ЗАДАНИЕ 1

Рассчитать количество времени, затраченное человеком на перемещение по улице между домами. Схема улицы: на одной стороне находятся только дома с четными номерами, на противоположной — нечетные. Количество домов (N) устанавливает пользователь. Человек находится в начале улицы у первого дома. Пользователь задает два числа, которые являются номерами домов для посещения этим человек. Время движения между соседними домами составляет 1 мин. Человек посещает дома в той последовательности, которая указана. После посещения последнего дома ему требуется вернуться в первый дом. Пример. Улица состоит из 10 домов. Человеку требуется пройти от дома №3 до дома №7. Следовательно, ему для этого потребуется 1 мин, чтобы дойти до дома №3, и 2 мин, чтобы пройти от дома №3 до дома №7 и ещё 3 минуты до дома №1. Ответ: 6 мин.

Код программы:

coordinates.h

```
#ifndef COORDINATES_H
#define COORDINATES_H
#include <iostream>
#include <math.h>
/*!
 * \brief Класс, позволяющий работать с координатами
 *
 * Создаёт поле шириной 2 ячейки и произвольной длины
 * Примерная схема расположения ячеек -
 * 0 |2|4|6|8|10  |\n
 *   |------------|\n
 * 1 |1|3|5|7|9|11|\n
```

```
 *
 * Все чётные ячейки расположены сверху, все нечётные - снизу
 *
 *
*/
class coordinates
{
public:
    /*! \brief Конструктор, по которому задаётся текущее расположение в координатах
    */
    coordinates(uint64_t x, uint64_t y);
     //! Конструктор, по которому задаётся текущее расположение по номеру ячейки
    //! \throw std::out_of_range В случае передачи конструтору переменной, равной нулю
    coordinates(uint64_t number);
     //! Пустой конструктор, задающий значение (1; 0), в программе не используется
    coordinates();
    //! Получить координату x
    uint64_t x() const;
    //! Получить координату y
    uint64_t y() const;
    //! Получить номер текущей ячейки
    uint64_t number() const;
    //! Поменять координату y на противоположную. Т.е. с 0 на 1 и с 1 на 0,
```

```
        void changeY();
        //! Сдвинутся на 1 вправо по координате x
        void moveRight();
        //! Сдвинутся на 1 влево по координате x
        void moveLeft();
        //! \brief Оператор, сравнивающий 2 координаты. Возвращает истинное значение, если координаты не равны
        //! Координаты не равные, если у них отличается хотя бы 1 значение.
        bool const operator !=(const coordinates &otherCoords);
        //! \brief Оператор, сравнивающий 2 координаты. Возвращает истинное значение, если координаты равны
        bool const operator ==(const coordinates &otherCoords);
    private:
        //! Координата x
        uint64_t m_x;
        //! Координата y
        uint64_t m_y;
        //! Номер ячейки
        uint64_t m_number;
        //! \brief Обновить номер ячейки
        //! \details Для получения ячейки текущая координата x умножается на 2, и, если координата y нечётная, отнимается 1
        void updateNumber();
    };
    #endif // COORDINATES_H
```

coordinates.cpp

```
    #include "coordinates.h"
```

```
coordinates::coordinates(uint64_t x, uint64_t y)
{
   m_x = x;
   m_y = y;
   updateNumber();
}
coordinates::coordinates(uint64_t number)
{
   if(number == 0)
        throw std::out_of_range ("The number passed to the class constructor cannot be 0!");
   m_x = ceil(double(number) / 2);
   m_y = number % 2;
   m_number = number;
}
coordinates::coordinates()
{
   m_x = 1;
   m_y = 0;
   m_number = 1;
}
uint64_t coordinates::x() const
{
   return m_x;
}
uint64_t coordinates::y() const
{
```

```
    return m_y;
}
void coordinates::changeY()
{
    if (m_y == 0)
        m_y = 1;
    else
        m_y = 0;
    updateNumber();
}
void coordinates::moveRight()
{
    m_x += 1;
    updateNumber();
}
void coordinates::moveLeft()
{
    m_x -= 1;
    updateNumber();
}
const bool coordinates::operator!=(const coordinates &otherCoords)
{
    return !(m_x == otherCoords.x() and m_y == otherCoords.y());
}
const bool coordinates::operator ==(const coordinates &otherCoords)
{
    return m_x == otherCoords.x() and m_y == otherCoords.y();
```

```
}
uint64_t coordinates::number() const
{
   return m_number;
}
void coordinates::updateNumber()
{
   m_number = (m_x * 2) - m_y;
}
```

