Stepeni

c++

```
#include <iostream>
#include <cassert>

using namespace std;

int main() {
  // ucitavamo uglove u stepenima i minutima
  int ugao1, ugao2;
  cin >> ugao1 >> ugao2;
  int ugao3 = 180 - (ugao1 + ugao2);
  assert(0 < ugao3 && ugao3 < 180);

  // ako je bar jedan ugao tup, trougao je tupougli
  if (ugao1 > 90 || ugao2 > 90 || ugao3 > 90)
    cout << "tupougli" << endl;
  // u suprotnom, ako je bar jedan ugao prav, trougao je pravougli
  else if (ugao1 == 90 || ugao2 == 90 || ugao3 == 90)
    cout << "pravougli" << endl;
  // u suprotnom nema pravih ni tupih uglova, pa je trougao ostrougli
  else
    cout << "ostrougli" << endl;

  return 0;
}
```

Python

```
# ucitavamo uglove
ugao1 = int(input())
ugao2 = int(input())
ugao3 = 180 - (ugao1 + ugao2)

# ako je bar jedan ugao tup, trougao je tupougli
if ugao1 > 90 or ugao2 > 90 or ugao3 > 90:
    print("tupougli")
# u suprotnom, ako je bar jedan ugao prav, trougao je pravougli
elif ugao1 == 90 or ugao2 == 90 or ugao3 == 90:
    print("pravougli")
# u suprotnom nema pravih ni tupih uglova, pa je trougao ostrougli
else:
    print("ostrougli")
```

Iks-oks

c++

```
#include <iostream>

using namespace std;

int main() {
  // dimenzija kvadrata
  const int a = 100;
  // koordinate piksela
  int x, y;
  cin >> x >> y;
  // redni broj vrste i kolone u kojoj se nalazi piksel
  int k = (x - 1) / a, v = (y - 1) / a;
  // redni broj kvadrata
  int kvadrat = 3 * v + k + 1;
  cout << kvadrat << endl;
  return 0;
}
```

```
#include <iostream>

using namespace std;

int main() {
  // koordinate piksela
  int x, y;
  cin >> x >> y;

  // redni broj kvadrata
  int kvadrat;

  // analiziramo sve slucajeve
  if (1 <= x && x <= 100 && 1 <= y && y <= 100)
    kvadrat = 1;
  if (101 <= x && x <= 200 && 1 <= y && y <= 100)
    kvadrat = 2;
  if (201 <= x && x <= 300 && 1 <= y && y <= 100)
    kvadrat = 3;
  if (1 <= x && x <= 100 && 101 <= y && y <= 200)
    kvadrat = 4;
  if (101 <= x && x <= 200 && 101 <= y && y <= 200)
    kvadrat = 5;
  if (201 <= x && x <= 300 && 101 <= y && y <= 200)
    kvadrat = 6;
  if (1 <= x && x <= 100 && 201 <= y && y <= 300)
    kvadrat = 7;
  if (101 <= x && x <= 200 && 201 <= y && y <= 300)
    kvadrat = 8;
  if (201 <= x && x <= 300 && 201 <= y && y <= 300)
    kvadrat = 9;
```

```
  // ispisujemo resenje
  cout << kvadrat << endl;
  return 0;
}
```

Python

```
# dimenzija kvadrata
a = 100
# koordinate piksela
x = int(input())
y = int(input())
# redni broj vrste i kolone u kojoj se nalazi piksel
k = (x - 1) // a
v = (y - 1) // a
# redni broj kvadrata
kvadrat = 3 * v + k + 1
print(kvadrat)
```

```
# koordinate piksela
x = int(input())
y = int(input())

# analiziramo sve slucajeve
if 1 <= x and x <= 100 and 1 <= y and y <= 100:
    kvadrat = 1
if 101 <= x and x <= 200 and 1 <= y and y <= 100:
    kvadrat = 2;
if 201 <= x and x <= 300 and 1 <= y and y <= 100:
    kvadrat = 3;
if 1 <= x and x <= 100 and 101 <= y and y <= 200:
    kvadrat = 4;
if 101 <= x and x <= 200 and 101 <= y and y <= 200:
    kvadrat = 5;
if 201 <= x and x <= 300 and 101 <= y and y <= 200:
    kvadrat = 6;
if 1 <= x and x <= 100 and 201 <= y and y <= 300:
    kvadrat = 7;
if 101 <= x and x <= 200 and 201 <= y and y <= 300:
    kvadrat = 8;
if 201 <= x and x <= 300 and 201 <= y and y <= 300:
    kvadrat = 9;

# ispisujemo resenje
print(kvadrat)
```

