Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №8**

Дисциплина: Основы алгоритмизации и программирования

Тема: Блоковый ввод-вывод

Вариант 15

Выполнил:

Студент гр. ИВТ-20-2б

Чувашев Максим Алексеевич

Проверил:

Доцент кафедры ИТАС  
Полякова О.А

Пермь, 2021

**Цель работы**

Работа со структурами данных. Создание структур, объектов структур, полей и методов.

**Постановка задачи**

1. Создать структуру “Автомобиль”, которая будет иметь поля:

* Марка
* Год выпуска
* Цена
* Цвет

1. Удалить все элементы структуры, у которого год выпуска меньше заданного, добавить элемент в начало массива объектов структуры.

**Анализ задачи**

1. Типы данных, которые будут использованы:
   1. Тип integer:

* int MAX; // Переменная, хранящая количество объектов структуры.
* int year; // Переменная, хранящая год, который введет пользователь, и начиная с которого элементы структуры останутся в массиве.
* int Year; // Переменная, хранящая год элемента структуры(автомобиля).
* int Price; // Переменная, хранящая цену элемента структуры(автомобиля).
* int i; // Переменная, которая используется во внешнем цикле.
* int j; // Переменная, которая используется во внутреннем цикле.
  1. Тип string:
* string Mark; // Переменная, хранящая название элемента структуры(автомобиля).
* string Сolor; // Переменная, хранящая цвет элемента структуры(автомобиля).
  1. Тип void:
* void fill() // Функция(метод), которая будет заполнять элементы структуры.
* void Print() // Функция(метод), которая будет выводить на консоль элементы структуры и соответствующие этим элементам поля.
* void Check(int year, CARS\* car, int \*MAX) // Функция, которая будет сравнивать переменную year с полем объекта структуры Year.
* void Add(CARS\* car, int\* MAX) // Функция будет сдвигать все элементы вправо на 1 элемент.
  1. Динамический массив:
* CARS\* car = new CARS[MAX + 1]; // Динамический массив структуры CARS, имеющий имя car и имеющий размер MAX.

1. Какие с этими данными надо выполнить действия:
   1. int MAX;

* Необходимо присвоить значение данной переменной.

cout << "Введите количество машин:\t";

cin >> MAX;

while (MAX <= 0)

{

cout << "Введите количество машин:\t";

cin >> MAX;

}

* 1. int year;
* Необходимо ввести значение для данной переменной.

cout << "Введите год выпуска с которого начнется вывод:\t";

cin >> year;

while (year < 1806 || year > 2021)

{

cout << "\nВведите год выпуска:\t";

cin >> year;

}

* 1. int Year;
* Необходимо ввести значение для данного поля структуры.

cout << "Введите год выпуска:\t";

cin >> Year;

while (Year < 1806 || Year > 2021)

{

cout << "Введите год выпуска:\t";

cin >> Year;

}

* 1. int Price;
* Необходимо ввести значение для данного поля объекта структуры.

cout << "Введите цену автомобиля:\t";

cin >> Price;

while (Price < 0)

{

cout << "Введите цену автомобиля:\t";

cin >> Price;

}

* 1. int i и int j
* Переменные внешнего и внутреннего цикла, которые меняются автоматически на каждом круге.

for (int i = 0; i < \*MAX; i++)

{

if (car[i].Year < year)

{

for (int j = i; j < \*MAX - 1; j++) {

car[j] = car[j + 1];

}

(\*MAX)--;

i--;

}

}

* 1. void Check(int year, CARS\* car, int \*MAX)
* В данную функцию передается год, указатель на объект car структуры CARS и указатель на размер динамического массива
* Данная функция будет проходиться по динамическому массиву car структуры CARS и будет сравнивать поле Year каждого элемента массива с переменной year. В случае если значение поля Year элемента массива будет меньше переменной year, то данный элемент массива удаляется, а все последующий элементы массива сдвигаются влево на 1 элемент. Переменная размера массива так же уменьшается на 1.

void Check(int year, CARS\* car, int \*MAX)

{

for (int i = 0; i < \*MAX; i++)

{

if (car[i].Year < year)

{

for (int j = i; j < \*MAX - 1; j++) {

car[j] = car[j + 1];

}

(\*MAX)--;

i--;

}

}

}

* 1. void fill()
* Данный метод будет заполнять все поля объекта структуры.

void fill()

