УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационной безопасности

Кафедра инфокоммуникационных технологий

ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ ИНФОКОММУНИКАЦИОННЫХ СИСТЕМ
Часть 1

Лабораторная работа 4
Двумерный динамический массив. Дружественные функции. Чекбоксы и радиобаттоны в Qt

Минск 2022

Макейчик Е.Г.

Содержание

Лабораторная работа 4
Двумерный динамический массив. Дружественные функции. Чекбоксы и радиобаттоны в Qt

Цель работы: Изучить варианты создания и уничтожения динамических двумерных массивов. Научиться использовать дружественные функции. Научиться использовать чекбоксы и радиобаттоны в Qt.

Двумерный динамический массив

Двумерный динамический массив эквивалентен одномерному массиву, в котором каждый элемент является указателем. Этот одномерный массив является строкой; каждый указатель указывает на пространство хранения, которым является каждый столбец. Двухмерный динамический массив реализуется с помощью функций malloc, calloc, оператора new и др. для динамического выделения пространства для каждого элемента.

В программе ниже продемонстрировано создание целочисленного массива 3×4 с использованием оператора new.

```
#include <iostream>
using namespace std;
int main()
{
    // Создание двумерного массива
    int** mass = new int*[3]; // Создание массива указателей
    for (int i = 0; i < 3; i++)
    {
        mass[i] = new int[4]; // Создание еще одного массива для каждого указателя
    }
    // Заполнение массива
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            mass[i][j] = i + j;
            // Альтернатива: *(*(mass + i) + j) = i+j;
        }
    }
    // Вывод массива
    for (int i = 0; i < 3; i++)
    {
        cout << endl;
        for (int j = 0; j < 4; j++)
        {
            cout << " " << mass[i][j];
        }
    }
    // Удаление двумерного массива
    for (int i = 0; i < 3; i++)
    {
        delete[] mass[i]; // Освобождение памяти занятой элементами
    }
    delete[] mass; // Освобождение памяти занятой под массив указателей
}
```

