**Билет 10**

1. Дружественные функции и классы. Привести примеры.

Дружественные функции и классы — это функции или другие классы, которые могут обращаться к приватным и защищённым членам другого класса.

Пример:

#include <iostream>

**using** **namespace** std**;**

class MyClass **{**

private**:**

int secretValue**;**

public**:**

MyClass**()** **:** secretValue**(**42**)** **{}** // Конструктор инициализирует секретное значение

// Объявление дружественной функции

friend void showSecret**(**MyClass**&** obj**);**

**};**

// Дружественная функция может получать доступ к приватным данным класса

void showSecret**(**MyClass**&** obj**)** **{**

cout **<<** "Секретное значение: " **<<** obj**.**secretValue **<<** endl**;**

**}**

int main**()** **{**

MyClass myObject**;**

showSecret**(**myObject**);** // Доступ к приватному члену класса через дружественную функцию

**return** 0**;**

**}**

Здесь функция showSecret объявлена дружественной для класса MyClass, поэтому она может получить доступ к его приватному члену secretValue.

2. Используя объектно-ориентированный подход (допускается STL) написать программу, выполняющую действия:   
а) Создание динамического символьного массива, размер которого задается пользователем.

#include <iostream>

#include <vector>

**using** **namespace** std**;**

class CharArray **{**

private**:**

char**\*** arr**;** // Указатель на динамический массив

int size**;** // Размер массива

public**:**

// Конструктор для создания массива

CharArray**(**int size**)** **:** size**(**size**)** **{**

arr **=** **new** char**[**size**];** // Выделение памяти под массив

**}**

// Деструктор для освобождения памяти

**~**CharArray**()** **{**

**delete[]** arr**;**

**}**

// Метод для ввода данных в массив

void inputArray**()** **{**

cout **<<** "Введите " **<<** size **<<** " символов: "**;**

**for** **(**int i **=** 0**;** i **<** size**;** i**++)** **{**

cin **>>** arr**[**i**];**

**}**

**}**

// Метод для вывода массива

void printArray**()** **{**

**for** **(**int i **=** 0**;** i **<** size**;** i**++)** **{**

cout **<<** arr**[**i**]** **<<** " "**;**

**}**

cout **<<** endl**;**

**}**

// Возвращаем массив для дальнейшего использования

char**\*** getArray**()** **{**

**return** arr**;**

**}**

// Возвращаем размер массива

int getSize**()** **{**

**return** size**;**

**}**

**};**

int main**()** **{**

int n**;**

cout **<<** "Введите размер массива: "**;**

cin **>>** n**;**

CharArray myArray**(**n**);**

myArray**.**inputArray**();** // Ввод данных

myArray**.**printArray**();** // Вывод данных

**}**

в) шифровка и дешифровка текста методом многоалфавитной подстановки.

#include <iostream>

#include <string>

**using** **namespace** std**;**

class Cipher **{**

private**:**

string alphabet **=** "ABCDEFGHIJKLMNOPQRSTUVWXYZ"**;** // Алфавит для шифрования

// Метод для шифрования символа

char encryptChar**(**char c**,** int key**)** **{**

int index **=** alphabet**.**find**(**toupper**(**c**));** // Находим индекс символа в алфавите

**if** **(**index **==** string**::**npos**)** **{**

**return** c**;** // Если символ не найден (например, пробел или знак препинания), оставляем его без изменений

**}**

**return** alphabet**[(**index **+** key**)** **%** alphabet**.**size**()];** // Шифруем символ

**}**

// Метод для дешифрования символа

char decryptChar**(**char c**,** int key**)** **{**

int index **=** alphabet**.**find**(**toupper**(**c**));**

**if** **(**index **==** string**::**npos**)** **{**

**return** c**;**

**}**

**return** alphabet**[(**index **-** key **+** alphabet**.**size**())** **%** alphabet**.**size**()];** // Дешифруем символ

**}**

public**:**

// Метод для шифрования текста

string encrypt**(**string text**,** int key**)** **{**

string encryptedText **=** ""**;**

**for** **(**char c **:** text**)** **{**

encryptedText **+=** encryptChar**(**c**,** key**);**

**}**

**return** encryptedText**;**

**}**

// Метод для дешифрования текста

string decrypt**(**string text**,** int key**)** **{**

string decryptedText **=** ""**;**

**for** **(**char c **:** text**)** **{**

decryptedText **+=** decryptChar**(**c**,** key**);**

**}**

**return** decryptedText**;**

**}**

**};**

int main**()** **{**

string text**;**

int key**;**

cout **<<** "Введите текст для шифрования: "**;**

getline**(**cin**,** text**);**

cout **<<** "Введите ключ для шифрования (целое число): "**;**

cin **>>** key**;**

Cipher cipher**;**

string encryptedText **=** cipher**.**encrypt**(**text**,** key**);**

cout **<<** "Зашифрованный текст: " **<<** encryptedText **<<** endl**;**

string decryptedText **=** cipher**.**decrypt**(**encryptedText**,** key**);**

cout **<<** "Расшифрованный текст: " **<<** decryptedText **<<** endl**;**

**return** 0**;**

**}**

 **Метод encrypt** шифрует текст, используя ключ для сдвига символов.

 **Метод decrypt** восстанавливает исходный текст, используя тот же ключ.