{

cout << "\nВведите название автомобиля:\t";

cin.ignore();

getline(cin, Mark);

cout << "Введите год выпуска:\t";

cin >> Year;

while (Year < 1806 || Year > 2021)

{

cout << "Введите год выпуска:\t";

cin >> Year;

}

cout << "Введите цену автомобиля:\t";

cin >> Price;

while (Price < 0)

{

cout << "Введите цену автомобиля:\t";

cin >> Price;

}

cout << "Введите цвет автомобиля:\t";

cin >> Сolor;

}

* 1. void Print()
* Данный метод будет выводить все поля структуры на консоль.

void Print()

{

cout << "\n\n";

cout << "Марка:\t" << Mark << endl;

cout << "Год:\t" << Year << endl;

cout << "Цена:\t" << Price << endl;

cout << "Цвет:\t" << Сolor << endl;

}

* 1. void Add(CARS\* car, int\* MAX)
* Данная функция получает в качестве параметров указатель на динамический массив car структуры CARS и указатель на размер этого динамического массива. Функция будет проходится по всем элементам массива и сдвигать их влево на единицу, для того чтобы потом можно было добавить в начало еще 1 элемент. И после выполнения сдвига увеличиваем переменную MAX на единицу.

void Add(CARS\* car, int\* MAX)

{

for (int i = \*MAX; i > 0; i--) {

car[i] = car[i - 1];

}

(\*MAX)++;

}

* 1. CARS\* car = new CARS[MAX + 1];
* Данный массив будет заполнятся путем вызова для каждого элемента массива метода fill()

for (int i = 0; i < MAX; i++)

{

car[i].fill();

}

* После заполнения массив будет выведен на экран с помощью вызова для каждого элемента массива метода Print()

for (int i = 0; i < MAX; i++)

{

car[i].Print();

}

* Вызывается функция Check(year, car, &MAX), действия которой описаны в пункте 2.6.
* После удаления всех ненужных элементов вызывается функция Add(car, &MAX), действия которой описаны в пункте 2.9.
* Для первого элемента массива вызывается метод fill().

car[0].fill();

* Удаление динамического массива с помошью оператора delete.

 delete[] car;

 3. В каком виде эти данные представлены для решения задачи.

3.1. В виде структуры CARS, которая имеет:

3.1.1. Поля:

* string Mark; - Поле, содержащее марку машины.
* int Year; - Поле, содержащее год выпуска машины.
* int Price; - Поле, содержащее цену машины.
* string Сolor; - Поле, содержащее цвет машины.

3.1.2. Методы:

* void fill() \\ Действия данного метода описаны в пункте 2.7.
* void Print() \\ Действия данного метода описаны в пункте 2.8.

4. Какими операторами реализован ввод и вывод.

4.1. Ввод:

* С помощью оператора cin, например:

cin >> Year;

* С помощью функции getline():

getline(cin, Mark);

 4.2. Вывод:

* С помощью оператора cout, например:

cout << "Введите год выпуска:\t";

1. Какими действиями реализованы поставленные задачи

5.1. Цикл for

* Используется для перемещения между элементами динамического массива car и заполнения всех полей каждого элемента данного массива.

for (int i = 0; i < MAX; i++)

{

car[i].fill();

}

* Используется для перемещения между элементами динамического массива и вывода всех полей всех элементов данного массива на консоль.

for (int i = 0; i < MAX; i++)

{

car[i].Print();

}

* Используется для сдвига всех элементов массива на единицу влево.

for (int i = \*MAX; i > 0; i--) {

car[i] = car[i - 1];

}

* Используется для перемещению между элементами массива, удалению некоторых элементов, и сдвига элементов массива.

for (int i = 0; i < \*MAX; i++)

{

if (car[i].Year < year)

{

for (int j = i; j < \*MAX - 1; j++) {

car[j] = car[j + 1];

}

(\*MAX)--;

i--;

}

}

 5.2. Цикл while.

