa, poznavanje zbrajanja i množenja

Kategorija: ad hoc

5.3. Zadatak: Mlijeko

Autor: Nikola Dmitrović

Zamislimo da ispred nas na stolu stoji **5 praznih posuda** na kojima piše *posuda_A*, *posuda_B*, *posuda_C*, *posuda_D* i *posuda_E*. Pri tome u *posudu_A* možemo uliti A litara mlijeka, itd. Isto tako zamislimo da ispred sebe imamo bocu s N litara mlijeka. Kako bi sada u stvarnosti riješili ovaj zadatak?

Za prvu posudu, *posuda_A* vrijedi:

---

[1] član Državnog povjerenstva, voditelj natjecanja u kategoriji primjena algoritama (programski jezik Basic/Python/Pascal/C/C++), profesor informatike u XV. gimnaziji, Zagreb

- ako u boci ima dovoljno mlijeka da se *posudu_A* napuni do vrha (N>=A) tada ćemo iz boce u *posudu_A* preliti A litara mlijeka. U boci će nam tada ostati N-A litara mlijeka;
- ako u boci nema dovoljno mlijeka da se *posudu_A* napuni do vrha (N<A) tada ćemo iz boce u *posudu_A* preliti onoliko mlijeka koliko ga je ostalo u boci (N). U boci će nam tada ostati 0 litara mlijeka.

Isto razmišljanje primjenit ćemo i na posudu_B, posudu_C, posudu_D i posudu_E.

Programski kod (pisan u psudojeziku)

```
ulaz(A); ulaz(B); ulaz(C); ulaz(D); ulaz(E);
posuda_A := 0; posuda_B := 0; posuda_C := 0; posuda_D := 0; posuda_E := 0;
ako je N >= A onda
{
    posuda_A := A; N := N - A;
}
inače
{
    posuda_A := N; N := 0;
}
ako je N >= B onda
{
    posuda_B := B; N := N - B;
}
inače
{
    posuda_B := N; N := 0;
}
ako je N >= C onda
{
    posuda_C := C; N := N - C;
}
inače
{
    posuda_C := N; N := 0;
}
ako je N >= D onda
{
    posuda_D := D; N := N - D;
}
inače
{
    posuda_D := N; N := 0;
}
posuda_E := N // ono što je ostalo u boci prelijeva se u posudu_E
izlaz(posuda_A);izlaz(posuda_B);izlaz(posuda_C);izlaz(posuda_D);izlaz(posuda_E);
```

Potrebno znanje: naredba učitavanja i ispisivanja, naredba odlučivanja (grananja)

Kategorija: ad hoc

6.1. Zadatak: Godine

Autor: Matija Milišić[2]

Marin danas ima Z godina, a na fotografiji je imao X godina. To znači da je od fotografiranja prošlo točno Z-X godina. Petar je na fotografiji imao Y godina, a danas, Z-X godina kasnije, ima Y+Z-X godina.

Programski kod (pisan u psudojeziku)

```
ulaz(X);
ulaz(Y);
ulaz(Z);
izlaz(Y+Z-X);
```

Potrebno znanje: naredba učitavanja i naredba ispisa, poznavanje zbrajanja i oduzimanja

Kategorija: ad hoc

6.2. Zadatak: Gusari

Autor: Adrian Satja Kurdija[3]

Četiri puta treba izvršiti sljedeći postupak:

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

Programski kod (pisan u psudojeziku)

```
ulaz(N);
za i:=1 do 4 činiti
{
      izlaz(N div 4);
      N := N - N div 4;
}
```

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

Kategorija: ad hoc

6.3. Zadatak: Foto

Autor: Nikola Dmitrović

Za svaku od N fotografija moramo provjeriti ima li na vanjskom tvrdom disku, u trenutku pokušaja spremanja, dovoljno prostora da se ona cijela spremi na njega. Ako nema, tada ćemo neku pomoćnu varijablu povećati za jedan. Ako ima, smanjit ćemo veličinu raspoloživog prostora na disku za veličinu te slike.