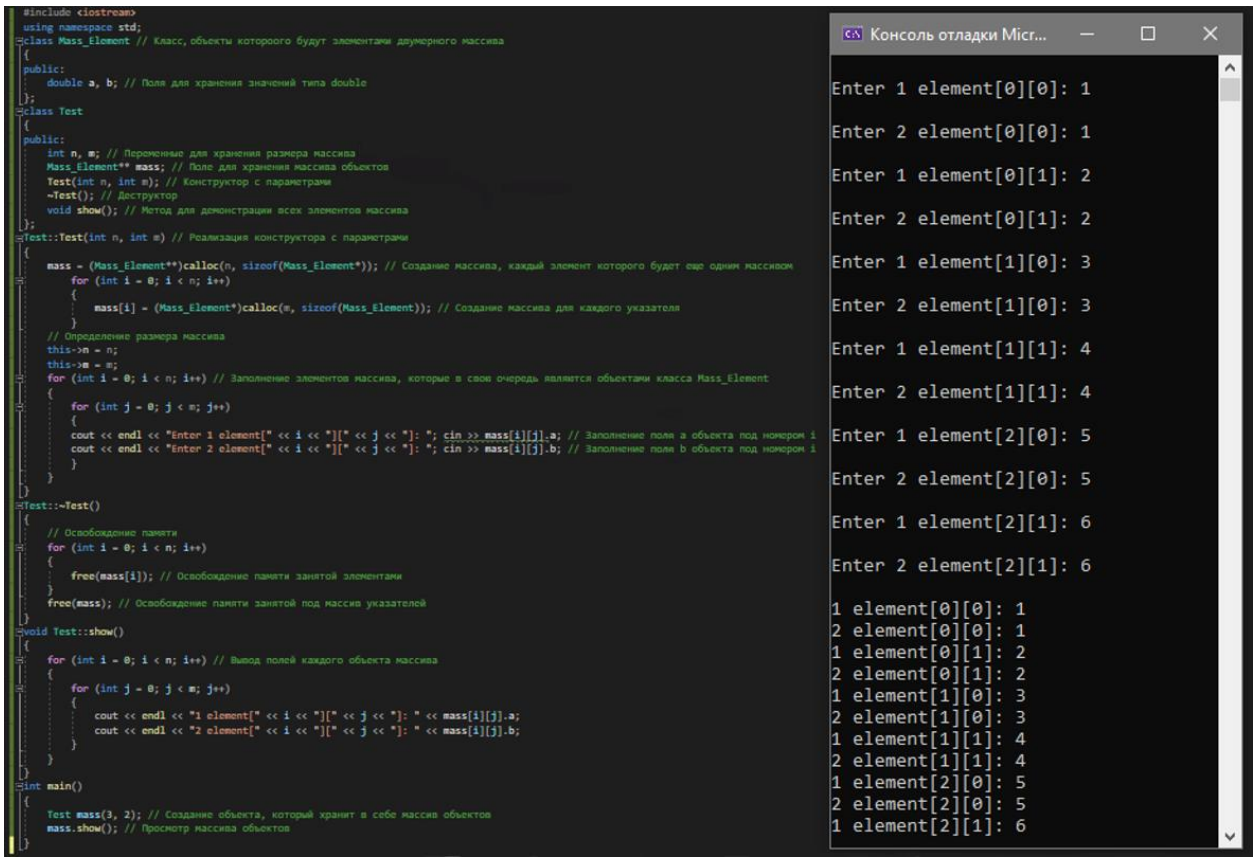
```
0 1 2 3
1 2 3 4
2 3 4 5
```

В программе ниже продемонстрировано создание целочисленного массива 3×4 с использованием функций для выделения и освобождения памяти.

```
#include <iostream>
using namespace std;
int main()
{
    // Создание двумерного массива
    int** mass = (int**)calloc(3, sizeof(int*)); // Создание массива указателей
    for (int i = 0; i < 3; i++)
    {
        mass[i] = (int*)calloc(4, sizeof(int)); // Создание еще одного массива для каждого указателя
    }
    // Заполнение массива
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            mass[i][j] = i + j;
            // Альтернатива: *(*(mass + i) + j) = i+j;
        }
    }
    // Вывод массива
    for (int i = 0; i < 3; i++)
    {
        cout << endl;
        for (int j = 0; j < 4; j++)
        {
            cout << " " << mass[i][j];
        }
    }
    // Удаление двумерного массива
    for (int i = 0; i < 3; i++)
    {
        free(mass[i]); // Освобождение памяти занятой элементами
    }
    free(mass); // Освобождение памяти занятой под массив указателей
}
```

```
0 1 2 3
1 2 3 4
2 3 4 5
```

В программе ниже продемонстрировано создание двумерного массива объектов.

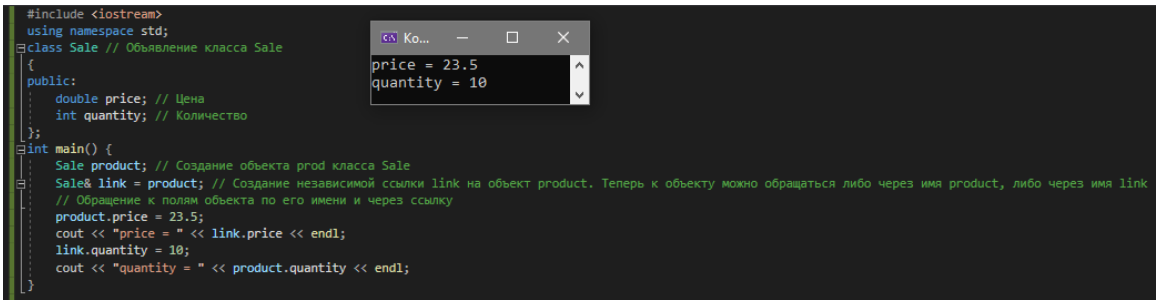


Ссылки на объекты

На объекты, как и на переменные базовых типов, можно выполнять ссылки. Ссылка – это фактически псевдоним объекта или переменной. Существует несколько способов использования ссылок.

Независимая ссылка на объект позволяет использовать для одного и того же объекта разные названия. Ссылка при объявлении сразу инициализируется. В качестве значения ссылки указывается объект, для которого создается ссылка. Перед именем ссылки в объявлении указывается оператор &. В качестве типа ссылки указывают имя класса, на объект которого выполняется ссылка. После создания ссылки к объекту можно обращаться через его имя или через имя ссылки на этот объект.

