Олимпиада СПбГУ по информатике 2019/20 учебного года

A	B	C	D	E	F	Sum
100	100	0	40	100	0	340

Task A ()

```
#include <iostream>

int main(void)
{
  int n;
  std::cin >> n;
  std::cout << n - 1 << std::endl;
#ifdef _TIm_MACRO
  system("pause");
#endif
  return 0;
}
```

Task B ()

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

using dbl_pair = std::pair<double, double>;
dbl_pair operator+(const dbl_pair& a, const dbl_pair& b)
{
  return dbl_pair(a.first + b.first, a.second + b.second);
}
dbl_pair operator-(const dbl_pair& a, const dbl_pair& b)
{
  return dbl_pair(a.first - b.first, a.second - b.second);
}
dbl_pair operator/(const dbl_pair& a, double b)
{
  return dbl_pair(a.first / b, a.second / b);
}
std::ostream& operator<<(std::ostream& Str, const dbl_pair& a)
{
  return Str << a.first << '␣' << a.second;
}

int main(void)
{
  int k;
  std::cin >> k;
  if (k == 6)
  {
    dbl_pair p[6];
    for (int i = 0; i < 6; i++)
    {
      std::cin >> p[i].first;
      std::cin >> p[i].second;
    }

    dbl_pair center = (p[0] + p[1] + p[2] + p[3] + p[4] + p[5]) / 6.0;
    std::sort(p, p + 6, [&center](const dbl_pair& a, const dbl_pair& b)->bool{return atan2(a.
        second - center.second, a.first - center.first) < atan2(b.second - center.second, b.first
        - center.first);});
    //std::cout << p[0] << std::endl << p[2] << std::endl << p[4] << std::endl;
    printf("%lf␣%lf\n%lf␣%lf\n%lf␣%lf\n", p[0].first, p[0].second, p[2].first, p[2].second, p[4].
        first, p[4].second);
  }
  else
  {
    dbl_pair p[6];
    for (int i = 0; i < 6; i += 2)
    {
      std::cin >> p[i].first;
      std::cin >> p[i].second;
    }

    dbl_pair center = (p[0] + p[2] + p[4]) / 3.0;
    auto mirror = [&center](const dbl_pair& a)->dbl_pair
    {
      return center - a + center;
    };
    // std::cout << p[0] << std::endl << mirror(p[2]) << std::endl << p[1] << std::endl << mirror(
        p[0]) << std::endl << p[2] << std::endl << mirror(p[1]) << std::endl;
    p[1] = mirror(p[4]);
    p[3] = mirror(p[0]);
    p[5] = mirror(p[2]);
    printf("%lf␣%lf\n%lf␣%lf\n%lf␣%lf\n%lf␣%lf\n%lf␣%lf\n%lf␣%lf\n", p[0].first, p[0].second, p
        [1].first, p[1].second, p[2].first, p[2].second, p[3].first, p[3].second, p[4].first, p
        [4].second, p[5].first, p[5].second);
  }
  return 0;
}
```

Task C ()

Task D ()

```cpp
#include <iostream>
#include <algorithm>
#include <string>
#include <queue>
#include <memory>
#include <utility>

using short_pair = std::pair<short, short>;
class cl : public std::pair<int, short_pair>
{
public:
  cl(int x, const short_pair& y) : std::pair<int, short_pair>(x, y){}
  cl(void){}
  bool operator<(const cl& a) const
  {
    if (a.first != first)
      return a.first < first;
    else
      return std::less<short_pair>()(a.second, second);
  }
};

int main(void)
{
  int n, m;
  short Ax, Ay, Bx, By;
  std::cin >> n >> m
           >> Ax >> Ay >> Bx >> By;
  Ax--;
  Ay--;
  Bx--;
  By--;
  std::unique_ptr<std::unique_ptr<cl[]>[]> Table;
  Table = std::make_unique<std::unique_ptr<cl[]>[]>(n);
  for (int i = 0; i < n; i++)
  {
    Table[i] = std::make_unique<cl[]>(m);
    for (int j = 0; j < m; j++)
    {
      std::cin >> Table[i][j].second.first;
      std::cin >> Table[i][j].second.second;
      Table[i][j].first = INT_MAX;
    }
  }
  std::priority_queue<cl> Q;
  Q.emplace(0, short_pair(Ax, Ay));

  while (!Q.empty())
  {
    cl First = Q.top();
    Q.pop();
    if (Table[First.second.first][First.second.second].first > First.first)
    {
      Table[First.second.first][First.second.second].first = First.first;
      // std::cout << First.second.first << ' ' << First.second.second << ' ' << First.first << std::endl;
      for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
        {
          int val = First.first + abs(i - First.second.first - Table[First.second.first][First.second.second].second.first) + abs(j - First.second.second - Table[First.second.first][First.second.second].second.second);
          if (val < Table[i][j].first)
            Q.push(cl(val, short_pair(i, j)));
        }
    }
  }
  std::cout << Table[Bx][By].first << std::endl;

  return 0;
}
```

Task E ()

```
#include <iostream>
#include <algorithm>
#include <string>
#include <queue>
#include <memory>
#include <utility>
#include <set>

using int_pair = std::pair<int, int>;
int_pair operator+(const int_pair& a, const int_pair& b)
{
  return int_pair(a.first + b.first, a.second + b.second);
}
int_pair operator-(const int_pair& a, const int_pair& b)
{
  return int_pair(a.first - b.first, a.second - b.second);
}
int_pair operator/(const int_pair& a, int b)
{
  return int_pair(a.first / b, a.second / b);
}
std::ostream& operator<<(std::ostream& Str, const int_pair& a)
{
  return Str << a.first << '␣' << a.second;
}

int main(void)
{
  long long h, w, s, t;
  int B;
  std::cin >> w >> h >> B;
  std::unique_ptr<int_pair[]> Black = std::make_unique<int_pair[]>(B);
  for (int i = 0; i < B; i++)
  {
    std::cin >> Black[i].first;
    Black[i].first--;
    std::cin >> Black[i].second;
    Black[i].second--;
  }

  int f_cnt = 1 << B;
  std::vector<int> clear(f_cnt, 0);

  int stage = 0;
  while (true)
  {
    int first = -1, second = -1;
    for (int i = 0; i < f_cnt; i++)
      if (clear[i] == stage)
      {
        first = i;
        break;
      }
    if (first == -1)
    {
      stage++;
      continue;
    }
    if (stage == B)
    {
      std::cout << "!␣" << first * w + 1 << "␣1" << std::endl;
      break;
    }
    for (int i = first + 1; i < f_cnt; i++)
      if (clear[i] == stage)
      {
        second = i;
        break;
      }
    if (second == -1)
    {
      stage++;
      continue;
    }
    if (stage == B)
    {
      std::cout << "!␣" << second * w + 1 << "␣1" << std::endl;
      break;
    }
    std::cout << "?␣" << Black[clear[first]].first + first * w + 1 << '␣' << Black[clear[first]].
        second + 1 << '␣'
                      << Black[clear[second]].first + second * w + 1 << '␣' << Black[clear[second
                          ]].second + 1 << std::endl;
    clear[first]++;
    clear[second]++;

    std::cin >> s >> t;
    s = (s - 1) / w;
    if (clear[s] == -1)
      clear[first] = -1;
    else
      clear[s] = -1;
  }

#ifdef _TIm_MACRO
  system("pause");
#endif
  return 0;
}
```

Task F ()