* Используется для предотвращения ввода некорректных данных поля Year.

while (Year < 1806 || Year > 2021)

{

cout << "Введите год выпуска:\t";

cin >> Year;

}

* Используется для предотвращения ввода некорректных данных поля Price.

while (Price < 0)

{

cout << "Введите цену автомобиля:\t";

cin >> Price;

}

* Используется для предотвращения ввода некорректных данных переменной MAX.

while (MAX <= 0)

{

cout << "Введите количество машин:\t";

cin >> MAX;

}

* Используется для предотвращения ввода некорректных данных переменной year.

while (year < 1806 || year > 2021)

{

cout << "\nВведите год выпуска:\t";

cin >> year;

}

**Полный код**

#include <iostream>

#include <string>

using namespace std;

struct CARS

{

string Mark;

int Year;

int Price;

string Сolor;

void fill()

{

cout << "\nВведите название автомобиля:\t";

cin.ignore();

getline(cin, Mark);

cout << "Введите год выпуска:\t";

cin >> Year;

while (Year < 1806 || Year > 2021)

{

cout << "Введите год выпуска:\t";

cin >> Year;

}

cout << "Введите цену автомобиля:\t";

cin >> Price;

while (Price < 0)

{

cout << "Введите цену автомобиля:\t";

cin >> Price;

}

cout << "Введите цвет автомобиля:\t";

cin >> Сolor;

}

void Print()

{

cout << "\n\n";

cout << "Марка:\t" << Mark << endl;

cout << "Год:\t" << Year << endl;

cout << "Цена:\t" << Price << endl;

cout << "Цвет:\t" << Сolor << endl;

}

};

void Check(int year, CARS\* car, int \*MAX)

{

for (int i = 0; i < \*MAX; i++)

{

if (car[i].Year < year)

{

for (int j = i; j < \*MAX - 1; j++) {

car[j] = car[j + 1];

}

(\*MAX)--;

i--;

}

}

}

void Add(CARS\* car, int\* MAX)

{

for (int i = \*MAX; i > 0; i--) {

car[i] = car[i - 1];

}

(\*MAX)++;

}

int main()

{

setlocale(LC\_ALL, "RU");

int MAX;

cout << "Введите количество машин:\t";

cin >> MAX;

while (MAX <= 0) {

cout << "Введите количество машин:\t";

cin >> MAX;

}

CARS\* car = new CARS[MAX + 1];

for (int i = 0; i < MAX; i++)

{

car[i].fill();

}

for (int i = 0; i < MAX; i++)

{

car[i].Print();

}

int year;

cout << "Введите год выпуска с которого начнется вывод:\t";

cin >> year;

while (year < 1806 || year > 2021)

{

cout << "\nВведите год выпуска:\t";

cin >> year;

}

Check(year, car, &MAX);

for (int i = 0; i < MAX; i++)

{

car[i].Print();

}

Add(car, &MAX);

car[0].fill();

for (int i = 0; i < MAX; i++)

{

car[i].Print();

}

delete[] car;

return 0;

}

**Блок-Схема**



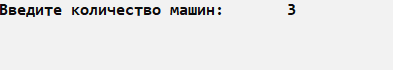






**Скриншоты результатов работы программы**

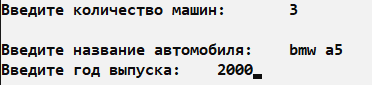
* 1. Нормальный ввод:
  2. Программа запрашивает у пользователя ввод количества автомобилей(Пример ввода: 3)



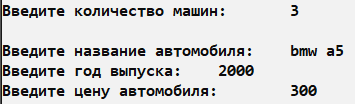
* 1. Программа запрашивает у пользователя ввод марки первого автомобиля. Ввод данного поля реализован через функцию getline(), поэтому мы можем вводить марку содержащую пробел, через оператор cin, такой опции не предоставилось бы. (Пример ввода: bmw a5).



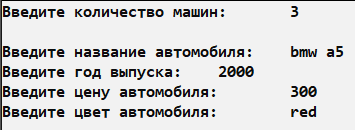
* 1. Программа запрашивает у пользователя ввод года выпуска первого автомобиля. Так как мы указываем год, здесь символ пробела попросту не нужен, поэтому ввод реализован через cin. (Пример ввода: 2000).



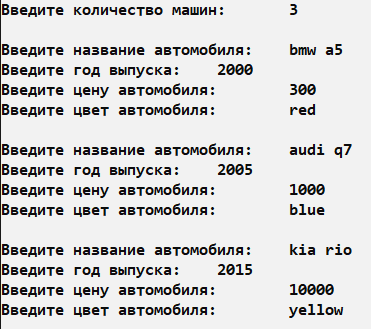
* 1. Программа запрашивает цену первого автомобиля. Ввод так же реализован через оператор cin. (Пример ввода: 300).



