**Вопросы к экзамену для повышения квалификации по С++:**

**Первый вопрос:**

* Операторы в С++ и приоритеты операций. Оператор присваивания и его сокращенные формы

<https://ru.cppreference.com/w/cpp/language/operator_precedence>

//http://natalia.appmat.ru/c%26c%2B%2B/lezione2.php

<https://studfile.net/preview/1674641/page:5/>

<https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%8B_%D0%B2_C_%D0%B8_C%2B%2B#%D0%9F%D1%80%D0%B8%D0%BE%D1%80%D0%B8%D1%82%D0%B5%D1%82%D1%8B_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BE%D0%B2>

++ Суффиксный инкремент (i-=-i, i=i+1;

-- Суффиксный декремент

() Вызов функции

[] Взятие элемента массива

. Выбор элемента по ссылке

-> Выбор элемента по указателю

typeid() RTTI (только C++; см typeid)

const\_cast Приведение типа (C++) (см const cast)

dynamic\_cast Приведение типа (C++) (см dynamic cast)

reinterpret\_cast Каламбур типизации (C++) (см reinterpret\_cast)

static\_cast Приведение типа (C++) (см static cast)

++ Префиксный инкремент Справа налево

-- Префиксный декремент

+ Унарный плюс

- Унарный минус

! Логическое НЕ

~ Побитовое НЕ

(type) Приведение типа

\* Разыменование указателя

& Взятие адреса объекта

sizeof Sizeof (размер)

new, new[] Выделение динамической памяти (C++)

delete, delete[] Освобождение динамической памяти (C++)

.\* Указатель на член (C++) Слева направо

->\* Указатель на член (C++)

\* Умножение

/ Деление

% Получение остатка от деления

+ Сложение

- Вычитание

<< Побитовый сдвиг влево

>> Побитовый сдвиг вправо

< Меньше

<= Меньше или равно

> Больше

>= Больше или равно

== Равенство

!= Неравенство

10 & Побитовое И (and)

11 ^ Побитовое исключающее ИЛИ (xor)

12 | Побитовое ИЛИ (or)

13 && Логическое И

14 || Логическое ИЛИ

15 ?: Тернарная условная операция Справа налево

= Присваивание

+= Сложение, совмещённое с присваиванием

-= Вычитание, совмещённое с присваиванием

\*= Умножение, совмещённое с присваиванием

/= Деление, совмещённое с присваиванием

%= Вычисление остатка от деления, совмещённое с присваиванием

<<= Побитовый сдвиг влево, совмещённый с присваиванием

>>= Побитовый сдвиг вправо, совмещённый с присваиванием

&= Побитовое «И», совмещённое с присваиванием

|= Побитовое «ИЛИ», совмещённое с присваиванием

^= Побитовое «исключающее ИЛИ» (xor), совмещённое с присваиванием

throw Оператор создания исключения (C++)

, Оператор «запятая»

Запись с использованием параллельного присваивания

a, b = b, a

«Традиционное» присвоение: требует дополнительной переменной и трёх операций

t = a

a = b

b = t

«Экономное» присвоение: не требует дополнительной переменной, но также содержит три операции

a = a + b

b = a - b

a = a - b

Еще более «экономное» присвоение: не требует дополнительной переменной, работает с битовыми операциями

a ^= b

b ^= a

a ^= b

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Язык С++.
* Типы данных в C++.
* Стандартные библиотечные функции. #include. Математические функции "math.h"

// -escape последовательности

\b -удаление последнего символа

\n -перейти на начало новой строки

\t -перейти (ТАВ)

\' одинарная кавычк

\" двойная кавычка

\? вопросительный знак

\\ обратная косая черта

\0 нулевой символ

\a звуковой сигнал

\b символ возврата на один символ назад ("забой")

\f перевод страницы -

\n перевод строки - новая строка

\r возврат каретки

\t горизонтальная табуляция

\v вертикальная табуляция

\\ обратнйы слеш

// Целочисленные типы и Типы чисел с плавающей точкой

Целочисленные типы представлены следующими типами:

**short, unsigned short, int, unsigned int, long, unsigned long, long long и unsigned long long:**

**short a = -10;**

**unsigned short b= 10;**

**int c = -30;**

**unsigned int d = 60;**

**long e = -170;**

**unsigned long f = 45;**

**long long g = 89;**

Типы чисел с плавающей точкой иили дробные числа представлены такими типами:

