Statistix.

Szkodliwy wirus zniszczył repozytorium z kodami źródłowymi pewnego popularnego pakietu programów biurowych. Wirus nie oszczędził żadnej działającej kopii tego oprogramowania. Zapanował chaos. Bez tego jakże popularnego oprogramowania gospodarka światowa będzie pogrążać się w coraz głębszej recesji.

Zostaliście Państwo poproszeni o pomoc w pracach nad odtworzeniem podstawowej funkcjonalności tego pakietu. Wszyscy mieszkańcy Błękitnej Planety z niecierpliwością oczekują na napisane przez Państwa moduły. Nie zawiedźcie pokładanych w Was nadziei!

1 Wartość oczekiwana

Państwa zadaniem jest napisanie funkcji o nazwie **expval**. Argumentami tej funkcji są kolejno:

- liczba typu `unsigned int`, nazwijmy ją `n`
- wskaźnik do stałej typu `double`, nazwijmy go `x`
- wskaźnik do stałej typu `double`, nazwijmy go `p`
- wskaźnik do funkcji o jednym argumencie typu `double` oraz o wartości typu `double`. Nazwijmy ten wskaźnik `f`
- referencja do zmiennej typu `double`, nazwijmy ją `r`

Zakładamy, że pod adresem `x` znajduje się tablica o rozmiarze `n+1`, natomiast pod adresem `p` znajduje się tablica o rozmiarze `n`. Elementy tablicy `p[i]` powinny spełniać oba poniższe warunki:

- `p[i]>=0` dla `i=0,1,2,...,n-1`
- `p[0]+...+p[n-1]<1`

Jeśli którykolwiek z podanych warunków nie jest spełniony, to funkcja ma zwrócić wartość `false` i nie zmieniać wartości zmiennej `r`. Jeżeli prawdopodobieństwa `p[i]` są poprawne (spełniają oba powyższe warunki), to funkcja **expval** ma obliczyć wartość wyrażenia

```
f(x[0])p[0]+... +f(x[n-1])p[n-1] + f(x[n])(1-p[0]-...-p[n-1])
```

jako liczbę typu `double` i podstawić wynik do zmiennej `r`. W tym przypadku wartością funkcji jest `true`.

2 Mediana warunkowa.

Mediana niemalejącego skończonego ciągu liczb rzeczywistych $a_1, a_2, \ldots, a_n$ to

- liczba $a_{(n+1)/2}$ gdy n jest liczbą nieparzystą,
- średnia liczb $a_{n/2}$ oraz $a_{n/2+1}$, gdy n jest liczbą parzystą.

Na przykład mediana ciągu $1, 1, 7, 9, 12$ to 7, a mediana ciągu $1, 3, 9, 12$ to $(3 + 9)/2 = 6$. Medianę skończonego zbioru liczb (w zbiorze elementy nie mogą się powtarzać i ich kolejność nie jest określona) definiujemy jako medianę uporządkowanego rosnąco ciągu elementów tego zbioru. Zatem mediany zbiorów

$$\{1, 3, 9, 12\}, \qquad \{4, 8, 2, 9\}, \qquad \{9, 7, 3, 5\}$$

są takie same i równe 6. Państwa zadaniem jest napisanie funkcji o nazwie **median**, której argumentami są kolejno

- liczba typu `unsigned int`, nazwijmy ją `n`.
- wskaźnik do stałej typu `int`, nazwijmy go `t`. Zakładamy, że tablica `t` ma co najmniej `n` elementów.
- wskaźnik do funkcji o jednym argumencie typu `int` i wartości typu `int`. Nazwijmy ten wskaźnik do funkcji `f`.
- wskaźnik do funkcji o jednym argumencie typu `int` oraz o wartości typu `bool`. Nazwijmy ten wskaźnik `p`
- referencja to zmiennej typu `double`, nazwijmy ją `r`.

Zadaniem funkcji jest obliczenie mediany zbioru liczb

`Z =` { `f(t[i]) : p(t[i])=true` }.

Czyli obliczamy medianę zbioru wartości funkcji `f(t[i])` dla argumentów `t[i]` spełniających warunek `p`. Możliwe są dwa przypadki:

1. zbiór `Z` jest pusty – wtedy funkcja **median** ma zwrócić wartość `false` i nie zmieniać referencji `r`;
2. zbiór `Z` jest niepusty – wtedy funkcja **median** oblicza medianę zbioru `Z`, wstawia wynik do referencji `r` i zwraca `true`.

Uwaga: w zbiorze elementy nie mogą się powtarzać.