mover.h

```
#ifndef MOVER_H
#define MOVER_H
#include "coordinates.h"
/*!
 * \brief Класс, создающий поле и перемещающегося по нему юнита
 *
 * Юнит может двигаться в 8 направлениях - вправо, влево, вниз, вверх и по-диагонали
 * Если поле, на которое юнит пытается подвинуться, не существует, функция возвращает false
 *
 */
class mover
{
public:
   //! Конструктор класса
   //! \param coordinatesOfMover Задаёт изначальную позицию юнита
```

```
        //! \param sizeOfStreet Задаёт размер поля, а конкретнее - номер крайней ячейки
        //!
        //! \throw std::out_of_range В случае передачи конструтору переменной, равной нулю
        mover(uint64_t coordinatesOfMover, uint64_t sizeOfStreet);
        //! Пустой деструктор
        ~mover();
        //! Получить текущую позицию
        const coordinates &currentPosition() const;
        //! Сдвинуться вверх
        bool moveUp();
        //! Сдвинуться вниз
        bool moveDown();
        //! Сдвинуться влево
        bool moveLeft();
        //! Сдвинуться
        bool moveRight();
        //! Сдвинуться вверх и вправо
        bool moveUpAndRight();
        //! Сдвинуться вверх и влево
        bool moveUpAndLeft();
        //! Сдвинуться вниз и вправо
        bool moveDownAndRight();
        //! Сдвинуться вниз и влево
        bool moveDownAndLeft();
        //! Перейти на другую сторону и сдвинуться на 1 вправо
```

```
        bool switchSideAndGoRight();
        //! Перейти на другую сторону и сдвинуться на 1 влево
        bool switchSideAndGoLeft();
        //! Перейти на другую сторону
        bool switchSide();
        //! \brief Дойти до ячейки с номером point
            //! \return Возвращает число шагов от начального положения до конечной точки
        uint64_t moveTo(uint64_t point);
        //! \brief Дойти до ячейки с номером point
            //! \return Возвращает число шагов от начального положения до конечной точки
        uint64_t moveTo(const coordinates &point);
        //! \brief Дойти до ячейки с номером point
            //! \return Возвращает число шагов от начального положения до конечной точки
        uint64_t goHome();
    private:
        //! \brief Проверить возможность движения по координате
        //!
         //! Возвращает true, если позиция moveTo находится на расстоянии не больше 1 от текущей позиции, и если такая точка на поле существует
        bool isMoveAvailable(coordinates moveTo);
        //! Размер поля
        coordinates m_sizeOfStreet;
        //! Текущая позиция юнита
        coordinates m_currentPosition;
```

```
    //! Изначальная позиция
    coordinates m_startingPosition;
};
#endif // MOVER_H
```

mover.cpp

```
#include "mover.h"
mover::mover(uint64_t coordinateOfMover, uint64_t sizeOfStreet) :
    m_sizeOfStreet(sizeOfStreet),
    m_currentPosition(coordinateOfMover),
    m_startingPosition(coordinateOfMover)
{
}
mover::~mover()
{
}
bool mover::isMoveAvailable(coordinates moveTo)
{
    return ((moveTo.x() >= m_currentPosition.x() - 1) and (moveTo.x() <=
m_currentPosition.x() + 1)
            and (m_currentPosition != moveTo)) and (moveTo.number() <=
m_sizeOfStreet.number());
}
const coordinates &mover::currentPosition() const
{
    return m_currentPosition;
}
```

```
bool mover::moveUp()
{
   // Если юнит уже наверху поля, функция возвращает false
   if (m_currentPosition.y() == 0)
      return false;
                              if(isMoveAvailable(coordinates(m_currentPosition.x(),
m_currentPosition.y() - 1)))
   {
      m_currentPosition.changeY();
      return true;
   }
   return false;
}
bool mover::moveDown()
{
   // Если юнит уже внизу поля, функция возвращает false
   if (m_currentPosition.y() == 1)
      return false;
                              if(isMoveAvailable(coordinates(m_currentPosition.x(),
m_currentPosition.y() + 1)))
   {
      m_currentPosition.changeY();
      return true;
   }
   return false;
}
bool mover::moveLeft()
```

```
    {
        if (m_currentPosition.x() == 1)
            return false;
                    if(isMoveAvailable(coordinates(m_currentPosition.x() - 1,
m_currentPosition.y())))
        {
            m_currentPosition.moveLeft();
            return true;
        }
        return false;
    }
    bool mover::moveRight()
    {
                    if(isMoveAvailable(coordinates(m_currentPosition.x() + 1,
m_currentPosition.y())))
        {
            m_currentPosition.moveRight();
            return true;
        }
        return false;
    }
    bool mover::moveUpAndRight()
    {
        // Если юнит уже наверху поля, функция возвращает false
        if (m_currentPosition.y() == 0)
            return false;
```

```
                    if(isMoveAvailable(coordinates(m_currentPosition.x() + 1,
m_currentPosition.y() - 1)))
        {
          m_currentPosition.changeY();
          m_currentPosition.moveRight();
          return true;
        }
        return false;
      }
      bool mover::moveUpAndLeft()
      {
        // Если юнит уже наверху поля, функция возвращает false
        if (m_currentPosition.y() == 0)
          return false;
        if (m_currentPosition.x() == 1)
          return false;
                    if(isMoveAvailable(coordinates(m_currentPosition.x() - 1,
m_currentPosition.y() - 1)))
        {
          m_currentPosition.changeY();
          m_currentPosition.moveLeft();
          return true;
        }
        return false;
      }
      bool mover::moveDownAndRight()
      {
```

```
        // Если юнит уже внизу поля, функция возвращает false
        if (m_currentPosition.y() == 1)
            return false;
                if(isMoveAvailable(coordinates(m_currentPosition.x() + 1,
m_currentPosition.y() + 1)))
        {
          m_currentPosition.changeY();
          m_currentPosition.moveRight();
          return true;
        }
        return false;}
      bool mover::moveDownAndLeft()
      {
        // Если юнит уже внизу поля, функция возвращает false
        if (m_currentPosition.y() == 1)
          return false;
        if (m_currentPosition.x() == 1)
          return false;
      if(isMoveAvailable(coordinates(m_currentPosition.x() - 1,
m_currentPosition.y() + 1)))
        {
          m_currentPosition.changeY();
          m_currentPosition.moveLeft();
          return true;
        }
        return false;
      }
```

```
bool mover::switchSideAndGoRight()
{
   // try to move down and right or down and left
   // If it didn't work, then return false
   if(moveUpAndRight() == 0)
      if (moveDownAndRight() == 0)
         return false;
   return true;
}
bool mover::switchSideAndGoLeft()
{
   // try to move up and right or up and left
   // If it didn't work, then return false
   if(moveDownAndLeft() == 0)
      if(moveDownAndLeft() == 0)
         return false;
   return true;
}
bool mover::switchSide()
{
   // Try to move up or down
   if(moveDown() == 0)
      if(moveUp() == 0)
         return false;
   return true;
}
uint64_t mover::moveTo(uint64_t point)
```

```
{
    // Перегружает метод
    return moveTo(coordinates(point));
}
uint64_t mover::moveTo(const coordinates &point)
{
    uint64_t steps = 0;
    // Первым делом нужно перейти на нужную линию поля
    if(m_currentPosition.y() != point.y() and m_currentPosition.x() < point.x())
    {
        if (switchSideAndGoRight() == true)
            steps++;
    }
        else if(m_currentPosition.y() != point.y() and m_currentPosition.x() > point.x())
    {
        if (switchSideAndGoLeft() == true)
            steps++;
    }
       else if(m_currentPosition.y() != point.y() and m_currentPosition.x() == point.x())
    {
        if (switchSide() == true)
            steps++;
    }
    // Теперь добраться до нужной точки до прямой, пока они не совпадут
    if(m_currentPosition.x() < point.x())
```

```
        {
            while(m_currentPosition != point)
            {
                moveRight();
                steps++;
            }
        }
        else if(m_currentPosition.x() > point.x())
        {
            while(m_currentPosition.x() != point.x())
            {
                moveLeft();
                steps++;
            }
        }
        return steps;
    }
    uint64_t mover::goHome()
    {
        return moveTo(m_startingPosition);
    }
```