**float, double и long double:**

**float a = -10.45;**

**double b = 0.00105;**

**long double c = 30.890045;**

Функция

abs( a ) - модуль или абсолютное значение от |а| abs(-3.0)= 3.0 abs(5.0)= 5.0

sqrt(a) - корень квадратный из а, причём а не отрицательно sqrt(9.0)=3.0

pow(a, b) возведение а в степень b pow(2,3)=8

ceil( a ) округление а до наименьшего целого, но не меньше чем а ceil(2.3)=3.0 ceil(-2.3)=-2.0

floor(a) округление а до наибольшего целого, но не больше чем а floor(12.4)=12 floor(-2.9)=-3

fmod(a, b) вычисление остатка от a/b fmod(4.4, 7.5) = 4.4 fmod( 7.5, 4.4) = 3.1

exp(a) вычисление экспоненты (е) в степени (а) exp(0)=1

sin(a) a задаётся в радианах

cos(a) a задаётся в радианах

log(a) натуральный логарифм a(основанием является экспонента) log(1.0)=0.0

log10(a) десятичный логарифм а Log10(10)=1

asin(a) арксинус a, где -1.0 < а < 1.0 asin(1)=1.5708

\*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Арифметические выражения. Приемы, используемые для минимизации вычислений: вынесение общих множителей за скобки, использование дополнительных переменных

<https://prog-cpp.ru/c-operation/>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Ввод чисел с клавиатуры и вывод чисел в окно программы, формат ввода и вывода

cin >> cout<<

printf ()

scanf\_s ()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Средства разработки программ разветвляющейся структуры. Условные и безусловные операторы перехода
* <https://prog-cpp.ru/c-if/>
* Условный оператор if // операторы if (если) – else (иначе)
* 
* Тернарные операции

безусловный переход goto (идти к).

ветвления do \ while

выбора switch case(переключатель)

Оператор ветвления switch (оператор множественного выбора)

Оператор if позволяет осуществить выбор только между двумя вариантами. Для того, чтобы производить выбор одного из нескольких вариантов необходимо использовать вложенный оператор if. С этой же целью можно использовать оператор ветвления switch.

Общая форма записи

switch (ЦелоеВыражение)

{

case Константа1: БлокОпераций1;

break;

case Константа2: БлокОпераций2;

break;

. . .

case Константаn: БлокОперацийn;

break;

default: БлокОперацийПоУмолчанию;

break;

}

Оператор ветвления switch выполняется следующим образом:

вычисляется ЦелоеВыражение в скобках оператора switch;

полученное значение сравнивается с метками (Константами) в опциях case, сравнение производится до тех пор, пока не будет найдена метка, соответствующая вычисленному значению целочисленного выражения;

выполняется БлокОпераций соответствующей метки case;

если соответствующая метка не найдена, то выполнится БлокОперацийПоУмолчанию, описанный в опции default.

Альтернатива default может отсутствовать, тогда не будет произведено никаких действий.

Опция break; осуществляет выход из оператора switch и переход к следующему за ним оператору. При отсутствии опции break будут выполняться все операторы, начиная с помеченного данной меткой и кончая оператором в опции default.

Константы в опциях case должны быть целого типа (могут быть символами).

цикл for(int i=0; i<n; i++)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Средства разработки программ циклической структуры. Циклы с предусловием и постусловием.
* Вычисление и вывод данных в виде таблицы.
* Приёмы вычисления сумм, произведений и экстремальных значений. Вычисление суммы бесконечного ряда с заданной точностью

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Рекурсия. Использование рекурсии для вычисления числа Фибоначчи

<https://prog-cpp.ru/fibonacci/>

Числа Фибоначчи – это ряд чисел, в котором каждое последующее число равно сумме двух предыдущих:

1, 1, 2, 3, 5, 8, 13 и т. д.

То есть последовательность всегда начинается с двух единиц. А каждое следующее число является определяется по формуле:

Числа Фибоначчи

Для определения чисел Фибоначчи часто используется рекурсивный алгоритм:

Если n = 1 или n = 2, вернуть 1 (поскольку первый и второй элементы ряда Фибоначчи равны 1).

Вызвать рекурсивно функцию с аргументами n-1 и n-2.

Результат двух вызовов сложить и вернуть полученное значение.

Реализация с использованием рекурсии

#include <stdio.h>

int fibonacci(int N) // рекурсивная функция

{

if (N == 1 || N == 2)

return 1; // первые 2 числа равны 1

return fibonacci(N - 1) + fibonacci(N - 2); // складываем предыдущие 2 числа

}

int main()

{

int N;

printf("N=");

scanf("%d", &N); // вводим число N

for (int i = 1; i <= N; i++) // В цикле выводим N чисел Фибоначчи

printf("%d ", fibonacci(i));

getchar(); getchar();

return 0;

}

Реализация с использованием цикла

В этом алгоритме используется свойство, что для определения следующего числа Фибоначчи используются только два предыдущих значения.

