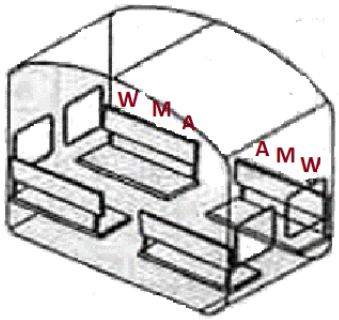
Алешин Илья Дмитриевич

Балл: 100

Задача №1

Условие

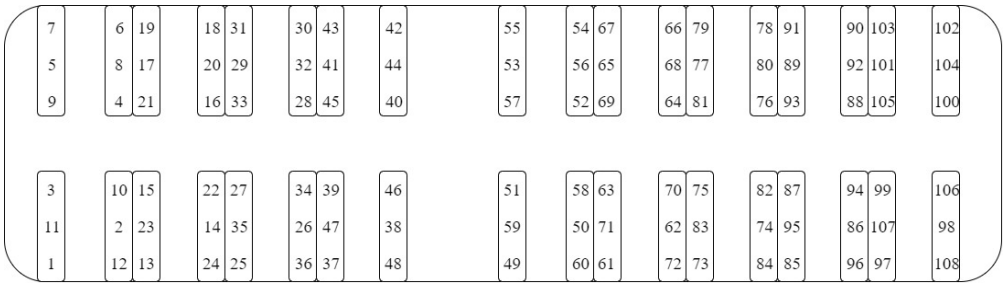
Витя и **Оля** очень любят путешествовать. В основном они путешествуют по железной дороге. Однажды они ехали в вагоне, где были только сидячие места, и заинтересовались расположением сидений в своем купе. Купе выглядело примерно так



Тип места обозначается следующим образом:

- Сиденье у окна: **W**
- Среднее место: **M**
- Место у прохода: **A**

Нумерация мест в вагоне приведена на следующем рисунке



Ребятам стало интересно узнать номер и тип сиденья, расположенного рядом с ними. В случае, когда сиденье расположено в середине, необходимо указать оба соседних номера.

Напишите программу, которая поможет ребятам решить эту задачу.

Входные данные

На вход подается одно целое число **N (**1 <= **N** <= 108**),** обозначающее номер места.

Выходные данные

Целое число, и большая латинская буква **W, M** или **A**, разделенные одним пробелом, которые будут соответствовать номеру и типу соседнего сиденья. Если таких сидений два, то данные по второму сидению выводится в той же строке через пробел.

Примеры

Входные данные	Выходные данные
19	17 M
29	31 W 33 A

Исходный код

```
#include <iostream>


int main()
{
 int n;
 std::cin >> n;
 int ost = n % 6;
 int del = n / 6;
 if ((del % 2 == 0 || (del % 2 == 1 && ost == 0)) && (!(del % 2 == 0 && ost == 0)))
 {
  if (ost == 1)
   std::cout << n + 10 << " M";
  if (ost == 2)
   std::cout << n + 10 << " W " << n + 8 << " A";
  if (ost == 3)
   std::cout << n + 8 << " M";
  if (ost == 4)
   std::cout << n + 4 << " M";
  if (ost == 5)
   std::cout << n + 2 << " W " << n + 4 << " A";
  if (ost == 0)
   std::cout << n + 2 << " M";
 }
 else {
  if (ost == 1)
   std::cout << n - 2 << " M";
  if (ost == 2)
   std::cout << n - 2 << " W " << n - 4 << " A";
  if (ost == 3)
   std::cout << n - 4 << " M";
  if (ost == 4)
   std::cout << n - 8 << " M";
  if (ost == 5)
   std::cout << n - 10 << " W " << n - 8 << " A";
  if (ost == 0)
   std::cout << n - 10 << " M";
 }

 return 0;
}
```

Задача №2

Условие

В кассе, продающей билеты по цене 500 р., нет возможности расплатиться картой, а в начале работы нет даже денег для сдачи. Перед кассой стоит очередь, у каждого человека в очереди есть только одна купюра. У части – 500 р., у части – 1000 р. Сколькими способами можно выстроить очередь так, чтобы у кассира всегда была сдача с тысячерублевой купюры?

Напишите программу, которая решит эту задачу.

Сначала на вход программы подаются два натуральных числа: количество человек $\boldsymbol{n}$ с купюрой 500 р, $1 < \boldsymbol{n} \leq 18$ и количество человек с купюрой 1000 р. $\boldsymbol{k}$, $1 < \boldsymbol{k} \leq 18$.

Выведите одно целое число – сколькими способами можно выстроить очередь.