main.cpp

```
    #include <iostream>
    #include "mover.h"
    using namespace std;
    int main()
    {
```

```
uint64_t sizeOfStreet, firstPoint, secondPoint;
int64_t input[3];
for (int i = 0; i < 3; i++)
{
   cin >> input[i];
   if (cin.fail())
   {
      cerr << "Error. Input is not integer number. Please, try again" << endl;
      return 1;
   }
   if (input[i] <= 0)
   {
      cerr << "Error. Numbers cannot be less or equal to 0!" << endl;
      return 1;
   }
}
sizeOfStreet = input[0];
firstPoint = input[1];
secondPoint = input[2];
if (firstPoint > sizeOfStreet or secondPoint > sizeOfStreet)
{
   cerr << "Error. Points is not available in that field" << endl;
   return 1;
}
mover *mover;
try
{
```

```
        mover = new class mover(1, sizeOfStreet);
    }
    catch (std::out_of_range error)
    {
      cerr << error.what() << endl;
      return 1;
    }
    uint64_t steps = 0;
    steps += mover->moveTo(firstPoint);
    steps += mover->moveTo(secondPoint);
    steps += mover->goHome();
    cout << steps << endl;
    return 0;
}
```

Результаты тестирования

Таблица 1 — Таблица с результатами тестирования программы из 1 задания

Тестовый набор	**Ожидаемый результат**	**Действительный результат**
10 5 6	5	5
21 3 16	14	14
82 85 3	Error. Points is not available in that field	Error. Points is not available in that field
362 42 -12	Error. Numbers cannot be less or equal to 0!	Error. Numbers cannot be less or equal to 0!
132 6.7 43	Error. Input is not integer number. Please, try again	Error. Input is not integer number. Please, try again

ЗАДАНИЕ 2

Текст задания

Тоже, что и в задании №1. Только пользователь задает неограниченное количество домов, которые должен посетить человек. Принцип ходьбы заключается в том, что человек каждый раз идет в самый дальний дом. Как только он закончит обход, то перемещается в конец улицы (вправо). Пример. На улице 20 домов. Пользователь ввел числа 4, 1, 8, 15, 20, 4, 5. Маршрут человека будет следующим: 20 – 1 – 15 – 4 – 8 – 4 – 5 и уйти в конец к 20 дому.