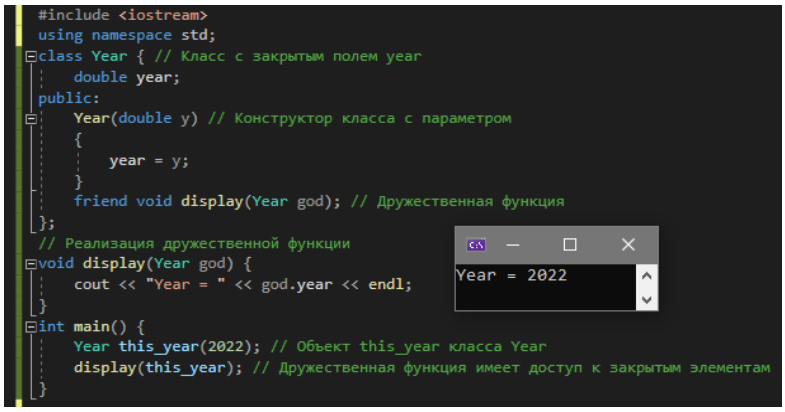
имя_класса имя_объекта;
имя_класса &имя_ссылки=имя_объекта;



Дружественные функции

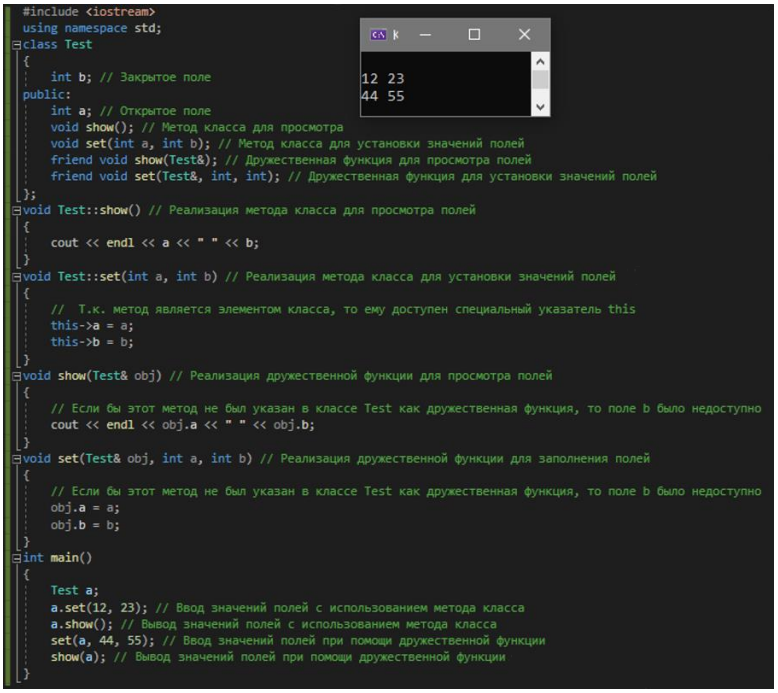
Как отмечалось ранее, элементы класса могут быть закрытыми и открытыми. К закрытым элементам класса доступ существует только внутри класса. Однако нередко случается необходимость, чтобы элемент класса оставался закрытым, но некоторые внешние методы имели к нему доступ. Такие внешние методы, которые имеют доступ к закрытым элементам класса, называются *дружественными функциями*.

Чтобы задекларировать метод как дружественную функцию для класса, необходимо указать прототип этой функции в открытой public секции описания класса, предварив его ключевым словом friend.



Особенности использования дружественной функции:

- дружественную функцию класса необходимо декларировать в описании этого класса. В форме описания приводится прототип, а ее полное определение – вне класса, при этом не нужно указывать операцию привязки к данному классу, т.к. она не является элементом данного класса.
- при обращении к дружественной функции не нужно использовать операции привязки (.) или (→).
- т.к. как дружественная функция – обычная функция, у нее отсутствует первый скрытый параметр this. Для обращения к элементам класса из дружественной функции используют или ссылку, или указатель.



Дружественные функции класса кроме ссылки или указателя на объекты класса могут иметь обычные параметры, например, для инициализации данных объектов класса. В примере, представленном выше используется дружественная функция set(Test& obj, int a, int b), которая принимает в качестве аргументов еще переменные типа int, для установки новых значений полям.