Алгоритм при этом будет следующий

Ввести номер N определяемого элемента.

Проинициализировать два первых элемента a и b значениями 1, и если N<=2 вывести 1

Выполнять нижеследующие действия N-2 раза

Сложить a и b, присвоив результат третьей переменной c.

Поменять начальные значения: a = b, b = c

Вывести значение b.

#include <stdio.h>

int main()

{

int N;

printf("N="); // вводим число N

scanf("%d", &N);

int a = 1, b = 1, c;

if (N <= 2) // Первые два числа (a и b) равны 1

printf("1 ");

else

{

for (int i = 3; i <= N; i++)

{

c = a + b; // вычисляем следующее число как сумму двух предыдущих

a = b; b = c; // перемещаем два предыдущих числа

}

printf("%d ", b); // выводим последнее число

}

getchar(); getchar();

return 0;

}

Результат выполнения - такой же, как в предыдущем случае.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Уточнение корней уравнений: метод простых итераций, метод половинного деления, метод касательных

<https://prog-cpp.ru/digital-find/>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Организация программ со структурой вложенных циклов. Транспонирование матриц

<https://prog-cpp.ru/c-cycles/>

#include <iostream>

#include <iomanip>

//----------------

void transpose(int matrix[5][5]) //либо int matrix[][5], либо int (\*matrix)[5]

{

int t;

for(int i = 0; i < 5; ++i)

{

for(int j = i; j < 5; ++j)

{

t = matrix[i][j];

matrix[i][j] = matrix[j][i];

matrix[j][i] = t;

}

}

}

//----------------

int main()

{

int m[5][5] =

{

{1,2,3,4,5},

{6,7,8,9,10},

{11,12,13,14,15},

{16,17,18,19,20},

{21,22,23,24,25}

};

transpose(m);

// вывод

for(int i = 0; i < 5; ++i)

{

for(int j = 0; j < 5; ++j)

std::cout << std::setw(4) << m[i][j];

std::cout << '\n';

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Массивы. Сохранение результатов вычислений в массиве

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Метод сортировки массива вставками ЛР5

void insertionSort(int arr[], int lenD)

{

int key = 0;

int i = 0;

for (int j = 1; j < lenD; j++) {

key = arr[j];

i = j - 1;

while (i >= 0 && arr[i] > key) {

arr[i + 1] = arr[i];

i = i - 1;

arr[i + 1] = key;

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Метод прямого обмена (метод пузырька) и его модификация
* void bubbleSort(int arr[], int lenD)
* {
* int tmp = 0;
* for (int i = 0; i < lenD; i++) {
* for (int j = (lenD - 1); j >= (i + 1); j--)
* {
* if (arr[j] < arr[j - 1])
* {
* tmp = arr[j];
* arr[j] = arr[j - 1];
* arr[j - 1] = tmp;
* std::cout << " " << arr[j] << "\t" ;
* }
* }std::cout << "\n";
* }
* }

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Метод прямого выбора и его модификации: сортировка методом поиска минимального/максимального элемента, сортировка методом поиска индекса минимального/максимального элемента
* /////Сортировка выбором (Selection sort)
* void selectionSort(int arr[], int lenD)
* {
* int min = 0;
* int tmp = 0;
* int count;
* for (int i = 0; i < lenD; i++) {
* min = i;
* count = i;
* std::cout << " # " << ++count << " = " ;
* for (int k = i; k < lenD; k++) {
* if (arr[min] > arr[k]) {
* min = k;
* }
* }
* tmp = arr[i];
* arr[i] = arr[min];
* arr[min] = tmp;
* std::cout << " [ " << arr[min] << " ] " << std::endl;
* } std::cout << std::endl;
* }

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Матрицы.
* Обработка матриц.
* Многомерные статические массивы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Массивы указателей.
* Динамические массивы.
* Операции new, new [], delete, delete [], malloc, free

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Программирование с использованием подпрограмм. Объявление функций. Глобальные переменные. Передача параметров. Передача массивов в качестве параметров

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Рекурсивные функции, перегружаемые функции, параметры со значениями по умолчанию, функции с переменным числом параметров

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Указатели, указатели на функции, передача указателей на функции в качестве параметров

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Второй вопрос:**

* Классы.
* Конструктор и деструктор.
* Квалификаторы прав доступа.
* Указатель *this*.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Перегрузка стандартных операций.
* <https://habr.com/ru/post/132014/>
* <https://metanit.com/cpp/tutorial/5.14.php>
* Друзья классов.
* <https://ravesli.com/urok-126-druzhestvennye-funktsii-i-klassy/>
* <https://metanit.com/cpp/tutorial/5.5.php>
* Наследование классов