Язык программирования

C++

Код программы

coordinates.h

Аналогично заданию 1

coordinates.cpp

Аналогично заданию 1

mover.h

Аналогично заданию 1

mover.cpp

Аналогично заданию 1

main.cpp

```
#include <iostream>
#include <vector>
#include <algorithm>
#include "mover.h"
```

```
using namespace std;
int main()
{
    uint64_t sizeOfStreet = 0;
    vector<int64_t> input;
    while (!cin.eof())
    {
        uint64_t number;
        cin >> number;
        if(cin.eof())
            break;
        while (cin.fail()){
            cerr << "Error. Input is not integer number. Please, try again" << endl;
            return 1;
        }
        if (number <= 0)
        {
            cerr << "Error. Numbers cannot be less or equal to 0!" << endl;
            return 1;
        }
        if(sizeOfStreet > 0)
            input.push_back(number);
        else
            sizeOfStreet = number;
    }
    for(int i = 0; i < input.size(); i++)
    {
```

```
    if (input[i] > sizeOfStreet)
    {
        cerr << "Error. Points is not available in that field" << endl;
        return 1;
    }
}
mover *mover;
try
{
    mover = new class mover(1, sizeOfStreet);
}
catch (std::out_of_range error)
{
   cerr << error.what() << endl;
   return 1;
}
uint64_t steps = 0;
sort(input.begin(), input.end());
uint64_t var = ceil(double(input.size()) / 2) ;
for(int i = 0; i < var; i++)
{
    steps += mover->moveTo(input.at(input.size() - 1 - i));
    steps += mover->moveTo(input.at(i));
}
steps += mover->moveTo(mover->sizeOfStreet());
cout << steps << endl;
return 0;
```

}

Результаты тестирования

Таблица 2 — Таблица с результатами тестирования программы из 2 задания

Тестовый набор	Ожидаемый результат	Действительный результат
10 5 6 2 3 4 5	12	12
20 10 3 5 1	19	19
132	65	65
36 2	18	18
6 23 12	Points is not available in that field	Error. Points is not available in that field

ЗАДАНИЕ 3

Текст задания

Создать массив из случайного количества элементов в диапазоне от 20 до 50 со значениями, которые генерируются случайным образом в диапазоне от –100 до 30. Если количество отрицательных элементов больше половины, то случайным образом переопределяются знаки у каждого числа. Переопределение происходит до тех пор, пока не будет достигнуто требуемое соотношение отрицательных и положительных элементов. Посчитать количество четных положительных элементов, расположенных на нечетных местах; посчитать количество всех цифр отрицательных элементов; посчитать сумму всех нечетных чисел, расположенных на четных местах. Результаты всех подсчетов сложить. Если число четное, то полученный результат прибавить к значению отрицательных элементов; нечетный - отнять. Записать исходный массив и конечный в файл txt в строку для каждого. Расчетные значения через ; записать в третью строку.

Язык программирования

C++

Код программы

Main.cpp

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <fstream>
using namespace std;
//! Инициализировать массив случайными числами в диапаоне от -100 до 30
```

```
//! \param arr Адрес начала массива
//! \param size Размер массива
void initArray(int64_t * arr, uint64_t size)
{
   for (int i = 0; i < size; i++)
   {
      arr[i] = rand() % 130 - 99;
   }
}
//! Копировать значения одного массива в другой
//! \warning Предполагается то, что на новый массив уже выделена память, не меньше чем на оригинальный
void copyArray(int64_t * originalArr, int64_t * newArray, uint64_t size)
{
   for(int i = 0; i < size; i++)
   {
      newArray[i] = originalArr[i];
   }
}
//! Проверить массив на то, что больше половины элементов - не отрицательные
//! \param arr Адрес начала массива
//! \param size Размер массива
bool checkArray(int64_t * arr, uint64_t size)
{
   uint64_t amountNegative = 0;
   uint64_t amountPositive = 0;
```

```
    for(int i = 0; i < size; i++)
    {
        if(arr[i] < 0)
            amountNegative++;
        else
            amountPositive++;
    }
    return (amountPositive > amountNegative);
}
//! Случайно поменять знаки у чисел в массиве
//! \param arr Адрес начала массива
//! \param size Размер массива
void changeSignes(int64_t * arr, uint64_t size)
{
    for(int i = 0; i < size; i++)
    {
        if (rand() % 2)
            arr[i] = -arr[i];
    }
}
//! Проверить чётность числа
bool isEven(int64_t num)
{
    return !(num%2);
}
//! Проверить нечётность числа
bool isOdd(int64_t num)
```

```
{
    return num%2;
}
//! Посчитать количество четных положительных чисел на нечётных позициях
uint64_t countEvenPositiveOnOddPositions(int64_t * arr, uint64_t size)
{
    uint64_t counter = 0;
    for(int i = 1; i < size; i += 2)
    {
        if(arr[i] > 0 && isEven(arr[i]))
            counter++;
    }
    return counter;
}
//! Посчитать, сколько цифр содержит число
uint64_t howManyDigitsContain(int64_t num)
{
    num = abs(num);
    uint64_t count = 0;
    while(num > 0)
    {
        num /= 10;
        count++;
    }
    return count;
}
```

```
//! Посчитать сколько цифр содержат в себе все отрицательные числа массива
uint64_t countDigitsInNegativeNumbers(int64_t * arr, uint64_t size)
{
   uint64_t count = 0;
   for(int i = 0; i < size; i++)
   {
      if(arr[i] < 0)
         count += howManyDigitsContain(arr[i]);
   }
   return count;
}
//! Посчитать сумму нечётных чисел на чётных позициях
int64_t countSumOfOddNumbersOnEvenPosition(int64_t * arr, uint64_t size)
{
   int64_t sum = 0;
   for(int i = 0 ; i < size; i += 1)
   {
      if(isOdd(arr[i]))
         sum += arr[i];
   }
   return sum;
}
//! Найти количество отрицательных элементов в массиве
uint64_t getAmountOfNegativeNumbers(int64_t * arr, uint64_t size)
{
   uint64_t count;
```