---

[2] student Fakulteta elektrotehnike i računarstva, osvajač brončane medalje na IOI 2011., srebrne medalje na CEOI 2011. te dviju brončanih medalja na IMO 2011. i IMO 2012, dvostruki državni prvak iz informatike (2. i 4. razred)

[3] univ. bacc. math., student PMF-MO, višestruki državni prvak iz matematike i informatike, osvajač srebrne i brončane medalje na IOI-ju te dviju brončanih medalja na IMO olimpijadi znanja

Programski kod (pisan u psudojeziku)

```
ulaz(disk);
ulaz(N);
koliko := 0;
za i:=1 do N činiti
{
      ulaz(slika);
      ako je disk - slika < 0 onda
            koliko := koliko + 1
      inače
            disk := disk – slika;
}
izlaz(koliko);
```

Potrebno znanje: naredba ponavljanja, naredba odlučivanja

Kategorija: ad hoc

7.1. Zadatak: Carina — Autor: Antun Razum[4]

Ako se vraćamo unatrag po granicama možemo postepeno promatrati koliki je novaca Marko imao prije nego je prošao pojedinu granicu. Očito je da je prije nego je prošao neku granicu imao duplo više nego nakon prelaska. Budući da je prešao 3 granice, Marko je na početku imao 2 * 2 * 2 = 8 puta više novaca nego na kraju. Potrebno je, dakle, ispisati 8 puta veći broj od učitanog.

Programski kod (pisan u psudojeziku)

```
ulaz(X);
izlaz(8 * X);
```

Potrebno znanje: naredba učitavanja i naredba ispisa, operator množenja

Kategorija: ad hoc

7.2. Zadatak: Ocijeni — Autor: Nikola Dmitrović

Za uspješno rješenje ovog zadatka nužno je točno prebrojiti koliko je zadanih zadataka bilo polovično riješeno, a koliko ih je bilo točno riješeno. Nakon što to odredimo, jednostavnim provjerama postavljenih uvjeta lako ćemo odrediti o kojoj je ocijeni riječ.

Programski kod (pisan u psudojeziku)

```
ulaz(N);
polovicnih := 0; cijelih := 0;
```

[4] student Fakulteta elektrotehnike i računarstva, osvajač srebrnih medalja na IOI 2012. i CEOI 2012., dvostruki državni prvak iz informatike (1. i 3. razred)

```
za i:=1 do N činiti
{
    ulaz(U, T);
    ako je T = U onda
        cijelih := cijelih + 1;
    ako je T > U div 2 i T <> U onda
        polovicnih := polovicnih + 1;
}
ako je polovicnih = 0 i cijelih = 0 onda
    izlaz('nedovoljan');
ako je polovicnih > 0 i cijelih = 0 onda
    izlaz ('dovoljan');
ako je polovicnih = 0 i cijelih = 1 onda
    izlaz ('dobar');
ako je polovicnih > 0 i cijelih = 1 onda
    izlaz ('vrlo dobar');
ako je cijelih >= 2 onda
    izlaz ('odlican');
```

Potrebno znanje: naredba ponavljanja, naredba odlučivanja

Kategorija: ad hoc

7.3. Zadatak: Cedeovi — Autor: Adrian Satja Kurdija

Treba nam niz riječi (stringova) koji će pamtiti koji CD se nalazi na mjestima 1, 2, ..., N u držaču. Na početku, jasno, taj niz redom popunimo stringovima zadanima na ulazu. Treba nam i dodatna varijabla u kojoj držimo ime CD-a koji trenutno svira.

Potom za svaki od unesenih K imena CD-a treba pronaći mjesto s kojeg ćemo taj CD izvući iz držača. To mjesto pronalazimo for-petljom koja prolazi po mjestima od 1 do N provjeravajući je li na tom mjestu CD koji tražimo. Kada pronađemo traženo mjesto, na njega upisujemo ime CD-a iz dodatne varijable (koji je dosad svirao), a ime CD-a koji smo tražili (i izvukli iz držača) upisujemo u dodatnu varijablu jer on sada svira. Ako traženi CD nismo pronašli u držaču, ne trebamo učiniti ništa jer on već svira.

Na koncu ispišemo niz koji predstavlja CD-ove u držaču.

Programski kod (pisan u psudojeziku)

```
ulaz(N);
za i := 1 do N činiti
  ulaz(drzac[i]);
ulaz(cd_koji_svira);
```

```
ulaz(K);
za j := 1 do K činiti
{
      ulaz(sljedeci_cd);
      za i := 1 do N činiti
            ako je drzac[i] = sljedeci_cd onda
            {
                  drzac[i] := cd_koji_svira;
                  cd_koji_svira := sljedeci_cd;
            }
za i := 1 do N činiti
      izlaz(drzac[i]);
```

Potrebno znanje: nizovi, naredba ponavljanja, naredba odlučivanja

Kategorija: ad hoc

8.1. Zadatak: Mlijeko — Autor: Nikola Dmitrović

Vidi zadatak 5.3.

8.2. Zadatak: Prosjek — Autor: Matija Milišić

Rješenje zadatka možemo podijeliti u dva dijela: prvi dio u kojem računamo prosjek temperatura i drugi u kojem tražimo prvi dan s temperaturom najbližom prosječnoj.

Prosjek dobivamo tako da prođemo kroz sve temperature, izračunamo njihov zbroj, a onda taj zbroj podijelimo s brojem dana N.