Чекбоксы и радиобаттоны в Qt

Чекбоксы в Qt

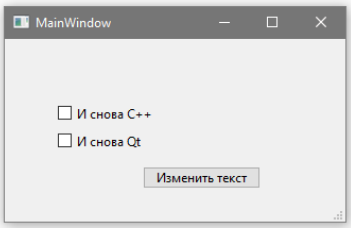
QCheckBox – это виджет, который имеет два состояния: установлен и не установлен. Флажки обычно используются для представления функций в

приложении, которые можно включить или отключить, не затрагивая другие виджеты, смотреть.

```cpp
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->label->setVisible(false); // В конструкторе, с помощью метода setVisible скрываем изображения C++, аргумент false
    ui->label_2->setVisible(false); // В конструкторе, с помощью метода setVisible скрываем изображения Qt, аргумент false
}

MainWindow::~MainWindow()
{
    delete ui;
}


void MainWindow::on_checkBox_stateChanged(int arg1) // Параметр arg1 определяет установлен ли флажок
{
    // При клике на чекбокс необходимо выяснить появился флажок или исчез
    if(arg1==Qt::Checked) // Проверяем аргумент
    {
       ui->label->setVisible(true); // Если флажок появился - отображаем картинку
    }
    else
    {
       ui->label->setVisible(false); // Если флажок исчез - скрываем картинку
    }
}


void MainWindow::on_checkBox_2_stateChanged(int arg1) // Параметр arg1 определяет установлен ли флажок
{
     // При клике на чекбокс необходимо выяснить появился флажок или исчез
     if(arg1==Qt::Checked) // Проверяем аргумент
     {
        ui->label_2->setVisible(true); // Если флажок появился - отображаем картинку
     }
     else
     {
        ui->label_2->setVisible(false); // Если флажок исчез - скрываем картинку
     }
}
```

Для изменения текста в чекбоксе в процессе работы программы необходимо использовать метод setText класса QCheckBox.

```cpp
void MainWindow::on_pushButton_clicked()
{
    ui->checkBox->setText("И снова C++"); // Меняем текст первого чекбокса используя метод setText
    ui->checkBox_2->setText("И снова Qt"); // Меняем текст второго чекбокса используя метод setText
}
```



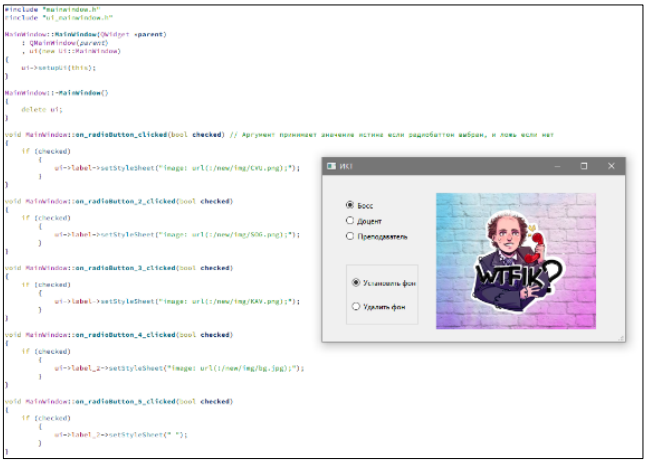
Для того чтобы узнать состояние чекбокса необходимо использовать метод isChecked класса CheckBox.

```cpp
if(ui->checkBox->isChecked())
{
    // Если флажок установлен то сделать ...
}

if(!ui->checkBox->isChecked())
{
    // Если флажок не установлен то сделать ...
}
```

Радиобаттоны в Qt

QRadioButton – это переключатель, который можно включить (отметить) или выключить (снять флажок). Переключатели обычно предоставляют пользователю выбор «один из многих». В группе переключателей одновременно можно проверить только один переключатель; если пользователь выбирает другой переключатель, ранее выбранный переключатель отключается

Для корректной работы радиобаттоны необходимо Скомпоновать по сетке или использовать GroupBox, смотреть.



Для того чтобы сделать радиобаттоны активными по умолчанию необходимо в конструкторе использовать метод setChecked.