```
    for(int i = 0; i < size; i++)
    {
        if(arr[i] < 0)
            count++;
    }
    return count;
}
//! Записать в файл значение массива
void writeValueOfArray(ofstream &fout, int64_t * arr, uint64_t size)
{
    for(int i = 0; i < size; i++)
        fout << arr[i] << " ";
    fout << endl;
}
int main()
{
    srand(time(0));
    uint64_t size = rand() % 30 + 20;
    int64_t arr[size];
    int64_t originalArr[size];
    initArray(originalArr, size);
    copyArray(originalArr, arr, size);
    while(checkArray(arr, size) == false)
        changeSignes(arr, size);
                                uint64_t evenPositiveOnOddPositions =
countEvenPositiveOnOddPositions(arr, size);
```

```
            uint64_t digitsInNegativeNumbers = countDigitsInNegativeNumbers(arr, size);
                                int64_t sumOfOddNumbersOnEvenPositions = countSumOfOddNumbersOnEvenPosition(arr, size);
                        int64_t allParameters = evenPositiveOnOddPositions + digitsInNegativeNumbers + sumOfOddNumbersOnEvenPositions;
            uint64_t amountOfNegativeNumbers = getAmountOfNegativeNumbers(arr, size);
            if(isEven(allParameters))
               allParameters += amountOfNegativeNumbers;
            else
               allParameters -= amountOfNegativeNumbers;
            ofstream fout;
            fout.open("3.txt");
            if(!fout.is_open())
            {
               cerr << "Cannot open file to write" << endl;
               return 1;
            }
            writeValueOfArray(fout, originalArr, size);
            writeValueOfArray(fout, arr, size);
             fout << evenPositiveOnOddPositions << ";" << digitsInNegativeNumbers << ";" << sumOfOddNumbersOnEvenPositions << endl;
            fout.close();
            return 0;
         }
```

Результаты тестирования

Таблица 3 — Таблица с результатами тестирования программы из 3 задания

Тестовый набор	**Ожидаемый результат**	**Действительный результат**
Запуск в домашнем каталоге пользователя	19 -23 -24 -1 24 -58 -13 18 11 28 -66 -47 -32 -7 -27 -83 -53 -85 -21 20 13 -34 12 -6 -33 28 -71 -66 -93 7 -8 25 -45 -61 -5 -52 -20 -19 -23 24 -1 24 58 -13 18 -11 28 -66 47 -32 7 -27 83 -53 85 21 -20 13 34 12 -6 -33 28 71 -66 93 7 8 25 45 -61 -5 52 20 6;27;251	19 -23 -24 -1 24 -58 -13 18 11 28 -66 -47 -32 -7 -27 -83 -53 -85 -21 20 13 -34 12 -6 -33 28 -71 -66 -93 7 -8 25 -45 -61 -5 -52 -20 -19 -23 24 -1 24 58 -13 18 -11 28 -66 47 -32 7 -27 83 -53 85 21 -20 13 34 12 -6 -33 28 71 -66 93 7 8 25 45 -61 -5 52 20 6;27;251
Запуск в домашнем каталоге пользователя	25 -28 -83 -50 24 -7 -50 -22 17 26 21 -30 -16 -72 -41 -23 -91 7 -5 19 -65 -33 -11 -20 -50 25 -28 83 -50 24 -7 -50 -22 17 -26 -21 30 16 72 41 23 91 -7 -5 19 -65 -33 -11 20 50 3;21;150	-25 -28 -83 -50 24 -7 -50 -22 17 26 21 -30 -16 -72 -41 -23 -91 7 -5 19 -65 -33 -11 -20 -50 25 -28 83 -50 24 -7 -50 -22 17 -26 -21 30 16 72 41 23 91 -7 -5 19 -65 -33 -11 20 50 3;21;150
Запуск в домашнем каталоге пользователя	-25 -28 -83 -50 24 -7 -50 -22 17 26 21 -30 -16 -72 -41 -23 -91 7 -5 19 -65 -33 -11 -20 -50 25 -28 83 -50 24 -7 -50 -22 17 -26 -21 30 16 72 41 23 91 -7 -5 19 -65 -33 -11 20 50 3;21;150	-88 -21 -14 -40 26 -79 -50 -12 -21 -11 -97 -3 -63 -15 10 26 -2 -44 -95 -51 -14 -13 0 -88 21 14 -40 26 79 50 12 -21 11 -97 3 63 -15 -10 26 -2 44 -95 -51 -14 -13 0 3;21;-115
Запуск в домашнем каталоге пользователя	10 -29 -80 -44 30 -40 -73 9 -53 -17 -79 -47 -40 -77 26 -73 -48 -94 -22 -39 -10 -29 80 44 30 40 73 -9 -53 -17 79 47 40 77 26 73 -48 94 -22 -39 3;15;202	10 -29 -80 -44 30 -40 -73 9 -53 -17 -79 -47 -40 -77 26 -73 -48 -94 -22 -39 -10 -29 80 44 30 40 73 -9 -53 -17 79 47 40 77 26 73 -48 94 -22 -39 3;15;202

Запуск происходит в директории, где у пользователя нет прав на создание файла	Cannot open file to write	Cannot open file to write

ЗАДАНИЕ 4

Текст задания

Пользователь вводит строку из произвольного числа символов. В середину каждого слова (если четное число символов) добавить соседний справа символ из таблицы кодировки для первого символа слова; если нечетное количество символов в слове, то требуется удалить центральный символ. Словом считается конструкция, отделенная знаками препинания, пробелами или служебными символами.

Язык программирования

C++/Qt

Код программы

main.cpp