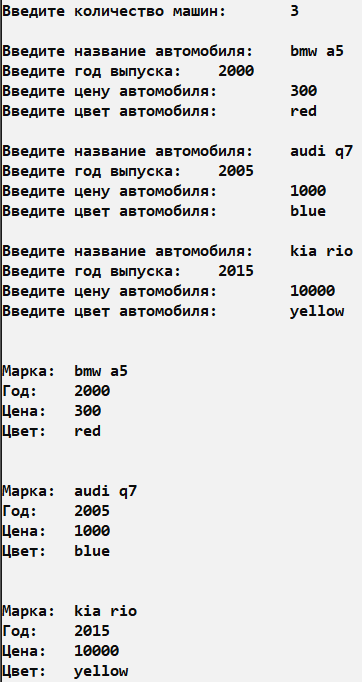
* 1. Программа запрашивает ввод цвета первого автомобиля. Ввод осуществляется через оператор cin. (Пример ввода: red).



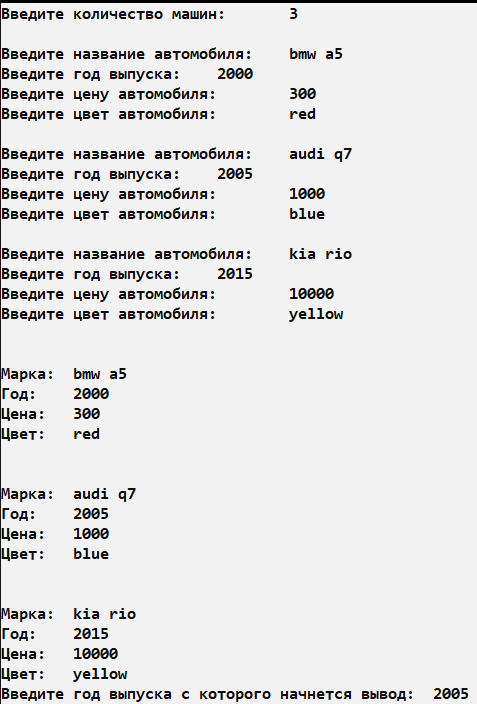
* 1. Далее программа будет делать повторные запросы заполнения полей, описанных в пунктах 1.2, 1.3, 1.4, 1.5, но уже для оставшихся автомобилей. В данном случае для двух оставшихся. Заполняем все запрашиваемые поля структуры для всех автомобилей.



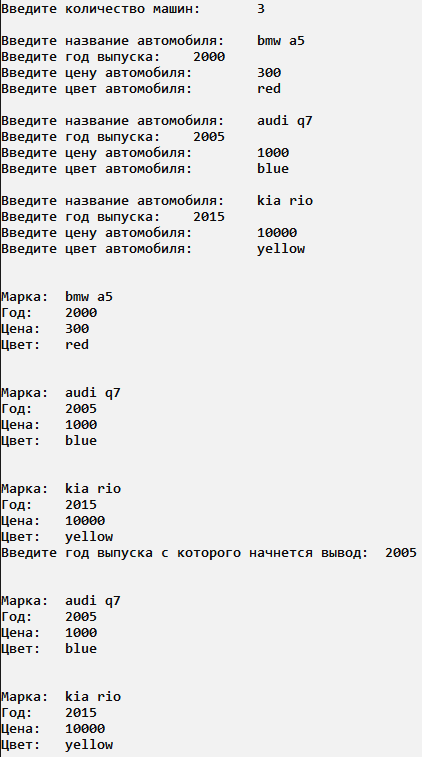
* 1. Программа выводит все автомобили со всеми введенными характеристиками на консоль.



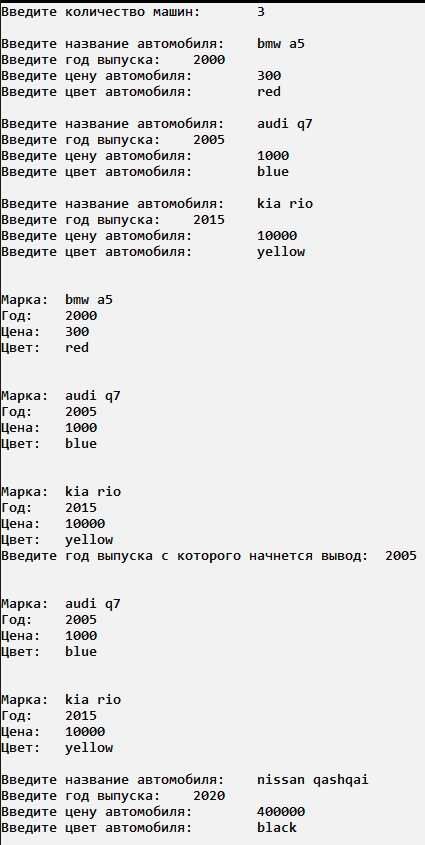
* 1. Программа запрашивает ввод пользователем года, начиная с которого автомобили останутся в массиве, все остальные, которые старше данного указанного года будут удалены. (Пример ввода: 2005).