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->radioButton->setChecked(true); // Делаем активными по умолчанию первый и четвертый radioButton
    ui->radioButton_4->setChecked(true);

    if (ui->radioButton->isChecked()) // Если первый радиобаттон активирован, показать изображение
    {
            ui->label->setStyleSheet("image: url(:/new/img/CVU.png);");
    }
    if (ui->radioButton_4->isChecked()) // Если четвертый радиобаттон активирован, показать изображение
    {
            ui->label_2->setStyleSheet("image: url(:/new/img/bg.jpg);");
    }
}
```

Задание к лабораторной работе 4

Для выполнения лабораторной работы необходимо установить и настроить Visual Studio и инструменты Qt. Задание к лабораторной работе 4 состоит из нескольких задач. Задачи выполняются по вариантам (например, Вариант 1 – номера по списку в группе: 1, 9, 17, 25; Вариант 2 – номера по списку в группе: 2, 10, 18, 26 и т.д.).

Вариант 1

Задача 1

Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

Вы очень любите игру "Heroes of Might and Magic" и поэтому решили написать свою версию этой замечательной игры. Разумеется, написать игру полностью вы не успеете в связи с кучей других заказов, но прописать поле битвы за предстоящие выходные вполне. Поле битвы будет выглядеть как квадратная матрица размером N×N (N>1). В каждую ячейку вводится количество солдат, стоящих на этой ячейке. Солдаты, расположенные в ячейках выше главной диагонали – ваши; солдаты, расположенные в ячейках ниже главной диагонали, принадлежат противнику; солдаты, расположенные на главной диагонали, являются ничейными и вообще не учитываются при сражении. Вам необходимо создать класс Arena в котором будет двумерный массив N×N и заполнить его случайными значениями в диапазоне от 0 до 1000. Написать дружественную функцию run, которая выведет количество ваших солдат и количество солдат противника.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (количество ваших солдат и количество солдат противника). Для реализации выбора необходимо использовать QCheckBox. За основу необходимо взять изображения, размещенные снизу.

Вариант 2

Задача 1

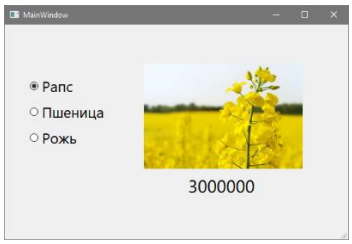
Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

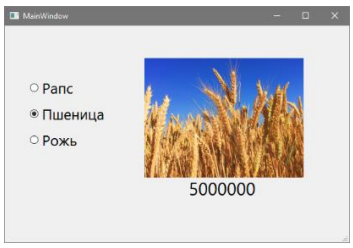
Вас попросили помыть линзы телескопа, и вы решили выполнить задание добросовестно и помыть их и изнутри. После этого вас в срочном порядке перевели в лабораторию растениеводства и направили на недельную работу в поле. Приехав на место, вы увидели поле, разделенное на участки очень похожие на матрицу размером N×M. На каждом участке растет разное количество растений, и ваша задача посчитать количество всех растений. Вам необходимо создать класс Garden в котором будет двумерный массив N×M и заполнить его случайными значениями в диапазоне от 0 до 100. Написать дружественную функцию run, которая выведет количество всех растений в поле.

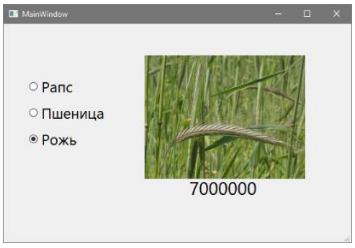
Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

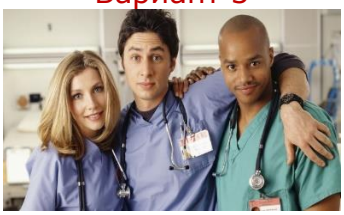
Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (рапс, пшеница, рожь). Для реализации выбора необходимо использовать QRadioButton. За основу необходимо взять изображения, размещенные снизу.







Вариант 3



Задача 1

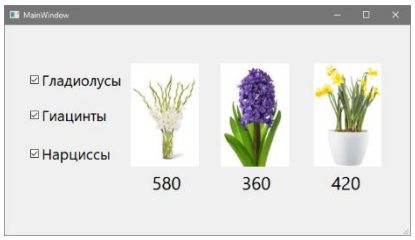
Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

Работа главврача очень сложна и для того, чтобы отдохнуть вы иногда выглядываете в окно чтобы полюбоваться на клумбу с цветами. Вы уже давно заметили, что клумба разделена на участки и очень похожа на матрицу размером N×M. Сегодня у вас не очень загруженный день, и вы решили посчитать количество цветов по краю клумбы. Вам необходимо создать класс Lower_Bed в котором будет двумерный массив размером N×M и заполнить его случайными значениями от 0 до 10. Написать дружественную функцию run, которая посчитает количество цветов по краям клумбы.