Пример

Входные данные	Выходные данные
2 2	2

Исходный код

```
#include <iostream>

using namespace std;
long long cnt = 0;
int a, b;
void dfs(string s)
{
 if (s.size() == a + b)
 {
  int num1 = 0, num2 = 0;
  bool f = 0;
  for (int i = 0; i < s.size(); ++i)
  {
   if (s[i] == '5')
    num1++;
   else
    num2++;
   if (num2 > num1)
    f = 1;
  }
  if (f == 0 && num1 == a && num2 == b) {
   cnt++;
  }
  return;
 }
 dfs(s + "5");
 dfs(s + "1");
}

int main()
{
 cin >> a >> b;
 dfs("");
 cout << cnt;
 return 0;
}
```

Задача №3

Условие

Исполнитель Новемариум получает на вход натуральное число X (не превышающее 10^6). По этому числу, точнее по его представлению в девятеричной системе счисления строится новое число Y по следующим правилам.

В девятеричном представлении числа X зачеркиваются последние две цифры (гарантируется, что в девятеричном представлении числа X больше 2-х цифр).

1. Если получившееся число четное, то в конец полученной цепочки цифр дописывается две цифры – сумма оставшихся цифр (тоже в девятеричном представлении, причем если сумма состоит из одной цифры, то перед ней надо вписать 0).

   Например, $4862_{10} = 6602_7$. После зачеркивания остается 66_9 – четное число. Сумма оставшихся цифр $12_{10}=13_9$. Новое число $Y = 6613_9 = 4872_{10}$.

   Другой пример, $3140_{10} = 4268_9$, после зачеркивания остается четное число 42_9. Сумма оставшихся цифр $6_{10} = 6_9$. Новое число Y= 42**0**$6_9 = 3084_{10}$.

2. Если получившееся после зачеркивания последних двух цифр число нечетное, то в конец полученной цепочки цифр дописывается одна цифра – количество оставшихся цифр. Например, $4859_{10} = 6588_9$. После зачеркивания остается 65_9 – нечетное число. Количество оставшихся цифр $2_{10}=2_9$. Новое число $Y = 652_7 = 533_{10}$.

Напишите программу, которая будет искать наибольшее число, которое получается на заданном интервале [A,B], и наибольшее число, из которого оно было получено.

На вход программы подаётся два целых числа A и B ($90 \le A \le B \le 1\,000\,000$), записанных через пробел.

Программа должна вывести два числа наибольшее полученное число и через пробел число, из

которого оно было получено исполнителем.

Входные данные	Вывод	Примечание
4859 4862	4872 4862	4859 -> 533 4860 -> 4872 4861 -> 4872 **4862 -> 4872** Наибольшее значение, получаемое на заданном интервале 4872. Оно получается из трех чисел 4860, 4861 и 4862. Наибольшее из них 4862.

Исходный код

```
#include <iostream>
#include <vector>
using namespace std;

int f(int x) {
 int otv = 0;
 vector<int> ans;
 while (x != 0)
 {
  ans.push_back(x % 9);
  x /= 9;
 }
 for (int i = ans.size() - 1; i >= 0; --i)
 {
  otv += ans[i];
  otv *= 10;
 }
 return otv;
}


int main()
{
 int a, b;
 cin >> a >> b;
 int mx = -1;
 int nmx = -1;
 for (int i = a; i <= b; ++i) {
  int t = i;
  int per = f(i) / 1000;
  if (per % 2 == 0)
  {
   int sum = 0;
   int h = per;
   while (h != 0)
   {
    sum += h % 10;
    h /= 10;
   }
   sum = f(sum) / 10;
   per *= 100;
   per += sum;
   long long ch = 1;
   int ans = 0;
   while (per != 0)
   {
    ans += (per % 10) * ch;
    ch *= 9;
    per /= 10;
   }
   if (ans >= mx)
   {
    mx = ans;
    nmx = i;
   }
  }
  else
  {
   int h = per;
   int cnt = 0;
   while (h != 0)
   {
    cnt++;
    h /= 10;
   }
```

```
   cnt = f(cnt) / 10;
   per *= 10;
   per += cnt;
   long long ch = 1;
   int ans = 0;
   while (per != 0)
   {
    ans += (per % 10) * ch;
    ch *= 9;
    per /= 10;
   }
   if (ans >= mx)
   {
    mx = ans;
    nmx = i;
   }
  }
 }
 cout << 81 * 9 * 9 + 4 * 81 * 9 + 81 * 6 + 9 + 2 << endl;
 cout << mx << " " << nmx;
 return 0;
}
```

Задача №4

Условие

Передатчик отправляет по одному слову в секунду. Слово состоит из строчных латинских букв. Сообщение состоит из нескольких слов. Слова в одном сообщении заканчиваются на одну букву. Напишите программу, которая найдет сообщение, содержащее больше всего слов.

Формат ввода

В первой строке вводится сначала целое число ***n*** – количество слов (***n ≤ 1000***), затем в ***n*** следующих строках записано по слову. Слова состоят только из строчных латинских букв.

Формат вывода

Вывести одно целое число – сколько слов было в искомом сообщении.