<https://code-live.ru/post/cpp-class-inheritance/>

<https://metanit.com/cpp/tutorial/5.10.php>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Шаблоны функций и классов

<https://code-live.ru/post/cpp-template-functions/>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Наследование классов. <https://code-live.ru/post/cpp-class-inheritance/>
* Множественное наследование. <https://ravesli.com/urok-161-mnozhestvennoe-nasledovanie/>
* Виртуальные функции и абстрактные классы.
* <http://www.c-cpp.ru/books/chisto-virtualnye-funkcii-i-abstraktnye-tipy>
* Локальные классы

<https://it.wikireading.ru/36019>

<https://studfile.net/preview/2593535/page:21/>

<https://intellect.icu/2-8-lokalnye-i-vlozhennye-klassy-5321>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Структуры и объединения.
* <https://habr.com/ru/post/334988/>
* <https://ravesli.com/urok-61-struktury/>
* Он похож на class, но есть два мелких различия:
* по умолчанию в struct действует видимость public, а в class — private по умолчанию struct наследует члены базовых структур/классов как публичные члены, а class — как приватные члены
* Структура как тип и совокупность данных.
* Объединения разнотипных данных.
* Битовые поля структур и объединений
* <http://www.c-cpp.ru/books/bitovye-polya>
* <http://www.c-cpp.ru/books/obedineniya>
* <http://www.c-cpp.ru/books/struktury>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Наследование и другие возможности классов.
* Множественное наследование
* и виртуальные базовые классы.

<https://habr.com/ru/post/445948/>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Работа с текстовыми файлам в С++. Назначение, особенности.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Функции записи и чтения текстового файла

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Режимы работы с текстовыми файлами

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Работа с бинарными файлами (Си-подобные функции). Назначение, особенности.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Функции записи и чтения бинарного файла

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Режимы работы с бинарными файлами

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Обработка исключений в С++. Операторы try, catch, throw.
* <http://www.c-cpp.ru/books/obrabotka-isklyucheniy>
* <https://metanit.com/cpp/tutorial/6.1.php>

try {

// блок try

catch (тип1 аргумент) {

// блок catch

catch (тип2 аргумент) {

// блок catch

catch (типЗ аргумент) {

// блок catch

}

...

catch (типN аргумент) {

// блок catch

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Однонаправленный список.
* Структурная диаграмма.
* Функция добавления элемента.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_+\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Двунаправленный список. <https://prog-cpp.ru/data-dls/>
* <https://purecodecpp.com/archives/2940> (++\_)
* Структурная диаграмма.
* -
* Функция просмотра списка. <https://ci-plus-plus-snachala.ru/?p=60>
* <http://www.codenet.ru/progr/cpp/1/h21.php>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Классы. <https://code-live.ru/post/cpp-classes/>
* Конструктор и деструктор. <https://code-live.ru/post/cpp-continue-learning-classes/>
* Квалификаторы прав доступа. public(open); private(только класс и его наследники+френд); protected(только внутри класса);
* Указатель *this*.
* **Указатель \*this** является скрытым параметром, который неявно добавляется к каждому методу класса. В большинстве случаев нам не нужно обращаться к нему напрямую, но, при необходимости, это можно сделать. Стоит отметить, что this является константным указателем — вы можете изменить значение исходного объекта, но вы не можете указать this указывать на что-то другое!
* Если у вас есть функции, которые возвращают void, то возвращайте \*this вместо void. Таким образом, вы сможете соединить несколько методов в одну «цепочку». Это чаще всего используется при перегрузке операторов.

Основные понятия

**Классы** в программировании состоят из свойств и методов. Свойства — это любые данные, которыми можно характеризовать объект класса. В нашем случае, объектом класса является студент, а его свойствами — имя, фамилия, оценки и средний балл.

У каждого студента есть имя — name и фамилия last\_name . Также, у него есть промежуточные оценки за весь семестр. Эти оценки мы будем записывать в целочисленный массив из пяти элементов. После того, как все пять оценок будут проставлены, определим средний балл успеваемости студента за весь семестр — свойство average\_ball.

Методы — это функции, которые могут выполнять какие-либо действия над данными (свойствами) класса. Добавим в наш класс функцию calculate\_average\_ball(), которая будет определять средний балл успеваемости ученика.

Методы класса — это его функции.

Свойства класса — его переменные.