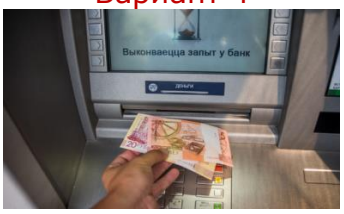
Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (Гладиолусы, Гиацинты, Нарциссы). Для реализации выбора необходимо использовать QCheckBox. За основу необходимо взять изображения, размещенные снизу.



Вариант 4



Задача 1

Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

В банке есть хранилище, и ваш начальник спать не может, не зная сколько в среднем хранится денег в ячейках и сколько ячеек, в которых количество денег больше среднего значения. Вам необходимо ему помочь, ибо у сонного начальника сотрудники не получают премий. Хранилище давно напоминало вам матрицу размера N×M, а с матрицами вы работать умете. Вам необходимо создать класс Storage в котором будет двумерный массив размером N×M и заполнить его случайными значениями от 0 до 100000. Написать дружественную функцию run, которая посчитает среднее количество денег во всех ячейках и количество ячеек в которых хранится денег больше среднего значения.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (среднее количество денег во всех ячейках и количество ячеек в которых хранится денег больше среднего значения). Для реализации выбора необходимо использовать QCheckBox. За основу необходимо взять изображения, размещенные снизу.

Вариант 5

Задача 1

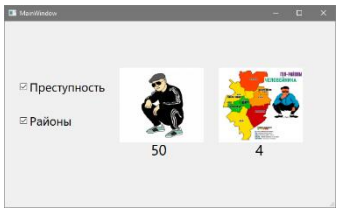
Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

Вам нужно составить график патрулей по вашему городу, но для этого необходимо знать уровень преступности в каждом районе, чтобы знать сколько и куда необходимо отправить патрулей (нет смысла посылать много патрулей в район, где живут пенсионеры, они все равно дома сидят и на улицу не выходят). Когда вы общались с заключенными программистами, они научили вас такой крутой штуке как матрица и вы решили разделить ваш город на равные участки чтобы получилась матрица размером N×M и сформировать каждому участку свой уровень преступности. Вам необходимо создать класс City в котором будет двумерный массив размером N×M и заполнить его случайными значениями от 0 до 100. Написать дружественную функцию run, которая выведет средний уровень преступности в городе и количество районов в которых уровень преступности превосходит среднее значение.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (средний уровень преступности в городе и количество районов в которых уровень преступности превосходит среднее значение). Для реализации выбора необходимо использовать QCheckBox. За основу необходимо взять изображения, размещенные снизу.



Вариант 6

Задача 1

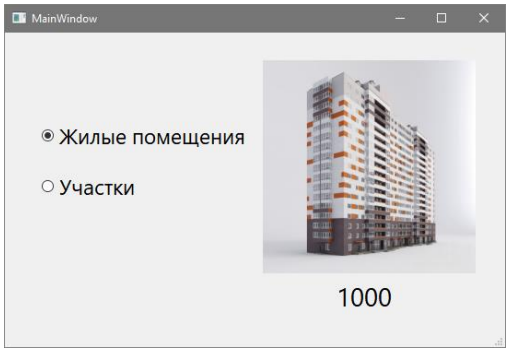
Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

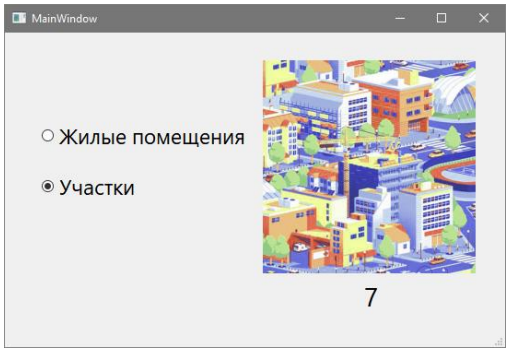
Вы решили запустить социальную программу, благодаря которой каждая многодетная семья, у которой больше четырех детей получала бы бесплатно квартиру. Но перед этим вам не помешало бы узнать сколько вообще жилых домов в вашем городе. На курсе "Высшая математика за 2 дня легко и с картинками" вам рассказывали про матрицу, и вы решили разделить ваш город на одинаковые участки в результате чего у вас получилась матрица N×M. Вам необходимо создать класс City в котором будет двумерный массив размером N×M и заполнить его случайными значениями от 0 до 1000. Написать дружественную функцию run, которая выведет среднее количество жилых помещений в городе и количество участков в которых количество жилых домов меньше среднего.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (среднее количество жилых помещений в городе и количество участков в которых количество жилых домов меньше среднего). Для реализации выбора необходимо использовать QRadioButton. За основу необходимо взять изображения, размещенные снизу.