3 Największy wspólny dzielnik.

Państwa zadaniem jest napisanie funkcji o nazwie **gcd**. Jej argumentami mają by kolejno

- liczba typu `unsigned int`, nazwijmy ją `n`.
- wskaźnik do stałej typu `int`, nazwijmy go `t`. Zakładamy, że tablica `t` ma co najmniej `n` elementów.
- wskaźnik typu `int` z domyślną wartością `nullptr`, nazwijmy ten wskaźnik `r`.

Zadaniem funkcji jest obliczenie największego wspólnego dzielnika zbioru liczb

`Z =` { `t[i] : t[i]!=0` }.

Możliwe są dwa przypadki:

- zbiór `Z` jest pusty – wtedy funkcja ma zwrócić wartość `0` jako liczbę typu `unsigned int` i nie może modyfikować elementów tablicy `r`;
- zbiór `Z` jest niepusty – wtedy funkcja oblicza największy wspólny dzielnik zbioru `Z` i zwraca go jako liczbę typu `unsigned int`. Ponadto, jeśli `r` nie jest wskaźnikiem zerowym, to zakładamy, że pokazuje na tablicę długości `n`. Wtedy należy do tablicy `r` przepisać (z zachowaniem kolejności) wszystkie elementy tablicy `t` podzielone przez obliczony największy wspólny dzielnik zbioru `Z`.

4 Zliczanie warunkowe.

Państwa zadaniem jest napisanie funkcji o nazwie **count**. Jej argumentami są kolejno

- liczba typu `unsigned int`, nazwijmy ją `n`.
- wskaźnik do stałej typu `int`, nazwijmy go `t`. Zakładamy, że tablica `t` ma co najmniej `n` elementów.
- wskaźnik do funkcji o dwóch argumentach typu `int` oraz o wartości typu `bool` z domyślną wartością `nullptr`. Nazwijmy ten wskaźnik `p`.

Jeśli wskaźnik `p` jest równy `nullptr`, to funkcja ma obliczyć i zwrócić liczbę elementów zbioru

`Z =` { `(t[i],t[j]) : i,j=0,...,n-1` }

jako liczbę typu `unsigned int`. Gdy podano funkcję `p` jako argument (`p` jest różne od `nullptr`), to należy zwrócić liczbę elementów zbioru

`Z =` { `(t[i],t[j]) : p(t[i],t[j])=true, i,j=0,...,n-1` }

Uwaga:
zliczamy liczbę różnych par elementów tablicy, a nie liczbę różnych par indeksów tablicy.

5 Rozwiązanie.

- do systemu BaCa należy wysłać jeden plik o nazwie `statistix.h` (wielkość liter ma znaczenie) zawierający implementację powyższych funkcji **(bez funkcji main)**.
- w pliku `statistix.h` zabronione jest użycie dyrektywy preprocesora `#include`
- pierwszą linią każdego z programów testujących będzie dyrektywa włączenia pliku `statistix.h`, po czym nastąpią testy sprawdzające zgodność implementacji z wymaganiami przedstawionymi w opisie każdej z funkcji
- programy testowe będą kompilowane z włączoną opcją `-std=c++11` (potrzebne dla `nullptr`)
- w rozwiązaniu można implementować dowolną liczbę funkcji pomocniczych

Test jawny:

```
#include "statistix.h"
#include <iostream>
using namespace std;

int f(int x) { return x*(x-10); }
bool TRUE(int x) { return true; }
bool EVEN(int x) { return x%2==0; }
bool ASYMMETRIC_REL(int a, int b) { return 3*a<b; }
double g(double x) { return 7*x-2; }

int main(){
  double r=0.0;
  int t[] = {6,30,12,-81,9,-9,15,6,30,33,21,18};
  unsigned n = sizeof(t)/sizeof(int);

  cout << boolalpha;
  cout << median(n,t,f,TRUE,r) << endl;
  cout << "median_all=" << r << endl;
  cout << median(n,t,f,EVEN,r) << endl;
  cout << "median_even=" << r << endl;
  cout << "gcd=" << gcd(n,t) << endl;
  cout << "count=" << count(n,t,ASYMMETRIC_REL) << endl;

  double x[] = {4,3,2,1};
  double p[] = {0.125,0.25,0.125};
  cout << expval(3,x,p,g,r) << endl;
  cout << "expval=" << r << endl;
}
```

Oczekiwane wyjście:

```
true
median_all=157.5
true
median_even=84
gcd=3
count=24
true
expval=12
```

Wszelkie uwagi/pytania dotyczące interpretacji treści zadania proszę zgłaszać na forum kursu.

Autor zadania: Daniel Wilczak