Примеры

Входные данные	Выходные данные
3 abc bac sdfe	2
4 me pudge purge tou	3

Исходный код

```
#include <iostream>
#include <vector>
#include<string>
#include <map>
#include<cmath>
using namespace std;
```

```
int main()
{
 int n;
 cin >> n;
 map<char, int> mp;
 int mx = -1;
 for (int i = 0; i < n; ++i)
 {
  string s;
  cin >> s;
  map<char, int> ::iterator it;
  it = mp.find(s[s.size() - 1]);
  if (it == mp.end())
   mp[s[s.size() - 1]] = 0;
  mp[s[s.size() - 1]]++;
  if (mx < mp[s[s.size() - 1]]) {
   mx = mp[s[s.size() - 1]];
  }
 }
 cout << mx << endl;
 return 0;
}
```

Задача №5

Условие

Информация передается по каналу связи в виде пакетов. Каждый пакет представляет собой целое положительное число в двоичной системе счисления. Для обнаружения помех последний разряд в пакете подбирают таким образом, чтобы количество единиц в разрядах пакета было четным. В каждом пакете на практике никогда не искажается больше одного разряда.

Напишите программу, которая по распечатке пакетов, записанных в десятичной системе счисления, найдет самое большое значение, прошедшее без искажений. Известно, что как минимум один пакет прошел без искажений.

Формат ввода

В строке вводится сначала целое число n – количество пакетов ($n \le 1000$), затем n натуральных чисел, все числа отделены друг от друга одним или несколькими пробелами.

Формат вывода

Вывести одно целое число – самое большое значение, прошедшее без искажений.

Пример

Входные данные	Выходные данные
4 1025 496 882 1056	1056

Исходный код

```
#include <iostream>
#include <vector>
#include <algorithm>
#include<string>
using namespace std;


int main()
{
 int n;
 cin >> n;
 int mx = -1;
 for (int i = 0; i < n; ++i)
 {
  int a;
  cin >> a;
  int t = a;
  vector<int> ans;
  while (t != 0)
  {
   ans.push_back(t % 2);
   t /= 2;
```

```
  }
  reverse(ans.begin(), ans.end());
  int sum = 0;
  if (sum % 2 == 0)
  {
   if (a > mx)
    mx = a;
  }
 }
 cout << mx;
 return 0;
}
```

Задача №6

Условие

В пошаговой браузерной игре игрок командует космическим фрегатом. Во время боя у него есть выбор из нескольких команд: «Поднять щиты», «Ракетный залп», «Орудийный залп» и «Маневр уклонения». Каждый ход можно выполнить только одну команду. Из-за игровой механики нельзя выполнить команду «Орудийный залп» в следующий ход после «Маневра уклонения». Нельзя выполнить команду «Ракетный залп» в следующий ход после подъема щитов. Нельзя два хода подряд поднимать щиты.

Напишите программу, которая определит, сколько существует выполнимых последовательностей команд длиной n.

На вход программе подается натуральное число n ($n \le 15$) – количество ходов.

Вывести целое число – количество выполнимых последовательностей команд длиной n.

Пример

Ввод	Вывод
2	13

Исходный код

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main()
{
 int n;
 cin >> n;
 int a = 1, b = 1, c = 1, d = 1;
 for (int i = 1; i < n; ++i)
 {
  int a1 = a;
  int b1 = b;
  int c1 = c;
  int d1 = d;
  a = b1 + c1 + d1;
  b = b1 + c1 + d1;
  c = a1 + b1 + c1;
  d = a1 + b1 + c1 + d1;
 }
 cout << a + b + c + d;
 return 0;
}
```

Задача №7

Условие

В электрической цепи требуется собрать заданное сопротивление. В распоряжении сборщика находится несколько типов резисторов, которые подключают последовательно. Для каждого из резисторов известно его сопротивление в омах.

Напишите программу, которая по заданным характеристикам резисторов и заданном сопротивлении найдет количество способов, с помощью которых можно собрать цепь с заданным сопротивлением.

На вход программе в первой строке подаются два натуральных числа R ($R \leq 100$) и k ($k \leq 8$) – заданное сопротивление (Ом) и количество типов резисторов соответственно. Далее в k строках идет натуральное число r_i – сопротивление типа резистора под номером i. Количество резисторов каждого типа считается неограниченным.

Вывести одно целое число – количество способов, которыми можно собрать заданное сопротивление. Если из резисторов нельзя собрать цепочку, вывести 0.

Пример

Ввод	Вывод
9 2 3 2	5

Исходный код

```
#include <iostream>
#include <vector>
#include <algorithm>
#include<string>
using namespace std;
vector<int> vec;
int r;
long long cnt = 0;
void dfs(int sum)
{
 if (sum > r)
  return;
 if (sum == r)
 {
  cnt++;
  return;
 }
 for (int i = 0; i < vec.size(); ++i)
 {
  dfs(sum + vec[i]);
 }
}


int main()
{
 int k;
 cin >> r >> k;
 for (int i = 0; i < k; ++i)
 {
  int a;
  cin >> a;
  vec.push_back(a);
 }
 dfs(0);
 cout << cnt;
 return 0;
}
```