```
#include <iostream>
#include <string>
#include <QString>
#include <QRegularExpression>
int main()
{
   std::string tmpString;
   std::getline(std::cin, tmpString);
   QString string(tmpString.c_str());
        auto words = string.split(QRegularExpression("\\b",
QRegularExpression::UseUnicodePropertiesOption));
   for(auto word = words.begin() + 1; word != words.end(); word++)
   {
      if(word->size() % 2 == 1 and word->simplified().length() > 1)
```

```
            word->remove(word->size()/2, 1);
        else if (word->size() % 2 == 0 and word->simplified().length() > 1)
            word->insert(word->size()/2, QChar(int(word->at(0).unicode()) + 1));
        std::cout << word->toStdString();
    }
    std::cout << std::endl;
    return 0;
}
```

Результаты тестирования

Таблица 4 — Таблица с результатами тестирования программы из 4 задания

Тестовый набор	**Ожидаемый результат**	**Действительный результат**
Словом считается конструкция, отделенная знаками препинания, пробелами или служебными символами.	СлоТвом считется констукция, отделпенная знаами препирнания, проблами ии служетбными симвлами.	СлоТвом считется констукция, отделпенная знаами препирнания, проблами ии служетбными симвлами.
345 3456 Hello ?132@qQ24 qwr!qw4?wer adsadaA	35 34456 Helo ?12@qQr24 qr! q4?wr adsdaA	35 34456 Helo ?12@qQr24 qr! q4?wr adsdaA
The quick brown fox jumps over the lazy dog	Te quck brwn fx jups ovper te lamzy dg	Te quck brwn fx jups ovper te lamzy dg
Съешь же этих мягких французских булок, да выпей чаю	Съшь жзе этюих мягнких францзских буок, деа выей чю	Съшь жзе этюих мягнких францзских буок, деа выей чю
F241 132 ASW - 1233456789	F2G41 12 AW - 12334256789	F2G41 12 AW - 12334256789

ЗАДАНИЕ 5

Текст задания

Посчитать количество слов в тексте из п. 4 и количество знаков препинания.

Язык программирования

C++/Qt

Код программы

main.cpp

```
#include <iostream>
#include <string>
#include <QString>
#include <QRegularExpression>
int main()
{
   std::string tmpString;
   std::getline(std::cin, tmpString);
   QString string(tmpString.c_str());
                auto words = string.split(QRegularExpression("\\b",
QRegularExpression::UseUnicodePropertiesOption));
   uint64_t amountOfWords = 0;
   uint64_t amountOfMarks = 0;
   for(auto word = words.begin() + 1; word != words.end(); word++)
   {
      if(word->length() % 2 == 1 and word->simplified().length() > 1)
      {
         word->remove(word->length()/2, 1);
```

```
            amountOfWords++;
        }
        else if (word->length() % 2 == 0 and word->simplified().length() > 1)
        {
            word->insert(word->length()/2, QChar(int(word->at(0).unicode()) +
1));
            amountOfWords++;
        }
        else if(word->simplified().length() == 1 && word->simplified()
[0].isPunct())
        {
            amountOfMarks++;
        }
    }
    std::cout << amountOfWords << " " << amountOfMarks << "\n";
    return 0;
}
```

Результаты тестирования

Таблица 5 — Таблица с результатами тестирования программы из 5 задания

Тестовый набор	Ожидаемый результат	Действительный результат
Словом считается конструкция, отделенная знаками препинания, пробелами или служебными символами.	10 3	10 3
345 3456 Hello ?132@qQ24 qwr!qw4? wer adsadaA	9 4	9 4

The quick brown fox jumps over the lazy dog	9 0	9 0
Съешь же этих мягких французских булок, да выпей чаю	9 1	9 1
F241 132 ASW - 1233456789	4 2	4 2

ЗАДАНИЕ 6

Текст задания

Пользователь с клавиатуры вводить любое целое положительное число, которое соответствует количеству символов строки. Символы строки генерируются случайным образом и должны соответствовать следующим правилам: в строке обязательно должна быть как минимум одна цифра, в строке обязательно должна быть как минимум одна прописная буква, в строке обязательно должен быть как минимум один символ из следующего набора: ! # % : , . ; * ( ) [ ] { } < > / ? @ & - + = в строке должна быть как минимум одна строчная буква, все буквы латинские. Расположение каждого символа случайно.

Язык программирования

C++/Qt

Код программы

main.cpp

```
#include<iostream>
#include<cstdlib>
#include<string>
#include<time.h>
using namespace std;
void generateAmount(uint64_t * amountOf, uint64_t size, int64_t length)
{
   int64_t amountSize = 0;
   for(int i = 0; i < size; i++)
   {
      if(amountSize >= length)
```

```
            {
                amountSize = 0;
                break;
            }
            amountOf[i] = rand() % (length - amountSize) + 1;
            amountSize += amountOf[i];
        }
        if(amountSize != length)
            generateAmount(amountOf, size, length);
    }
    //! Копировать значения одного массива в другой
    //! \warning Предполагается то, что на новый массив уже выделена память, не меньше чем на оригинальный
    void copyArray(uint64_t * originalArr, uint64_t * newArray, uint64_t size)
    {
        for(int i = 0; i < size; i++)
        {
            newArray[i] = originalArr[i];
        }
    }
    char generateSpecialSymbol(uint64_t number)
    {
        string symbols = {
          '!', '#', '%', ':', ',', '.', ';', '*', '(', ')', '[', ']', '{', '}', '<', '>', '/', '?', '@', '&', '-', '+', '='
        };
        return symbols.at(number % symbols.size());
    }
```

