Romanian Rhapsodies

Vremensko ograničenje	Memorijsko ograničenje	ulaz	izlaz
1 s	64 MB	standardni ulaz	standardni izlaz

Napisati program kojim se na osnovu temperature vode određuje njeno agregatno stanje. Ako je temperatura:

- veća od $0\circ$ C i manja od $100\circ$C - agregatno stanje je tečno
- ne veća od $0\circ$C - agregatno stanje je čvrsto,
- ne manja od $100\circ$C - agregatno stanje je gasovito.

Za temperaturu od tačno $0\circ$ smatra se da je agregatno stanje čvrsto, a za tačno $100\circ$ da je gasovito.

Ulaz

Temperatura - ceo broj od -100 do 200.

Izlaz

Na standardni izlaz ispisati jednu od sledećih reči: `cvrsto`, `tecno`, `gasovito`.

Primer

Ulaz

```
-10
```

Izlaz

```
cvrsto
```

REŠENJE 1 – ugnježdeno grananje

Python

```
t = int(input())

if t <= 0:
   print("cvrsto")
elif t < 100:
   print("tecno")
else:
```

```
    print("gasovito")
```

REŠENJE 2 – ugnježđeno grananje i logički operator **and** koji za promenljivu **t** proverava pripadnost intervalu

```
t = int(input())

if (t <= 0):
    print("cvrsto")
if (t > 0 and t < 100):
    print("tecno")
if (t >= 100):
    print("gasovito")
```

Chamber Symphony

Vremensko ograničenje	Memorijsko ograničenje	ulaz	izlaz
1 s	64 MB	standardni ulaz	standardni izlaz

Napiši program koji za dati redni broj meseca i godinu određuje broj dana u tom mesecu. Voditi računa o tome da li je godina prestupna (godina je prestupna ako je deljiva sa 4, a nije deljiva sa 100, osim ako je deljiva sa 400, kada jeste prestupna).

Ulaz

Sa standardnog ulaza učitavaju se dva broja:

- broj meseca m ($1 \leq m \leq 12$) i
- broj godine g ($1900 \leq g \leq 2100$)

Izlaz

Na standardni izlaz ispisati jedan ceo broj koji predstavlja broj dana u zadatom mesecu.

Primer 1

Ulaz

```
11
```

```
2014
```

Izlaz

```
30
```

Primer 2

Ulaz

```
2
2014
```

Izlaz

```
28
```

Primer 3

Ulaz

```
2
2016
```

Izlaz

```
29
```

Primer 3

Ulaz

```
2
2000
```

Izlaz

```
29
```

Primer 4

Ulaz

```
2

2100
```

Izlaz

```
28
```

REŠENJE 1

```
# ucitavamo mesec i godinu
mesec = int(input())
godina = int(input())
# odredjujemo broj dana u tom mesecu
brojDana = 0
# januar, mart, maj, jul, avgust, oktobar, decembar
if mesec == 1 or mesec == 3 or mesec == 5 or mesec == 7 or mesec == 8 or mesec == 10 or mesec == 12:
  brojDana = 31
# april, jun, septembar, novembar
elif mesec == 4 or mesec == 6 or mesec == 9 or mesec == 11:
    brojDana = 30
#februar
else:
     brojDana = 29 if (godina % 4 == 0 and godina % 100 != 0) or (godina % 400) == 0 else 28
# ispisujemo rezultat
print(brojDana)
```

REŠENJE 2

```
mesec = int(input())
godina = int(input())

# januar, mart, maj, jul, avgust, oktobar, decembar
if mesec in {1, 3, 5, 7, 8, 10, 12}:
   brojDana = 31
# april, jun, septembar, novembar
elif mesec in {4, 6, 9, 11}:
   brojDana = 30
#februar
else:
   if (godina % 4 == 0 and godina % 100 != 0) or godina % 400 == 0:
      brojDana = 29
   else:
      brojDana = 28

print(brojDana)
```

REŠENJE 3:

```
# ucitavamo mesec i godinu

mesec = int(input())
```

```
godina = int(input())


# broj dana u svakom mesecu

brojDanaUMesecu = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
brojDana = brojDanaUMesecu[mesec]
if mesec == 2 and  ((godina % 4 == 0 and godina % 100 != 0) or godina % 400 == 0):
   brojDana = brojDana + 1


# ispisujemo rezultat

print(brojDana)
```

Symphony No.5

Vremensko ograničenje	Memorijsko ograničenje	ulaz	izlaz
1 s	64 MB	standardni ulaz	standardni izlaz

Napiši program koji na osnovu broja osvojenih gemova dva igrača određuje ishod seta u tenisu.

Ulaz

Sa standardnog ulaza se unose dva prirodna broja (razdvojena razmakom) koji predstavljaju broj osvojenih gemova svakog igrača redom.

Izlaz

Ako je prvi igrač osvojio set na standardni izlaz ispisati poruku `pobedio prvi`. Ako je drugi igrač osvojio set ispisati `pobedio drugi`. Ako uneti rezultat neispravan ispisati `neispravno`. Ako set još nije završen ispisati `nije zavrseno`.

Primer 1

Ulaz

```
7

5
```

Izlaz

```
pobedio prvi
```

Primer 2

Ulaz

```
3
7
```

Izlaz

```
neispravno
```

REŠENJE

Treba da proverimo

da li je prvi igrač pobedio drugog,

a zatim i da li je drugi igrač pobedio prvog.

Ove dve provere su zapravo identične, pogodno je definisati funkciju kojom proveravamo da li je igrač koji je osvojio a gemova pobedio igrača koji je osvojio b gemova, a onda da je pozovemo tako što prosledimo broj osvojenih gemova prvog i drugog igrača, a zatim broj gemova drugog i prvog igrača.

