Курсовая работа

По дисциплине «Информатика»

Тема: «Объектно-ориентрованное программирование»

Выполнил:

студент группы [EMPTY]

[YOUR NAME]

Принял:

[TEACHER NAME]

Пояснительная записка имеет следующую структуру:

* Введение
* Класс, инкапсулирующий массив
* Имитационное моделирование
* Простое наследование и полиморфизм
* Перегрузка операций

Введение

**Объектно-ориентиированное программирование (ООП)** — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Основные принципы структурирования в случае ООП связаны с различными аспектами базового понимания предметной задачи, которое требуется для оптимального управления соответствующей моделью:

* абстракции для выделения в моделируемом предмете важного для решения конкретной задачи по предмету, в конечном счёте — контекстное понимание предмета, формализуемое в виде класса;
* инкапсуляция для быстрой и безопасной организации собственно иерархической управляемости: чтобы было достаточно простой команды «что делать», без одновременного уточнения как именно делать, так как это уже другой уровень управления;
* наследование для быстрой и безопасной организации родственных понятий: чтобы было достаточно на каждом иерархическом шаге учитывать только изменения, не дублируя всё остальное, учтённое на предыдущих шагах;
* полиморфизм для определения точки, в которой единое управление лучше распараллелить или наоборот — собрать воедино.

В моих лабораторных работах реализованы все 4 основных принципа ООП. Именно об я и буду рассказывать в данном документе.

1. Класс, инкапсулирующий массив

Условие задачи:

Описать класс CArrayRemoveIfOdd для работы с массивами динамической длины. Дополнительный метод: удаление элементов, стоящих на четных местах.

Элементы данной программы:

* CArrayRemoveIfOdd имя инкапсулированного класса
* Конструктор, которой устанавливает начальную длину динамического массива и сам массив на NULL
* Деструктор, который освобождает память, выделенную для массива
* Метод CreateArray() для создания массива заданной длины
* Метод SetValue() для присвоения массиву значение
* Метод FillArrayRandValues() для заполнения массива случайными данными
* Метод DisplayArray() отображает элементы масссива
* Метод RemoveIfOdd() для удаления элементов на нечетных местах
* Поле nCount содержит длинну массива
* Поле arr динамический массив

Фрагмент программного кода

// main.cpp

void menu (void)

{

cout << "1 - создание массива\n";

cout << "2 - изменение элементов массива\n";

cout << "3 - заполнение массива случайными числами\n";

cout << "4 - вывод на экран элементов массива\n";

cout << "5 - удаление элементов массива с нечетным индексом\n";

cout << "ESC - выход\n\n";

}

int main (void)

{

int key, n, k, val;

CArrayRemoveIfOdd ar; // создаётся объект класса, при этом автоматически вызывается конструктор

srand(time(NULL)); // включаем генератор случайных чисел

setlocale(0, "RUS"); // обеспечиваем вывод в консольное окно символов кириллицы

menu ();

do {

key = getch();

switch(key)

{

case '1':

cout << "Введите количество элементов массива: ";

scanf("%d", &n);

ar.CreateArray(n);

break;

case '2':

cout << "Введите номер элемента массива: ";

scanf("%d", &k);

cout << "Введите значение элемента массива: ";

scanf("%d",&val);

ar.SetValue(k, val);

break;

case '3':

cout << "Массив заполнен случайными числами\n";

ar.FillArrayRandValues();

break;

case '4':

cout << "Вывод на экран элементов массива\n";

ar.DisplayArray();

break;

case '5':

cout << "Элементов массива с нечетным индексом удалены\n\n";

ar.RemoveIfOdd ();

break;

}

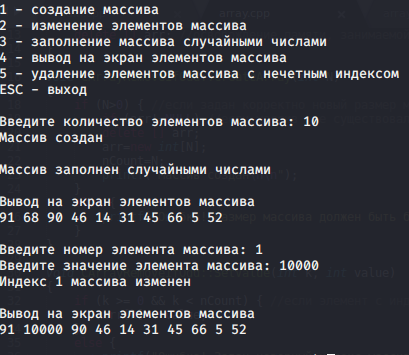
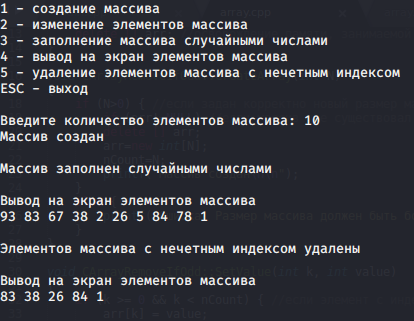
} while (key != KB\_ESCAPE); // код клавиши ESC

return 0; // объект класса уничтожается, автоматически вызывается деструктор

}

// main.cpp

Скриншоты работы программы:

 Рис. 1 Рис. 2

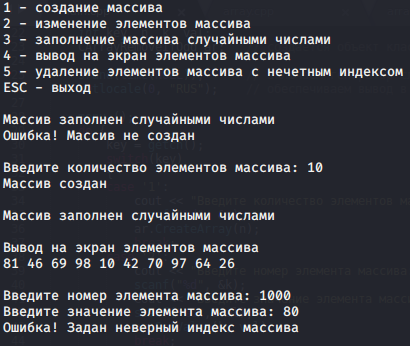
 Рис. 3

Рис. 1 показывает базовые операции, который можно произвести над массивами, такие как создание массива, заполнение случайными числами, добавление и удаление элементов.

Рис. 2 - удаление элементов массива стоящих на нечетных местах (вар 6). После создания массива и заполнения случайными числами (не обязательно) можно приступить к уделению данных элементов. Операция совершается при нажатии клавиши 5. На Рис. 2 можно увидеть что после данной операции количество элементов массива уменьшилось вдвое.

Рис. 3 показывает способность программы корректно обрабатывать ошибки, такие как удаление не существующего элемента массива, удаление элементов массива, который еще не создан. Программа способна не только выявить данные ошибки, но и вывести соответствующее сообщение пользователю.

2. Имитационное моделирование

Условие задачи:

Кондиционер имеет 3 кнопки: кнопка режима (холодный/горячий воздух); кнопка питания; кнопка работы. Кнопка режима действует только при выключенном питании. Кнопка работы приводит кондиционер в работу на одну минуту и только когда включено питание. Повторное нажатие на эту кнопку добавляет ещё одну минуту времени. При включенном питании нажатие на кнопку питания приводит к отключению кондиционера, обнулению счётчика времени и выключению питания. При выключенном питании нажатие кнопки питания приводит к включению кондиционера. Начальное состояние: питание выключено, режим – холодный воздух.

Элементы данной программы:

* Aircon класс имитирующий кондиционер
* enum state состояние включён/выключен
* enum mode определение режима работы кондиционера
* Поле curr\_button для описания состояния кнопки
* Поле curr\_mode для описания режима работы
* Поле curr\_power для описания состояния питания
* Поле time время, оставшееся до конца работы кондиционера
* Метод turn\_power() включает/выключает питание
* Метод change\_mode() изменяет режим работы
* Метод push\_button() включает/выключает работу кондиционера
* Метод deactivate() обрабатывает окончание работы кондиционера
* Метод Run() обрабатывает взаимодействие пользователя с кондиционером

Скриншоты работы программы:

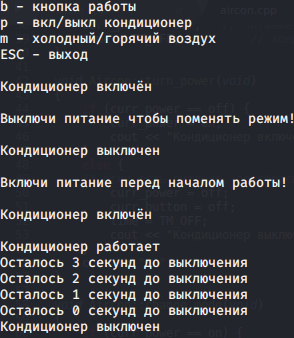
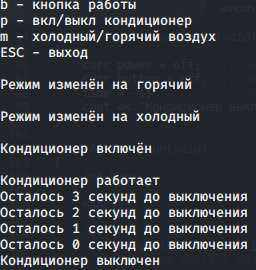


Рис. 1 Рис. 2

Рис. 1 отображает меню управления и выполнение правильной последовательности команд для работы виртуального кондиционера. Перед тем как включить устройство можно изменить режим на горячий или холодный, после включить питание и запустить сам кондиционер. После 5 сек. кондиционер выключится сам по себе. Таймер намеренно уменьшен, что правильно отобразить логи.

На Рис. 2 отображен механизм обработки ошибок и выведение предупредительного сообщения пользователю. Первая ошибка сообщает что нужно выключить питание перед тем как поменять режим. Вторая говорит о том что пользователь забыл включить питание.