```
// 0 - Lowercase;
// 1 - Uppercase;
// 2 - Nums;
// 3 - Special symbols
char generateSymbol(uint64_t type)
{
   if(type == 0)
      // From a to z
      return (rand() % 26) + 97;
   else if(type == 1)
      // From A to Z
      return (rand() % 26) + 65;
   else if(type == 2)
      // from 0 to 9
      return (rand() % 10) + 48;
   else
      return generateSpecialSymbol(rand());
}
int main()
{
   srand(time(0));
   int64_t length;
   cin >> length;
   if (cin.fail())
   {
      cerr << "Error. Input is not integer number. Please, try again" << endl;
      return 1;
```

```
}
if (length <= 0)
{
   cerr << "Error. Numbers cannot be less or equal to 0!" << endl;
   return 1;
}
string textString;
textString.reserve(length);
// 0 - Lowercase;
// 1 - Uppercase;
// 2 - Nums;
// 3 - Special symbols
uint64_t amountOf[4];
uint64_t amountOfReduced[4];
generateAmount(amountOf, 4, length);
copyArray(amountOf, amountOfReduced, 4);
for(int i = 0; i < length; i++)
{
   uint64_t type = rand() % 4;
   if(amountOfReduced[type] != 0)
   {
      textString += generateSymbol(type);
      amountOfReduced[type]--;
   }
   else
   {
      i--;
```

```
            }
        }
        cout << textString << endl;
        return 0;
    }
```

Результаты тестирования

Таблица 6 — Таблица с результатами тестирования программы из 6 задания

Тестовый набор	Ожидаемый результат	Действительный результат
10	Строка, соответствующая критериям	,Y1U=swkmz
12	Строка, соответствующая критериям	1j,Y1UFCwXGG
20	Строка, соответствующая критериям	SDzo%gFC068wk1JGGzeP
Hello	Error. Input is not integer number. Please, try again	Error. Input is not integer number. Please, try again
-6	Error. Numbers cannot be less or equal to 0!	Error. Numbers cannot be less or equal to 0!

ЗАДАНИЕ 7

Текст задания

Строку, сформированную в п. 6, преобразовать в jpg-картинку размером 720*360. Строка должна не выходить за границы. Цвет букв красный, на фоне зелёный шум, из левого нижнего угла в правый верхний угол размещена жёлтая линия толщиной 0,75 пт.

Язык программирования

C++/Qt

Код программы

main.cpp

```
#include<iostream>
#include<cstdlib>
#include<string>
#include<time.h>
#include<QImage>
#include<QPainter>
#include<QRectF>
#include<QString>
#include<QGuiApplication>
#include<QTimer>
using namespace std;
void generateAmount(uint64_t * amountOf, uint64_t size, int64_t length)
{
   int64_t amountSize = 0;
   for(int i = 0; i < size; i++)
   {
```

```
            if(amountSize >= length)
            {
               amountSize = 0;
               break;
            }
            amountOf[i] = rand() % (length - amountSize) + 1;
            amountSize += amountOf[i];
         }
         if(amountSize != length)
            generateAmount(amountOf, size, length);
      }
      //! Копировать значения одного массива в другой
      //! \warning Предполагается то, что на новый массив уже выделена память, не меньше чем на оригинальный
      void copyArray(uint64_t * originalArr, uint64_t * newArray, uint64_t size)
      {
         for(int i = 0; i < size; i++)
         {
            newArray[i] = originalArr[i];
         }
      }
      char generateSpecialSymbol(uint64_t number)
      {
         string symbols = {
          '!', '#', '%', ':', ',', '.', ';', '*', '(', ')', '[', ']', '{', '}', '<', '>', '/', '?', '@', '&', '-', '+', '='
         };
         return symbols.at(number % symbols.size());
```

```
        }
        void prepareString(QString &textString)
        {
          int charsInLine;
              for(charsInLine = 20; textString.size()/charsInLine > charsInLine/5;
charsInLine += 10);
          for(int i = charsInLine - 1; i < textString.size(); i += charsInLine)
          {
             textString.insert(i, "\n");
             i++;
          }
          return;
        }
        // 0 - Lowercase;
        // 1 - Uppercase;
        // 2 - Nums;
        // 3 - Special symbols
        char generateSymbol(uint64_t type)
        {
           if(type == 0)
              // From a to z
              return (rand() % 26) + 97;
           else if(type == 1)
              // From A to Z
              return (rand() % 26) + 65;
           else if(type == 2)
              // from 0 to 9
```