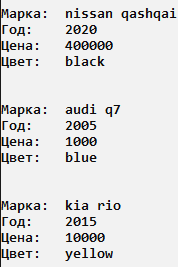
* 1. Программа выводит все оставшиеся автомобили, удовлетворяющие условию.



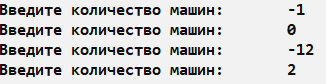
* 1. Программа должна добавить еще одну машину с введенными характеристиками в начало списка машин.
  2. Программа запрашивает ввод всех параметров новой машины исходя из пунктов 1.2, 1.3, 1.4, 1.5. Заполняем требуемые поля.



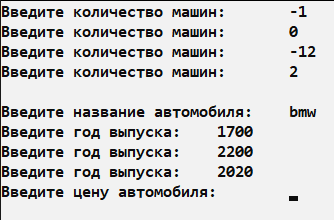
* 1. В конце программа выводит все оставшиеся автомобили удовлетворяющие условию + новый автомобиль с параметрами в начале списка.



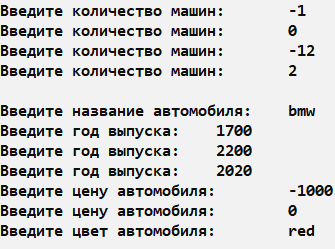
1. Попытка ввода с ошибками.
   1. Количество машин должно быть только положительным числом. (Пробуем ввести отрицательные в том числе 0) Программа продолжает выполнение только при вводе положительно значение (в данном примере 2)



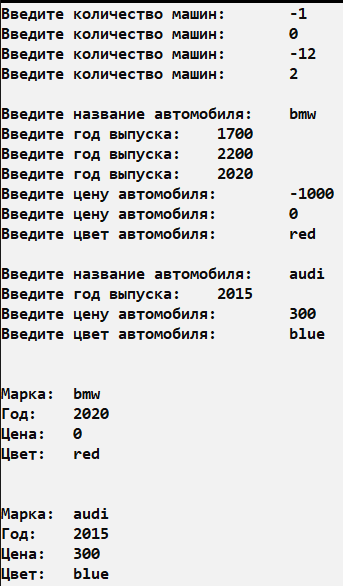
* 1. При вводе года выпуска автомобиля необходимо учитывать, что автомобили появились в 1806 году, а так же необходимо учитывать, что дата выпуска не может быть позже 2021 года.(Пробуем ввести данные не удовлетворяющие данным условиям). Программа продолжает выполнение только в случае ввода года, которое удовлетворяет условиям, в данном примере 2020.



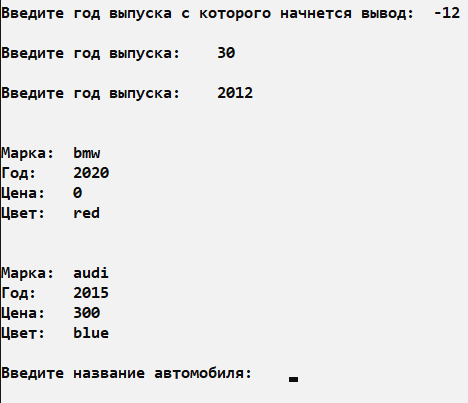
* 1. Так же необходимо учитывать, что цена автомобиля не может быть отрицательной, но может быть равной 0(например, машина получена в качестве подарка). Программа будет продолжать работу только при вводе положительной цены.



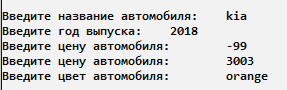
* 1. Продолжаем ввод для оставшихся автомобилей. Учитывая данные условия ввода.



* 1. Для ввода года, который станет условием существования автомобиля в массиве накладываем те же условия, что и для ввода года производства автомобиля. Программа продолжит свое выполнение только при вводе года, удовлетворяющего условию.



* 1. Заполняем все поля ввода для нового автомобиля который будет добавлен в начало списка машин. Программа продолжит работу только при условии соблюдения всех вышеописанных условий.



* 1. Программа выводит список машин, которые остались + добавленные.