**Конструктор класса** — это специальная функция, которая автоматически вызывается сразу после создания объекта этого класса. Он не имеет типа возвращаемого значения и должен называться также, как класс, в котором он находится. По умолчанию, заполним двойками массив с промежуточными оценками студента.

**Деструктор** класса вызывается при уничтожении объекта. Имя деструктора аналогично имени конструктора, только в начале ставится знак тильды ~. Деструктор не имеет входных параметров.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Работа с текстовыми файлам в С++. Назначение, особенности.
* *Файлом называют способ хранения информации на физическом устройстве. Файл — это понятие, которое применимо ко всему — от файла на диске до терминала.*
* *В C++ отсутствуют операторы для работы с файлами. Все необходимые действия выполняются с помощью функций, включенных в стандартную библиотеку. Они позволяют работать с различными устройствами, такими, как диски, принтер, коммуникационные каналы и т.д. Эти устройства сильно отличаются друг от друга. Однако файловая система преобразует их в единое абстрактное логическое устройство, называемое потоком.*
* *Текстовый поток — это последовательность символов. При передаче символов из потока на экран, часть из них не выводится (например, символ возврата каретки, перевода строки).*
* *Двоичный поток — это последовательность байтов, которые однозначно соответствуют тому, что находится на внешнем устройстве.*
* си <http://comp-science.narod.ru/Progr/file_c.htm>
* с++ <https://ravesli.com/urok-207-potoki-vvoda-vyvoda/>
* 
* Есть ещё несколько полезных функций класса istream, которые вы можете использовать:
* ignore() — игнорирует первый символ из потока.
* ignore(int nCount) — игнорирует первые nCount (количество) символов из потока.
* peek() — считывает символ из потока, при этом не удаляя его из потока.
* unget() — возвращает последний считанный символ обратно в поток, чтобы его можно было извлечь в следующий раз.
* putback(char ch) — помещает выбранный вами символ обратно в поток, чтобы его можно было извлечь в следующий раз.

//fstream - с файлом, который будет использоваться для операций ввода - вывода;

//ifstream - файловый ввод; inPutFile

//ofstream - файловый вывод.outPutfile

// ofstream fout;

// fout.open("file.txt");

// fout << "Привет, мир!";

// fout.close();

<https://ravesli.com/urok-209-funktsional-klassov-ostream-i-ios-formatirovanie-vuvoda/>

состояние потока

<https://ravesli.com/urok-211-sostoyaniya-potoka-i-validatsiya-polzovatelskogo-vvoda/>

А что, если нам нужно добавить данные в конец файла? Оказывается, конструкторы файлового потока принимают необязательный второй параметр, который позволяет указать программисту способ открытия файла. В качестве этого параметра можно передавать следующие флаги (которые находятся в классе ios):

app — открывает файл в режиме добавления;

ate — переходит в конец файла перед чтением/записью;

binary — открывает файл в бинарном режиме (вместо текстового режима);

in — открывает файл в режиме чтения (по умолчанию для ifstream);

out — открывает файл в режиме записи (по умолчанию для ofstream);

trunc — удаляет файл, если он уже существует.

Можно указать сразу несколько флагов путём использования побитового ИЛИ (|).

ifstream по умолчанию работает в режиме ios::in;

ofstream по умолчанию работает в режиме ios::out;

fstream по умолчанию работает в режиме ios::in ИЛИ ios::out, что означает, что вы можете выполнять как чтение содержимого файла, так и запись данных в файл.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Шаблоны функций и классов

<https://ravesli.com/urok-173-shablony-funktsij/>

template <typename T> // объявление параметра шаблона функции

T max(T a, T b)

{

return (a > b) ? a : b;

}

--------------------------------------------------

#include <iostream>

template <typename T>

const T& max(const T& a, const T& b)

{

return (a > b) ? a : b;

}

int main()

{

int i = max(4, 8);

std::cout << i << '\n';

double d = max(7.56, 21.434);

std::cout << d << '\n';

char ch = max('b', '9');

std::cout << ch << '\n';

return 0;

}

У шаблонов функций есть несколько недостатков, и было бы непростительно, если бы мы о них не поговорили:

Во-первых, некоторые старые компиляторы могут не поддерживать шаблоны функций или поддерживать, но с ограничениями. Однако сейчас это уже не такая проблема, как раньше.

Во-вторых, шаблоны функций часто выдают сумасшедшие сообщения об ошибках, которые намного сложнее расшифровать, чем ошибки обычных функций (мы рассмотрим это в следующем уроке).

В-третьих, шаблоны функций могут увеличить время компиляции и размер кода, так как один шаблон может быть «реализован» и перекомпилирован в нескольких файлах (мы рассмотрим это детальнее в следующих уроках).