```
        return (rand() % 10) + 48;
    else
        return generateSpecialSymbol(rand());
}
int main(int argc, char * argv[])
{
    srand(time(0));
    int64_t length;
    cin >> length;
    if (cin.fail())
    {
        cerr << "Error. Input is not integer number. Please, try again" << endl;
        return 1;
    }
    if (length <= 0)
    {
        cerr << "Error. Numbers cannot be less or equal to 0!" << endl;
        return 1;
    }
    QString textString;
    textString.reserve(length);
    // 0 - Lowercase;
    // 1 - Uppercase;
    // 2 - Nums;
    // 3 - Special symbols
    uint64_t amountOf[4];
    uint64_t amountOfReduced[4];
```

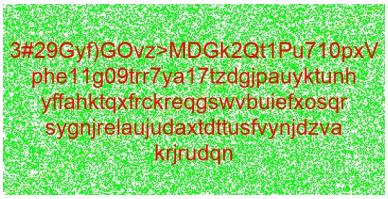
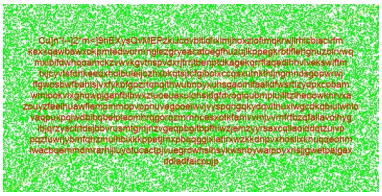
```
generateAmount(amountOf, 4, length);
copyArray(amountOf, amountOfReduced, 4);
for(int i = 0; i < length; i++)
{
   uint64_t type = rand() % 4;
   if(amountOfReduced[type] != 0)
   {
      textString += generateSymbol(type);
      amountOfReduced[type]--;
   }
   else
   {
      i--;
   }
}
QGuiApplication a(argc, argv);
QImage textImage(QSize(720, 360), QImage::Format_RGB32);
QPainter painter(&textImage);
for(int i = 0; i < textImage.size().width(); i += 2)
{
   for(int j = 0; j < textImage.height(); j += 2)
   {
      if(rand() % 2 == 1)
      {
         painter.setBrush(Qt::green);
      }
      else
```

```
            {
                painter.setBrush(Qt::white);
            }
            painter.fillRect(QRect(i, j, 5, 5), painter.brush());
        }
    }
    painter.setPen(QPen(Qt::yellow, 0.75));
    painter.drawLine(0, 360, 720, 0);
    prepareString(textString);
    QFont font("arial");
    font.setPixelSize(10);
    painter.setFont(font);
    auto fontMetric = painter.fontMetrics();
     while(painter.fontMetrics().size(Qt::TextWordWrap, textString).width() <=
700)
    {
        int fontSize = painter.font().pixelSize();
        auto newFont = painter.font();
        newFont.setPixelSize(fontSize + 1);
        painter.setFont(newFont);
    }
    int fontSize = painter.font().pixelSize();
    auto newFont = painter.font();
    newFont.setPixelSize(fontSize - 1);
    painter.setFont(newFont);
    painter.setPen(Qt::red);
```

```
            painter.drawText(QRect(10, 10, 700, 350), Qt::TextWordWrap |
Qt::AlignCenter, textString);
        if(textImage.save("textImage.jpg") == false)
        {
            cerr << "Error. Cannot save file" << endl;
            return 1;
        }
        QTimer::singleShot(0, QCoreApplication::instance(), SLOT(quit()));
        return a.exec();
    }
```

Результаты тестирования

Таблица 7 — Таблица с результатами тестирования программы из 7 задания

Тестовый набор	Ожидаемый результат	Действительный результат
10	Изображение,соответствующее критериям	
127	Изображение,соответствующее критериям	
782	Изображение, соответствующее критериям	

Hello	Error. Input is not integer number. Please, try again	Error. Input is not integer number. Please, try again
Запуск в директории, где у пользователя нет прав на запись	Error. Cannot save file	Error. Cannot save file

ЗАДАНИЕ 8

Текст задания

Пользователь с клавиатуры вводит строку, состоящую из нескольких слов. После завершения ввода каждое слово из строки должно зеркально отобразиться. Пример. Исходный текст: Человек написал текст. Преобразованный текст: кеволеЧ ласипан тскет.

Язык программирования

C++/Qt

Код программы

main.cpp

```
#include <iostream>
#include <string>
#include <QString>
#include <QRegularExpression>
int main()
{
   std::string tmpString;
   std::getline(std::cin, tmpString);
   QString string(tmpString.c_str());
                auto words = string.split(QRegularExpression("\\b",
QRegularExpression::UseUnicodePropertiesOption));
   for(auto word = words.begin() + 1; word != words.end(); word++)
   {
      if(word->simplified().length() > 1)
      {
         std::reverse(word->begin(), word->end());
```

```
            }
            std::cout << word->toStdString();
        }
        std::cout << std::endl;
        return 0;
    }
```

Результаты тестирования

Таблица 8 — Таблица с результатами тестирования программы из 8 задания

Тестовый набор	Ожидаемый результат	Действительный результат
Hello, how are you?	olleH, woh era uoy?	olleH, woh era uoy?
Привет, как твои дела?	тевирП, как иовт алед?	тевирП, как иовт алед?
А, зачем, в, этом, предложении, , . Так ? Много № знаков- - препинания? ? ? ?!?!	А, мечаз, в, мотэ, иинежолдерп . , ,каТ ? огонМ № воканз - - -яинаниперп!?!? ? ? ?	А, мечаз, в, мотэ, иинежолдерп . , ,каТ ? огонМ № воканз - - - яинаниперп!?!? ? ? ?
12345 54321 123444566666788888889000000000	54321 12345 000000000988888887666665444321	54321 12345 000000000988888887666665444321
1234567890987654321 - А переверни ка это	1234567890987654321 - А инревереп ак отэ	1234567890987654321 - А инревереп ак отэ