Вариант 7

Задача 1

Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

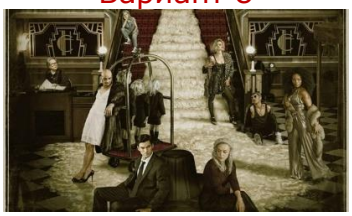
8 марта приближается, гольфы уже изготовлены и собраны по парам, но тут вы поняли, что такой подарок без коробки конфет – это минус к вашей прибыли от этого прекрасного праздника. Вы опытный управленец и заказали срочную доставку пары тонн конфет и прямоугольных коробок на Aliexpress. Но необходимо внимательней читать описание товара. Коробки, которые доставили имели слишком большое отделение для конфеты и в каждую ячейку могло поместится несколько конфет. Подумав немного, вы решили сделать это фишкой ваших коробок с конфетами типа: "Никогда не знаешь сколько конфет внутри коробки". Своим рабочим вы сказали класть в каждую ячейку разное количество конфет. Проверка первой партии показала, что рабочие решили немого схитрить и иногда не вкладывали конфету в ячейку. Это никуда не годится и вам срочно требуется программа для проверки наполненности коробки конфетами. Вам необходимо создать класс Box_Of_Candies в котором будет двумерный массив размером N×M и заполнить его случайными значениями от 0 до 5. Написать дружественную функцию run, которая выведет номера ячеек без конфет.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (крем-кокос, лесные ягоды, волшебный орешек). Для реализации выбора необходимо использовать QCheckBox. За основу необходимо взять изображения, размещенные снизу.

Вариант 8



Задача 1

Задача 1 выполняется с использованием Visual Studio. Пользовательский класс должен содержать конструктор с параметрами для создания динамических целочисленных массивов (оператор new или стандартная библиотечная функция calloc). Размеры массива – число строк и столбцов передается в конструктор через параметры, заполнение массива происходит в конструкторе. Класс так же должен содержать деструктор, который освободит память. Реализовать дружественные функции для: просмотра текущего состояния массива; установки новых значений элементов массива.

В своем отеле вы решили открыть ресторан с названием "Матрица". Название пришло вам в голову из-за того, что столы расположены на одинаковом расстояние друг от друга и схема их расположения напоминает вам старую добрую матрицу размером N×M. У вас современный ресторан и можно бронировать столик заранее, каждый стол вмещает в себя пять человек. Очень скоро приедет большая группа туристов, нужно срочно узнать, сколько столиков в ресторане уже занято. Вам необходимо создать класс Restaurant в котором будет двумерный массив размером N×M и заполнить его случайными значениями от –5 до 5 ("–" будет означать, что столик забронирован, но люди, которые его бронировали еще не пришли, это значит, что за этот столик вполне можно посадить туристов). Так же туристов можно посадить за пустые столы, то есть значение = 0). Написать дружественную функцию run, которая выведет количество столов, за которые можно посадить туристов.

Задача 2

Задача 2 выполняется с использованием инструментов Qt. Необходимо дать поясняющие комментарии к коду.

Для лучшей наглядности вам необходимо создать приложении, которое будет предоставлять возможность выбора с дальнейшим отображением (количество занятых столов, количество свободных столов). Для реализации выбора необходимо использовать QRadioButton. За основу необходимо взять изображения, размещенные снизу.