Данные недостатки довольно незначительны по сравнению с мощью и гибкостью шаблонов функций!

Примечание: Стандартная библиотека C++ имеет в своём арсенале шаблон функции max() (который находится в заголовочном файле algorithm), поэтому вы можете не реализовывать эту функцию вручную в будущем. Плюс, если вы пишете свои собственные шаблоны функций и используете стейтмент using namespace std;, то не забывайте о возможности возникновения конфликтов имён, так как компилятор не сможет определить, хотите ли вы использовать свою версию функции max() или версию std::max().

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Первая задача - Массивы:**

* В массиве K с индексами от 0 до 9, заданном начальными значениями, содержатся разные числа от 0 до 9 в произвольном порядке.
* Требуется поместить в массив Y зашифрованную произвольную последовательность X длины L≤30 из целых чисел от 0 до 9.
* Шифрование выполнить по следующему правилу: Yi=i-Kxi.
* Затем по данным из Y расшифровать последовательность и поместить в массив P.
* Использовать дополнительный массив T с начальными значениями, заданными следующим образом: Ti равно номеру ячейки массива K со значением i.
* Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main().
* Выделение памяти под массив производится динамически

#include <stdexcept>

#include <iostream>

#include <algorithm> // подключаем STL для shuffle

using namespace std;

#define endk (cout << endl)

#define endh (cout << "--------------------------------" << endl)

void arrK(int \*arrK, const int size);

void arrKRandom(int \*arrK, const int size);

void arrT(int \*arrK, int \*arrT, int size);

void arrXY(int \*arrX, int \*arrY, int size, int \*arrK);

void arrP(int \*arrP, int sizeL, int \* arrT, int \*arrY);

int main()

{endh;

const int size = 10;

const int sizeL = 30;

int \*K = new int[size];

int \*T = new int[size];

int \*Y = new int[sizeL];

int \*X = new int[sizeL];

int \*P = new int[sizeL];

arrK(K, size); // проинициализировали массив K

arrKRandom(K, size); //перемешали массив K

arrT(K,T, size); // проинициализировали массив T

arrXY(X, Y, sizeL, K); //создаем X рандом и шифуем по формуле Y[i] = i - K[X[i]]

arrP(P, sizeL, T, Y);// дешифруем и выдаем оригинал

delete[] K;

delete[] T;

delete[] X;

delete[] Y;

delete[] P;

// cin.get();

return 0;

}

void arrK(int \*arrK, const int size)

{

cout << "entered K elements: " << endl;

int numArr=0;

for (int i = 0; i < size; i++)// заполняем массив К

{

cin >> numArr;

//arrK[size] = numArr;

arrK[i] = numArr;

}

for (int i = 0; i < size; i++)// заполняем массив К

{

cout << arrK[i] << "\t";

}

endk;

endh;

}

void arrKRandom(int \*arrK, const int size)

{

random\_shuffle(arrK, arrK + size); // перемешиваем элементы массива K

cout << "K random: " << endl;

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

{

cout << arrK[i] << "\t";

}

endk;

endh;

}

void arrT(int \*arrK, int \*arrT, int size)

{

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

arrT[arrK[i]] = i;

}

cout << "T elements: " << endl;

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

cout << arrT[i] << '\t';

}

endk;

endh;

}

void arrXY(int \*arrX, int \*arrY, int size, int \*arrK)

{

cout << "X elements : " << endl;

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

{

arrX[i] = rand() % 10;

cout << arrX[i] << "\t";

}

endk;

endh;

cout << "Y elements to Y[i] = i - K[X[i]]: " << endl;

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

{

arrY[i] = i - arrK[arrX[i]];

cout << arrY[i] << "\t";

}

endk;

endh;

}

void arrP(int \*arrP, int sizeL, int \*arrT, int \*arrY)

{

cout << "P: deshifrate " << endl;

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

{

arrP[i] = arrT[i - arrY[i]];

cout << arrP[i] << "\t";