Džudo

c++

```
#include <iostream>

using namespace std;

int main() {
  // broj dzudista u raznim kategorijama
  int broj_do_50 = 0;
  int broj_od_51_do_75 = 0;
  int broj_od_76 = 0;
  // ukupan broj dzudista
  int n;
  cin >> n;
  for (int i = 0; i < n; i++) {
    int tezina;
    cin >> tezina;
    if (tezina <= 50)
      broj_do_50++;
    else if (tezina <= 75)
      broj_od_51_do_75++;
    else
      broj_od_76++;
  }

  cout << broj_do_50 << endl;
  cout << broj_od_51_do_75 << endl;
  cout << broj_od_76 << endl;

  return 0;
}
```

Python

```
# broj dzudista u raznim kategorijama
broj_do_50 = 0
broj_od_51_do_75 = 0
broj_od_76 = 0
# ukupan broj dzudista
n = int(input())
for i in range(n):
    tezina = int(input())
    if tezina <= 50:
        broj_do_50 += 1
    elif tezina <= 75:
        broj_od_51_do_75 += 1
    else:
        broj_od_76 += 1
print(broj_do_50)
print(broj_od_51_do_75)
print(broj_od_76)
```

Loto

c++

```
#include <iostream>

using namespace std;

int main() {
  int n;
  cin >> n;
  for (int b1 = 1; b1 <= n-2; b1++)
    for (int b2 = b1 + 1; b2 <= n-1; b2++)
      for (int b3 = b2 + 1; b3 <= n; b3++)
         cout << b1 << " " << b2 << " " << b3 << endl;
  return 0;
}
```

Python

```
n = int(input())
for b1 in range(1, (n-2)+1):
    for b2 in range(b1+1, (n-1)+1):
        for b3 in range(b2+1, n+1):
            print(b1, b2, b3)
```

Uglovi

c++

```
#include <iostream>
#include <cassert>

using namespace std;

// pretvara ugao dat u stepenima(s) i minutima (m) u
// ugao samo u minutima
int ugao(int s, int m) {
  return s * 60  + m;
}

int main() {
  // ucitavamo uglove u stepenima i minutima
  int ugao1_s, ugao1_m, ugao2_s, ugao2_m, ugao3_s, ugao3_m;
  cin >> ugao1_s >> ugao1_m;
  cin >> ugao2_s >> ugao2_m;

  // pretvaramo ih u uglove date samo u minutima
  int ugao1 = ugao(ugao1_s, ugao1_m);
  int ugao2 = ugao(ugao2_s, ugao2_m);
  int ugao3 = ugao(180, 0) - (ugao1 + ugao2);

  ugao3_s = ugao3 / 60; ugao3_m = ugao3 % 60;

  // prav ugao u minutima
  int pravUgao = ugao(90, 0);
  // ako je bar jedan ugao tup, trougao je tupougli
  if (ugao1 > pravUgao || ugao2 > pravUgao || ugao3 > pravUgao)
    cout << "tupougli" << endl;
  // u suprotnom, ako je bar jedan ugao prav, trougao je pravougli
  else if (ugao1 == pravUgao || ugao2 == pravUgao || ugao3 == pravUgao)
    cout << "pravougli" << endl;
  // u suprotnom nema pravih ni tupih uglova, pa je trougao ostrougli
  else
    cout << "ostrougli" << endl;

  return 0;
}
```

Python

```
# pretvara ugao dat u stepenima(s) i minutima (m) u
# ugao samo u minutima
def ugao(s, m):
    return s * 60 + m

# ucitavamo uglove u stepenima i minutima
ugao1_s = int(input())
ugao1_m = int(input())
ugao2_s = int(input())
ugao2_m = int(input())
# pretvaramo ih u uglove date samo u minutima
```

```
ugao1 = ugao(ugao1_s, ugao1_m)
ugao2 = ugao(ugao2_s, ugao2_m)
ugao3 = ugao(180, 0) - (ugao1 + ugao2)


# prav ugao u minutima
pravUgao = ugao(90, 0)
# ako je bar jedan ugao tup, trougao je tupougli
if ugao1 > pravUgao or ugao2 > pravUgao or ugao3 > pravUgao:
    print("tupougli")
# u suprotnom, ako je bar jedan ugao prav, trougao je pravougli
elif ugao1 == pravUgao or ugao2 == pravUgao or ugao3 == pravUgao:
    print("pravougli")
# u suprotnom nema pravih ni tupih uglova, pa je trougao ostrougli
else:
    print("ostrougli")
```