3. Простое наследование и полиморфизм

Условия задачи:

Создать базовый класс «таблица» из N\*N полей, определив в нем функции добавления элемента в таблицу, удаление элемента из таблицы по значению, вывод таблицы на экран. Породить от него класс «упорядоченная таблица», переопределив функцию добавления элементов.

Элементы данной программы:

* Table базовый класс
* Поле x, y координаты элемента таблицы
* Поле table[N][N] таблица
* Метод add() виртуальный, для реализации перегрузки
* Метод del() удалить элемент из таблицы
* Метод show() вывести элементы таблицы в консоль
* Класс SortedTable производный от класса Table
* Метод add() перегруженный

Фрагмент программного кода

// main.cpp

int main (void)

{

Table otable; /\* ordinary table. \*/

SortedTable stable; /\* sorted table. \*/

Table \*table; //указатель нужен для реализации полиморфизма

int key1, key2;

int val; /\* where insert value to. \*/

setlocale(0, "RUS"); // обеспесиваем вывод в консольное окно символов кириллицы

menu1 ();

do {

key1 = getch();

switch(key1) {

case '1':

table=&otable;

break;

case '2':

table=&stable;

break;

}

if (key1 != KB\_ESCAPE) {//выбираем действие для выбранного типа таблицы

menu2 ();

do {

key2 = getch();

switch (key2) {

case '1':

cout << "Введите число (add): ";

val = getnum ();

table->add(val);

break;

case '2':

cout << "Введите число (del): ";

val = getnum ();

table->del(val);

break;

case '3':

table->show();

break;

}

} while (key2 != KB\_ESCAPE);

}

cout << "Выход из таблицы\n\n";

menu1 ();

} while (key1 != KB\_ESCAPE);

return 0;

}

// main.cpp

Скриншоты работы программы:

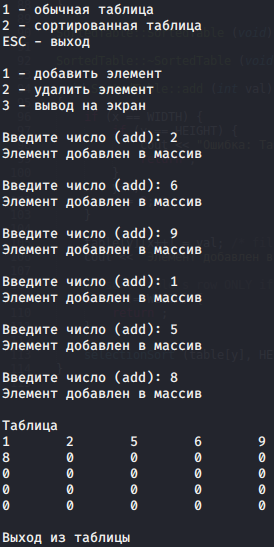
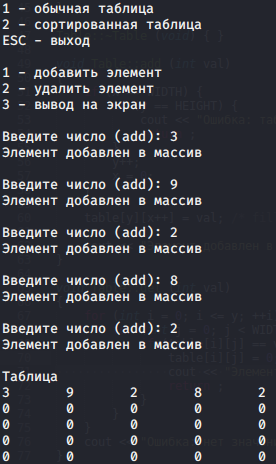


Рис. 1 Рис. 2

Рис. 1 отображает операции над обычной таблицей: добавление, удаление элементов и вывод. Сортировки в данном случае не происходит. Элементы отображаются в той же последовательности что были введены пользователем.

Рис. 2 отображает те же операции над сортированной таблицей. В данном случае после заполнения ряда, таблица делает сортировку от меньшего к большему.

Обработка ошибок

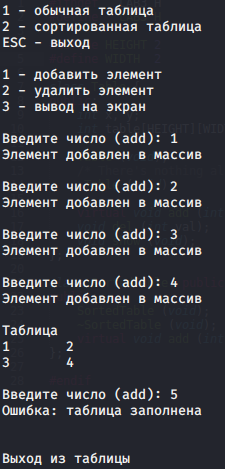
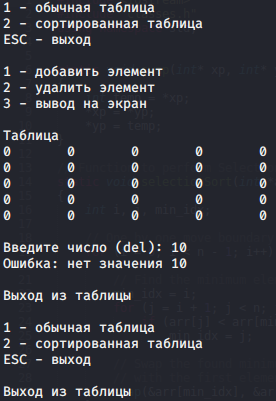


Рис . 3 Рис. 4

На Рис. 3 показано, как таблица обрабатывает ошибку связанную с удалением несуществующего элемента. В данном случае в таблице ничего не меняется. После выводится соответствующее сообщение пользователю.