}

endk;

endh;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу, которая в матрице A(m,n), m≤10, n≤12, меняет местами строку, содержащую максимальный элемент со строкой, содержащей минимальный элемент. Предполагается, что искомые элементы единственные. Вывести исходную и преобразованную матрицы, минимальный и максимальный элементы, а также номера строк, в которых они расположены. Если минимальный и максимальный элементы расположены в одной строке, то поменять местами столбцы, содержащие эти элементы. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под матрицу производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу, которая в каждой строке матрицы D(m,n), m≤10, n≤12, находит элемент, для которого модуль разности этого элемента и среднего арифметического элементов строки минимален. Вывести исходную матрицу так, чтобы после элементов строки матрицы располагались найденный элемент, среднее арифметическое и модуль их разности. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под матрицу производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу, которая в каждой строке матрицы G(m,n), m≤10, n≤12 находит сумму элементов, расположенных до максимального элемента и после максимального элемента. Если сумма не может быть вычислена (нет элементов до или после максимального элемента), то считать ее равной нулю. Вывести исходную матрицу, располагая в одной строке элементы строки матрицы, после которых вывести номер столбца максимального элемента и найденные суммы. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под матрицу производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу, которая в каждой строке матрицы D(m,n), m≤10, n≤12, находит элемент, для которого модуль разности этого элемента и среднего геометрического модулей всех элементов строки максимален. Предполагается, что матрица нулевых элементов не содержит. Вывести матрицу в виде матрицы, располагая рядом с каждой строкой найденный элемент и модуль искомой разности. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под матрицу производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Выполнить циклический сдвиг элементов массива Х(N), N≤10, в результате которого значение последнего элемента должно оказаться на месте первого, а остальные – сдвинутыми на одну позицию в сторону увеличения индекса. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Из массива Х, упорядоченного по невозрастанию значений элементов, переписать в массив Y числа, исключив их повторы и обеспечив упорядоченность по возрастанию. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Поместить элементы массива X в начало массива Y в обратном порядке, исключив элементы, превосходящие по абсолютной величине вводимое значение R. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Из массива Х(20), упорядоченного по неубыванию значений элементов, переписать в массив Y числа, исключив их повторы и добавив новое вводимое значение P так, чтобы не нарушить упорядоченность. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Поместить положительные элементы массива X в начало массива Y, а следом – его отрицательные элементы. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Выполнить циклический сдвиг элементов массива Х(N), N≤20, на K позиций, в результате которого последние K элементов займут место в начале массива, а остальные будут сдвинуты на K позиций в сторону увеличения индекса. Использовать дополнительный массив D. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по убыванию первых N, N ≤12, элементов массива X методом нахождения максимального элемента (или индекса максимального элемента). Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по убыванию первых N, N ≤12, элементов массива X методом нахождения минимального элемента (или индекса минимального элемента). Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по возрастанию первых N, N≤12, элементов массива X методом сортировки прямого обмена (методом пузырька). Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Последовательность S из нулей и единиц длиной L<30 зашифровать и поместить в массив D. Шифровать по следующему правилу: положить D1=S1, а далее Di=1, если Si=Si-1, иначе – 0. Затем по данным из D расшифровать последовательность и поместить в массив R. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по возрастанию первых N, N≤12, элементов массива X методом сортировки вставками. Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по возрастанию первых N, N≤12, элементов массива X методом нахождения минимального элемента (или индекса минимального элемента). Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Составить программу упорядочения по возрастанию первых N, N≤12, элементов массива X методом нахождения максимального элемента (или индекса максимального элемента). Следует реализовать ввод исходного массива, вывод требуемого массива и подпрограмму реализации работы с массивом в отдельных функциях, которые должны вызываться в функции main(). Выделение памяти под массив производится динамически

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Вторая задача - Текстовые файлы**:

1. Сформировать текстовый файл, в который записать фамилии людей и их заработную плату за месяц. Вывести на экран фамилию человека с самым высоким доходом.

#include <iostream>

#include <fstream>

#include <string>

#include <windows.h>

#include <io.h>

using namespace std;

//fstream - с файлом, который будет использоваться для операций ввода - вывода;

//ifstream - файловый ввод; inPutFile

//ofstream - файловый вывод.outPutfile

// ofstream fout;

// fout.open("file.txt");

// fout << "Привет, мир!";

// fout.close();

void AddFile();

void readFile();

class sportMANs

{

private:

public:

string name;

int yers;

int result;

//static int countMAN;

sportMANs()

{

name = "empty";

yers = result = 0;

//countMAN = 0;

}

sportMANs(string name, int yers, int result) {

this->name = name;

this->yers = yers;

this->result = result;

// //countMAN++;

}

void SetsportMANs(sportMANs arr[], int length)

{

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

{

cout << "name: "; cin >> arr[i].name; cout << endl;

cout << "yers: "; cin >> arr[i].yers; cout << endl;

cout << "result: "; cin >> arr[i].result; cout << endl;

}

};

void print(sportMANs arr[],int length)

{

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

{

cout << "name: " << arr[i].name << endl;

cout << "yers: " << arr[i].yers << endl;

cout << "result: " << arr[i].result << endl;

cout << endl;