Definicija te funkcije je jednostavna (igrač sa osvojenih a gemova je pobedio ako i samo ako je osvojio 6 gemova, dok je onaj drugi osvojio najviše 4 gema ili je osvojio 7 gemova, dok je onaj drugi osvojio 5 ili 6 gemova).

Provera ispravnosti rezultata je malo komplikovanija.
I nju ćemo implementirati u posebnoj funkciji.
Jednostavnosti radi, možemo sortirati dva broja osvojenih gemova, tako da znamo koji je od ta dva broja manji, a koji je veći. Niko ne može da ima više od 7osvojenih gemova, tako da ako je veći broj veći od 7 funkcija odmah može da vrati `false` (čime se konstatuje da je rezultat neispravan).

Ako je veći broj osvojio 7 gemova, manji broj je morao da osvoji ili 6 ili 5 (ako se to desilo funkcija može da vrati `true`, a u suprotnom vrednost `false`). U suprotnom su oba igrača osvojila najviše 6 gemova i u tom slučaju je svaki rezultat ispravan (ako je manji osvojio najviše 4gema, set je završen, a ako je osvojio 5 ili 6, još se igra), i funkcija može da vrati `true`.
Još jedna mogućnost je da definišemo funkciju koja proverava da li je set nezavršen. Opet je pogodno sortirati dva broja osvojenih bodova i zatim proveriti da li je veći broj manji od 6 ili je veći broj jednak 6, a manji broj je veći ili jednak 5.
Pošto pobeda prvog ili drugog igrača osigurava da je rezultat ispravan, ispravnost eksplicitno treba proveravati tek kada ustanovimo da niko još nije pobedio. Dakle, u glavnom programu proveravamo redom da li je prvi pobedio, u suprotnom da li je drugi pobedio, u suprotnom da li je rezultat neispravan i u suprotnom zaključujemo da set još nije završen. Za ovo je potrebno upotrebiti konstrukciju `elif` (ako bismo uslove ispitivali nezavisno mogli bismo, na primer, dobiti istovremeno i poruku da je prvi pobedio i da je rezultat ispravan).

REŠENJE

C++

```cpp
#include <iostream>
#include <algorithm>

using namespace std;

// da li je igrac a pobedio igraca b
bool pobedio(int a, int b) {
  return a == 6 && b <= 4 || // rezultati 6:4, 6:3, 6:2, 6:1 i 6:0
         a == 7 && (b == 5 || b == 6); // rezultati 7:6 i 7:5
}

// da li je trenutni rezultat a:b ispravan?
bool ispravno(int a, int b) {
  // odredjujemo veci i manji broj osvojenih gemova
  int veci = max(a, b), manji = min(a, b);
  // niko ne sme imati vise od 7 osvojenih gemova
  if (veci > 7) return false;
  // ako je neko osvojio 7 gemova, onaj drugi mora imati 5 ili 6
  if (veci == 7) return manji == 5 || manji == 6;
  // znamo da je veci osvojio <= 6 gemova, pa je toliko osvojio i
  // manji - sve takve kombinacije rezultata su ispravne
  return true;
}

int main() {
  int prvi, drugi;
  cin >> prvi >> drugi;
  if (pobedio(prvi, drugi))
    cout << "pobedio prvi" << endl;
  else if (pobedio(drugi, prvi))
    cout << "pobedio drugi" << endl;
  else if (ispravno(prvi, drugi))
    cout << "nije zavrseno" << endl;
  else
    cout << "neispravno" << endl;
  return 0;
}
```

Resenje 1 – bez potprograma

```
(prvi, drugi) = map(int, input().split())
   # odredjujemo veci i manji broj osvojenih gemova
veci = max(prvi, drugi); manji = min(prvi, drugi);
if (prvi == 6 and drugi <= 4 or # rezultati 6:4, 6:3, 6:2, 6:1 i 6:0
        prvi == 7 and (drugi == 5 or drugi == 6)):
   print("pobedio prvi")
elif (drugi == 6 and prvi <= 4 or # rezultati 6:4, 6:3, 6:2, 6:1 i 6:0
```

```
       drugi == 7 and (prvi == 5 or prvi == 6)):
   print("pobedio drugi")
elif  (veci >7) or (veci==7 and (manji <5 or manji==7)):
   print("neispravno")
else:
      print("nije zavrseno")
```

Python (sa potprogramima pobedio, ispravno)

```
# da li je igrac a pobedio igraca b
def pobedio(a, b):
  return (a == 6 and b <= 4 or # rezultati 6:4, 6:3, 6:2, 6:1 i 6:0
          a == 7 and (b == 5 or b == 6)) # rezultati 7:6 i 7:5

# da li je trenutni rezultat a:b ispravan?
def ispravno(a, b):
    # odredjujemo veci i manji broj osvojenih gemova
    veci = max(a, b); manji = min(a, b);
    # niko ne sme imati vise od 7 osvojenih gemova
    if veci > 7: return False
    # ako je neko osvojio 7 gemova, onaj drugi mora imati 5 ili 6
    if veci == 7: return manji == 5 or manji == 6
    # znamo da je veci osvojio <= 6 gemova, pa je toliko osvojio i
    # manji - sve takve kombinacije rezultata su ispravne
    return True

(prvi, drugi) = map(int, input().split())
if pobedio(prvi, drugi):
    print("pobedio prvi")
elif pobedio(drugi, prvi):
    print("pobedio drugi")
elif ispravno(prvi, drugi):
    print("nije zavrseno")
else:
    print("neispravno")
```