Sudoku

c++

```
#include <iostream>

using namespace std;

int main() {
  // dimenzije kvadratica
  const int a = 30;
  // koordinate piksela
  int x, y;
  cin >> x >> y;
  // vrsta i kolona u kojoj se nalazi kvadratic (na polju 9x9), brojano od 0
  int kolona = (x - 1) / a;
  int vrsta = (y - 1) / a;
  // vrsta i kolona u kojoj se nalazi kvadrat (na polju 3x3), brojano od 0
  int K = kolona / 3;
  int V = vrsta / 3;
  // redni broj kvadrata, brojano od 0
  int kvadrat = V * 3 + K;
  // ispis resenja (brojano od 1)
  cout << vrsta   + 1 << endl;
  cout << kolona  + 1 << endl;
  cout << kvadrat + 1 << endl;
  return 0;
}
```

Python

```
# dimenzije kvadratica
a = 30
# koordinate piksela
x = int(input())
y = int(input())
# vrsta i kolona u kojoj se nalazi kvadratic (na polju 9x9), brojano od 0
kolona = (x - 1) // a
vrsta = (y - 1) // a
# vrsta i kolona u kojoj se nalazi kvadrat (na polju 3x3), brojano od 0
K = kolona // 3
V = vrsta // 3;
# redni broj kvadrata, brojano od 0
kvadrat = V * 3 + K
# ispis resenja (brojano od 1)
print(vrsta + 1)
print(kolona + 1)
print(kvadrat + 1)
```

Kartice

c++

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

using namespace std;

int main() {
  int brojLosih = 0;
  int n, k;
  cin >> n >> k;
  for (int i = 0; i < n; i++) {
    vector<int> karte(k);
    for (int j = 0; j < k; j++)
      cin >> karte[j];
    if (!is_sorted(begin(karte), end(karte)))
      brojLosih++;
  }
  cout << brojLosih << endl;
  return 0;
}
```

Python

```
brojLosih = 0
n = int(input())
k = int(input())
for i in range(n):
  karte = [int(input()) for j in range(k)]
  if (not(all(karte[j] <= karte[j+1] for j in range(k-1)))):
      brojLosih += 1
print(brojLosih)
```

Prag

c++

```
#include <iostream>
#include <vector>

using namespace std;

int main() {
  int n;
  cin >> n;
  vector<int> poeni(n);
  for (int i = n-1; i >= 0; i--)
    cin >> poeni[i];

  int m;
  cin >> m;
  for (int i = 0; i < m; i++) {
    int prag;
    cin >> prag;
    int broj = distance(lower_bound(begin(poeni), end(poeni), prag), end(poeni));
    cout << broj << endl;
  }

  return 0;
}


#include <iostream>
#include <vector>

using namespace std;

int prvi_veci_ili_jednak(const vector<int>& a, int x) {
  int l = 0, d = a.size()-1;
  while (l <= d) {
    int s = l + (d - l) / 2;
    if (a[s] < x)
      l = s + 1;
    else
      d = s - 1;
  }
  return d + 1;
}

int main() {
  int n;
  cin >> n;
  vector<int> poeni(n);
  for (int i = n-1; i >= 0; i--)
    cin >> poeni[i];

  int m;
  cin >> m;
```

```
  for (int i = 0; i < m; i++) {
    int prag;
    cin >> prag;
    cout << n - prvi_veci_ili_jednak(poeni, prag) << endl;
  }

  return 0;
}
```

Python

```
import bisect

n = int(input())
poeni = list(map(int, input().split()))
poeni.reverse()
m = int(input())
pragovi = map(int, input().split())
for prag in pragovi:
  print(n - bisect.bisect_left(poeni, prag))
```

Par-nepar

c++

```
#include <iostream>

using namespace std;

int main() {
  int brojLosih = 0;
  int n, k;
  cin >> n >> k;
  for (int i = 0; i < n; i++) {
    bool uRedu = true;
    bool bilaNeparna = false;
    for (int j = 0; j < k; j++) {
      int karta;
      cin >> karta;
      if (karta % 2 == 1)
      bilaNeparna = true;
      else if (bilaNeparna)
      uRedu = false;
    }
    if (!uRedu)
      brojLosih++;
  }
  cout << brojLosih << endl;
  return 0;
}
```

Python

```
brojLosih = 0
n = int(input())
k = int(input())
for i in range(n):
  uRedu = True
  bilaNeparna = False
  for j in range(k):
    karta = int(input())
    if karta % 2 == 1:
        bilaNeparna = True
    elif bilaNeparna:
        uRedu = False
  if not(uRedu):
    brojLosih += 1
print(brojLosih)
```

Numeracija

c++