U drugom dijelu koristimo varijablu 'dan' u kojoj pamtimo redni broj traženog dana. Na početku postavimo 'dan' na 1. Prolazimo kroz sve dane i ako je trenutni dan bliži prosjeku, u varijablu 'dan' stavimo novu vrijednost. Udaljenost temperature nekog dana od prosjeka računamo kao apsolutnu vrijednost razlike temperature dana i prosjeka.

Ispišemo varijablu 'dan' i temperaturu tog dana.

Programski kod (pisan u psudojeziku)

```
ulaz(N);
za i := 1 do N činiti
      ulaz(temp[i]);
zbroj := 0;
za i := 1 do N činiti
      zbroj := zbroj + temp[i];
prosjek := zbroj/N;
dan := 1;
```

```
za i := 2 do N činiti
      ako je abs(temp[i]-prosjek) < abs(temp[dan]-prosjek) tada
            dan := i;
izlaz(dan);
izlaz(temp[dan]);
```

Potrebno znanje: nizovi, naredba ponavljanja, algoritam traženja minimalne vrijednosti

Kategorija: ad hoc

8.1. Zadatak: Bakterije — Autor: Antun Razum

Kako bi riješili ovaj zadatak bit će nam potrebne dvije matrice. Jedna u kojoj ćemo držati prethodno stanje preparata (p) i jedna u koju ćemo spremati novo stanje (novo). U svakom satu ćemo spremiti novo stanje u odgovarajuću matricu nakon čega ćemo tu matricu kopirati u matricu za prethodno stanje. Ovaj postupak ponovit ćemo K puta kako bi dobili stanje preparata nakon K sati.

Iduće stanje matrice računat ćemo na sljedeći način. Proći ćemo po svim poljima matrice, a zatim ćemo za svako polje proći po svim njegovim susjedima. Kako bi prošli kroz susjede na jednostavan način, koristit ćemo pomoćne nizove u kojima će nam za svaki od 8 osnovnih smjerova biti spremljeni pomaci za redak i stupac (vidi pseudokod). Tako ćemo prebrojati koliko neka ćelija ima susjeda i koliko bakterija u susjednim ćelijama svijetli. Nakon što smo to prebrojali sada samo za trenutnu ćeliju ispitamo je li zadovoljen uvjet da ona svijetli tj. vrijedi li da barem pola njenih susjednih bakterija svijetli. Ukoliko je uvjet ispunjen polje u novoj matrici postavimo tako da ta bakterija svijetli, a u suprotnom postavimo ga da ne svijetli.

Očigledno je da ovaj postupak s jednom matricom ne bi funkcionirao jer bi tijekom izvršavanja postupka neke bakterije promijenile stanje što bi onemogućilo daljnjim bakterijama pristup prethodnom stanju te bakterije koje je potrebno za izračun novog stanja. Postupak stoga radimo s dvije matrice. Isto tako mogli smo ga napraviti s K matrica tako da bi i-ta od njih predstavljala stanje preparata u i-tom satu. Dovoljno je koristiti samo dvije matrice jer kako bi izračunali novo stanje potrebno nam je samo prethodno stanje.

Programski kod (pisan u psudojeziku)

```
POMAK_RETKA := [-1, -1, 0, 1, 1, 1, 0, -1];
POMAK_STUPCA := [0, 1, 1, 1, 0, -1, -1, -1];
ulaz(N, M, K);
ulaz(p); // varijabla p je tipa dvodimenzionalno polje
za i := 1 do K činiti
{
    za red := 1 do N činiti
        za stupac := 1 do M činiti
        {
            broj_svjetlecih := 0;
```

```
            broj_susjeda := 0;
            za smjer := 1 do 8 činiti
            {
                novi_red := red + POMAK_RETKA[smjer];
                novi_stupac := stupac + POMAK_STUPCA[smjer];
                ako (novi_red >= 1) i (novi_red <= n) i
                                    (novi_stupac >= 1) i (novi_stupac <= m) onda
                {
                    broj_susjeda := broj susjeda + 1;
                    ako p[novi_red][novi_stupac] = '#' onda
                        broj_svjetlecih := broj_svjetlecih + 1;
                }
            }
            ako (2 * broj_svjetlecih >= broj_susjeda) onda
                novo[red][stupac] := '#'
            inače
                novo[red][stupac] = '.';
    za red := 1 do N činiti
        za stupac := 1 do M činiti
            p[red][stupac] := novo[red][stupac];
}
izlaz(p);
```

Potrebno znanje: dvodimenzionalno polje, višestruka naredba ponavljanja, naredba odlučivanja

Kategorija: ad hoc simulacija