На Рис. 4 показана обработка другой ошибки: переполнение. Если таблица полностью заполнена, при попытке добавить еще элемент программа выдает предупреждение пользователю. Данный механизм позволяет защитить данные в таблице.

4. Перегрузка операций

Условие задачи:

Создать класс, который запоминает целые двоичные числа в виде строк. Перегрузить операции –, – – и –=.

Элементы данной программы:

* Класс Bin2Dec основной класс по работе с бинарными строками
* Конструктор Bin2Dec() создает поле для бинарной строки
* Деструктор Bin2Dec() удаляет поле для бинарной строки
* Поле binstr содержит бинарную строку
* Поле count содержит длину бинарной строки
* Метод bin2dec конвертирует бинарную строку в десятичное число
* Метод dec2bin конвертирует десятичное число в бинарную строку
* Метод operator = перегруженный оператор присваивания
* Метод operator - перегруженный оператор вычитания
* Метод operator -- перегруженный оператор декремента
* Метод operator >> определение дружественной функции ввода строки
* Метод operator << определение дружественной функции вывода строки

Фрагмент программного кода

// main.cpp

int main (void)

{

int dec;

Bin2Dec bin1, bin2;

/\* ввети и вывести 1-ое число. \*/

cin >> bin1;

cout << bin1 << endl;

cout << "Ваше число (dec): " << bin1.convert() << endl << endl;

/\* вычесть из 2-го 1-ое число. \*/

cout << "Декремент первого числа\n";

bin1--;

cout << bin1 << endl;

cout << "Ваше число (dec): " << bin1.convert() << endl << endl;

cout << endl << endl;

/\* ввети и вывести 3-е число. \*/

cin >> bin2;

cout << bin2 << endl;

cout << "Ваше число (dec): " << bin2.convert() << endl << endl;

/\* вычесть из 1-го 2-ое число. \*/

cout << "Вычитаем 1-ое число из 2-его\n";

bin2 -= bin1;

cout << bin2 << endl;

cout << "Ваше число (dec): " << bin2.convert() << endl << endl;

/\* превратить бинарные строки в целое число. \*/

cout << "Превращаем строку в деситичное число\n";

dec = int (bin1);

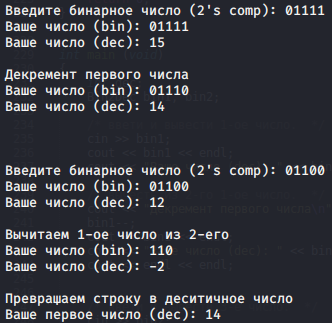
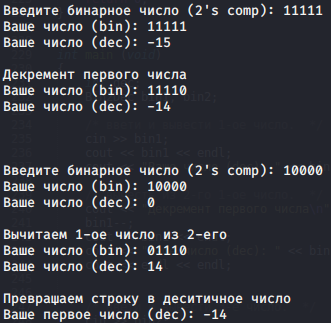
cout << "Ваше первое число (dec): " << dec << endl;

return 0;

}

// main.cpp

Скриншоты работы программы:

 Рис. 1 Рис. 2

На Рис. 1 программа считывает бинарную строку (дополнительный код) от пользователя и производит над этими вычисления. Такие как декремент числа, вычитание 2 чисел и операция сокращенного вычитания в C++ (-=). На данном рисунке изображена работы с положительными числами.

На Рис. 2 программа производит те же операции, но уже над отрицательными числами. Также производит конвертацию бинарной строки в десятичное число.

Итог

Занимаясь данными лабораторными работами я научился создавать классы на языке С++, полиморфизм (позднее связывание) и инкапсулировать объекты из реального мира. Более детальнее познакомиться с данными программами можно в лабораторных работах 1-4, где я детальнее объяснил принцип работы каждого из программ.

Список использованной литературы

<https://en.wikipedia.org/wiki/C%2B%2B_classes>

<https://habr.com/ru/post/147927/>

<https://habr.com/ru/post/418133/>

<https://www.tutorialspoint.com/cplusplus/cpp_data_encapsulation.htm#:~:text=Encapsulation%20is%20an%20Object%20Oriented,from%20outside%20interference%20and%20misuse.&text=This%20means%20that%20they%20can,other%20part%20of%20your%20program.>