```
#include <iostream>

using namespace std;

int main() {
  int n;
  cin >> n;
  int broj = 0;
  // krecemo od jednocifrenih brojeva
  int s = 10;  // interval [s/10, s-1] tj. [1, 9]
  int d = 1;   // broj cifara je 1
  while (s - 1 < n) {
    // dodajemo cifre koriscene za zapis brojeva u intervalu [s/10, s-1]
    broj += ((s - 1) - s/10 + 1) * d; // u intervalu [a, b] ima (a - b + 1) brojeva
    // prelazimo na sledeci interval, tj. brojeve sa jednom cifrom vise
    s *= 10;
    d += 1;
  }
  // preostali su brojevi u intervalu [s/10, n]
  // dodajemo njihove cifre
  broj += (n - s/10 + 1) * d;
  cout << broj << endl;
  return 0;
}
```

Python

```
n = int(input())
broj = 0
# krecemo od jednocifrenih brojeva
s = 10  # interval [s/10, s-1] tj. [1, 9]
d = 1   # broj cifara je 1
while s - 1 < n:
    # dodajemo cifre koriscene za zapis brojeva u intervalu [s/10, s-1]
    broj += ((s - 1) - s//10 + 1) * d # u intervalu [a, b] ima (a - b + 1) brojeva
    # prelazimo na sledeci interval, tj. brojeve sa jednom cifrom vise
    s *= 10
    d += 1

# preostali su brojevi u intervalu [s/10, n]
# dodajemo njihove cifre
broj += (n - s//10 + 1) * d
print(broj)
```

Origami

c++

```
#include <iostream>

using namespace std;

int main() {
  int a, b;
  cin >> a >> b;
  int bk = 0;
  while (b != 0) {
    bk += a / b;
    int ost = a % b;
    a = b;
    b = ost;
  }
  cout << bk << endl;
}
```

Python

```
a = int(input())
b = int(input())
bk = 0
while b != 0:
    bk = bk + a // b
    ost = a % b
    a = b
    b = ost
print(bk)
```

Koferi

c++

```
#include <iostream>
#include <vector>

using namespace std;

int main() {
  // ubrzavamo ucitavanje
  ios_base::sync_with_stdio(false);

  // ucitavamo trazeni zbir
  int trazeniZbir;
  cin >> trazeniZbir;

  // izracunavamo parcijalne sume elemenata niza
  int n;
  cin >> n;
  vector<int> S(n+1);
  for (int i = 0; i < n; i++) {
    int x;
    cin >> x;
    S[i+1] = S[i] + x;
  }

  // u sortiranom nizu parcijalnih suma trazimo da li postoje dva
  // elementa cija je razlika jednaka trazenom zbiru
  int l = 0, d = 1;
  while (d <= n) {
    if (S[d] - S[l] < trazeniZbir) {
      d++;
    } else if (S[d] - S[l] > trazeniZbir) {
      l++;
    } else {
      cout << l << endl;
      l++;
    }
  }
  return 0;
}
```

```
#include <iostream>

using namespace std;

int main() {
  // ubrzavamo ucitavanje
  ios_base::sync_with_stdio(false);

  // ucitavamo trazeni zbir
  int trazeniZbir;
```

```
  cin >> trazeniZbir;

  // ucitavamo elemente niza
  int n;
  cin >> n;
  int a[50000];
  for (int i = 0; i < n; i++)
    cin >> a[i];

  // granice segmenta
  int i = 0, j = 0;
  // zbir segmenta
  int zbir = a[0];
  while (true) {
    // na ovom mestu vazi da je zbir = sum(ai, ..., aj) i da
    // za svako i <= j' < j vazi da je sum(ai, ..., aj') < trazeniZbir

    if (zbir < trazeniZbir) {
      // prelazimo na interval [i, j+1]
      j++;
      // ako takav interval ne postoji, zavrsili smo pretragu
      if (j >= n)
         break;
      // izracunavamo zbir intervala [i, j+1] na osnovu zbira intervala [i, j]
      zbir += a[j];
    } else {
      // ako je zbir jednak trazenom, vazi da je sum(ai, ..., aj) = trazeniZbir
      // pa prijavljujemo interval
      if (zbir == trazeniZbir)
            cout << i << endl;
      // prelazimo na interval [i+1, j]
      // izracunavamo zbir intervala [i+1, j] na osnovu zbira intervala [i, j]
        zbir -= a[i];
        i++;
    }
  }

  return 0;
}
```

Python

```
trazeniZbir = int(input())
n = int(input())
a = list(map(int, input().split()))
i = 0
j = 0
zbir = a[0]
while True:
    if zbir < trazeniZbir:
      j += 1
      if j >= n:
         break;
      zbir += a[j]
    else:
      if zbir == trazeniZbir:
         print(i)
      zbir -= a[i]
      i += 1
```