**Задача лічилка**

**Задача Йосипа Флавія** (*Проблема Йосифа Флавія*) — математична задача.

Задача виникла на основі легенди. [Йосип Флавій](http://uk.wikipedia.org/wiki/%D0%99%D0%BE%D1%81%D0%B8%D0%BF_%D0%A4%D0%BB%D0%B0%D0%B2%D1%96%D0%B9) був [римським](http://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%80%D0%BE%D0%B4%D0%B0%D0%B2%D0%BD%D1%96%D0%B9_%D0%A0%D0%B8%D0%BC) істориком, [євреєм](http://uk.wikipedia.org/wiki/%D0%84%D0%B2%D1%80%D0%B5%D1%97) за походженням. Дія легенди відбувалася під час [Першої іудейської війни](http://uk.wikipedia.org/w/index.php?title=%D0%9F%D0%B5%D1%80%D1%88%D0%B0_%D1%96%D1%83%D0%B4%D0%B5%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%B2%D1%96%D0%B9%D0%BD%D0%B0&action=edit&redlink=1). [Легіон](http://uk.wikipedia.org/wiki/%D0%9B%D0%B5%D0%B3%D1%96%D0%BE%D0%BD) із 41 сикаріїв, що обороняв галілейський замок [Массада](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B0%D0%B4%D0%B0), не хотів здаватись в полон римляням. [Сикарії](http://uk.wikipedia.org/wiki/%D0%A1%D1%96%D0%BA%D0%B0%D1%80%D1%96%D1%97%22%20%5Co%20%22%D0%A1%D1%96%D0%BA%D0%B0%D1%80%D1%96%D1%97) стали в коло й домовились, що кожні два воїни будуть убивати третього, доки не загинуть всі. Самогубство - тяжкий гріх, але той, хто врешті-решт залишиться останнім, мусить це зробити. Йосип Флавій, командир цього легіону, нібито розрахував, де йому та його другу потрібно стати, щоб залишитись останніми, але не для того щоб вбити друга, а щоб здати замок римлянам.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Завдання** | **Масиви**  | **Динамічна пам’ять** | **Формула** | **Тип даних вектор** |
| Методом копіювання програмного коду створити проекти | #include<iostream>using namespace std;int main(){ int a[100000];int n,n0,k,k0,i,j;cin>>n>>k;for (i=1;i<=n;i++)a[i]=i;n0=n;k0=1;while (n>1){ cout<<"n="<<n<<" k0= "<<k0<<endl;if(k0+k<=n+1)k0=k0+k-1; else k0=(k%n)-(n-k0)-1;n=n-1;for(j=k0;j<=n; j++)a[j]=a[j+1];for (j=1;j<=n;j++)cout<<a[j]<<" ";cout<<endl;}} | # include <iostream># include <conio.h> // бібліотека з функцією GETCH using namespace std;struct Titem {// Створюємо структуру "Titem" int n; Titem \* next, \* prev;} \* first, \* s, \* p; int n, m; void add\_item (int v) { // Описуємо функцію "add\_item" додає в список порядкові номери учасників Titem \* p; if (s == NULL && first == NULL) { s = new Titem; s-> n = v; s-> next = NULL; s-> prev = NULL; first = s; } else {  p = new Titem; p-> n = v; p-> next = NULL; p-> prev = s; s-> next = p; s = p; }return;} int main () { int i; cout << "n ="; cin >> n; cout << "m ="; cin >> m; for (i = 0; i <n; i++) // Запускаємо функцію "add\_item" add\_item (i +1); s-> next = first; // Замикає список first-> prev = s; s = first; while (n> 1) {// Із списку з порядковими номерами учасників видаляємо m-ий // за рахунком стільки разів, поки не залишиться один учасник// (після видалення учасника вважати починаємо з наступного після видаленого) p = s; for (i = 1; i <m; i++) p = p-> next; p-> next-> prev = p-> prev; // Перед видаленням m-ого елемента списку вибудовується свзь між попереднім p-> prev-> next = p-> next; // і наступним елементами списку s = p-> next; delete p; n--; } cout<<"Rez="<<s.n; // Виводимо на екран номер залишився учасника getch (); return 0;} | #include<iostream>using namespace std;long long n,res,i,k;int main(){cin>>n>>k;for(i=1;i<=n;i++)res=(res+k)%i;cout<<res+1<<endl;} | //Дано натуральні числа m і n. Передбачається, що n чоловік стають в коло// i отримують номери, вважаючи проти годинникової стрілки. 1, 2, 3, ... n.// Потім починаючи з першого, також проти годинникової стрілки відлічується// М-ий чоловік (оскільки люди стоять по колу, то за N-им людиною стоїть перший).// Ця людина виходить з кола, після чого, починаючи з наступного,// Знову відраховується м-ий людина і так до тих пір, поки з усього кола// Не залишиться одна людина. Визначити його номер.#include <iostream>#include <vector>//////////////////////////////////////////////////////////////////////////////////////typedef std::vector<int> T\_nums;//////////////////////////////////////////////////////////////////////////////////////int get\_counting\_out\_last\_num(int m, int n){ T\_nums nums; //Рахуємо, щоо контейнер заповнено проти годинникової стрілки. for(int i = 1; i <= n; ++i) { nums.push\_back(i); }  int pos\_cur = 0;   while(nums.size() > 1) { pos\_cur += m - 1; pos\_cur %= nums.size();  nums.erase(nums.begin() + pos\_cur);  } return nums.front();}int main(){ std::locale::global(std::locale("")); std::cout << "n = "; int n; std::cin >> n; std::cout << "m = "; int m; std::cin >> m; std::cout << "Залишився номер "  << get\_counting\_out\_last\_num(m, n) << std::endl;} |
| Протестувати задачі приN=5 k=3N=10 k=3N=100 k=3N=1000 k=3N=10000 k=3N=1000000 k=3 |  |  |  |  |