}

}

friend ostream& operator<<(ostream& os, const sportMANs& point);

friend istream& operator>>(istream& is, sportMANs& point);

friend void AddFile();

friend void readFile();

};

ostream& operator<<(ostream& os, const sportMANs& point)

{

os << point.name << " " << point.yers << " " << point.result<<"\n";

return os;

}

istream& operator>>(istream& is, sportMANs& point)

{

is >> point.name >> point.yers >> point.result;

return is;

}

void AddFile() {

string path = "file.txt";// cin>>path;//создаем новое название файла если надо:

int length;//число(для массива Класса)

cout << "введите число участиков : "; cin >> length;

cout << endl;

int \*arrMan = new int[length];

sportMANs \*addNewMans = new sportMANs[length];

addNewMans->SetsportMANs(addNewMans, length);

fstream fs;

fs.open(path, fstream::in | fstream::out | fstream::app);

if (!fs.is\_open())

{

cout << "~incorect file name~!" << endl;

}

else

{

cout << "~file\_is\_open~!" << endl;

for (int j = 0; j < length; j++)

{

SetConsoleCP(1251);

//fs << "-----------------------" << endl;

fs << addNewMans[j].name << " " << addNewMans[j].yers << " " <<addNewMans[j].result<< "\n";

SetConsoleCP(866);

}

cout << "value saved!" << endl;

cout << endl;

}

fs.close();

delete[] addNewMans;

}

void readFile()

{

fstream fs1;

string path = "file.txt";

cout<<"choise file "<<endl;

//cin >> path;//создаем новое название файла если надо

string name = "";

int years, result, length, sizeM;

years = result = 0;

length = 5000;

char \*str = new char[length];

sizeM = 0;

ifstream base(path);

while (true)

{

base.getline(str, length, '\n');

if (base.eof()) { break; }

sizeM++;

}

base.close();

delete[] str;

sportMANs \*oldMans = new sportMANs[sizeM];

fs1.open(path, fstream::in | fstream::out | fstream::app);

if (!fs1.is\_open())

{

cout << "~file not faund~!" << endl;

}

else

{

cout << "file opened!" << endl;

while (true)

{

int max, index, xxx=0;

max = index = 0;

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

{

SetConsoleCP(1251);

fs1 >> name >> years >> result;

if (fs1.eof())

{

break;

}

oldMans[i].name = name;

oldMans[i].yers = years;

oldMans[i].result = result;

if (oldMans[i].result > max) {

max = oldMans[i].result;

index = i;

}

SetConsoleCP(866);

}

cout << "----------------------- " << endl;

cout << "max num is a player " << endl;

cout << oldMans[index].name << endl;

cout << oldMans[index].yers << endl;

cout << oldMans[index].result << endl;

cout << "----------------------- " << endl;

if (fs1.eof())

{

break;

}

}fs1.close();

}

oldMans->print(oldMans, sizeM);

delete[] oldMans;

};

//void maxElement(int arr[], int length) {

//

//};

void welcom() {

int switch\_on, rerollo = true;

cout << "choise:" << endl;

cout << "one(1):new file or add write" << endl;

cout << "one(2):read file" << endl;

cout << "zero(0):to exit" << endl;

cin >> switch\_on;

while (rerollo)

{

switch (switch\_on)

{

case 1:AddFile(); rerollo = false; break;

case 2:readFile(); rerollo = false; break;

case 0: cout << "zero = goodbye" << endl; rerollo = false; break;

default:cout << "incorrect Num" << endl;break;

}

}

}

int main()

{ setlocale(LC\_ALL, "rus");

welcom();

//---------------записываем--file--------------//

//AddFile();

//--------------считваем--file----------------//

//readFile();

system("pause");

return 0;

}

2. Сформировать текстовый файл, в котором для группы спортсменов указаны сведения о фамилии и результате в беге на дистанцию 5000м (минуты, секунды). Вывести на экран информацию о лучшем спортсмене.

так же как и 3-1

3. Сформировать текстовый файл, в который занести в каждую строку информацию о названии журнала и цену издания. Найти самый дорогой журнал и вывести на экран его название.

как и 3-1

4. Сформировать текстовый файл с информацией о названии городов и численности их жителей. Найти самый крупный город и вывести на экран его название.

3-1

5. Сформировать текстовый файл с названииями городов и информацией о средней температуре за год. Вывести на экран название самого теплого города.

3-1

6. Сформировать текстовый файл с информацией о фамилии абитуриента, двух оценках (по математике и физике) и признаке (зачет/незачет по русскому языку). Для вводимого проходного балла сформировать список абитуриентов, зачисленных на обучение, и вывести его на экран.

7. Сформировать текстовый файл с информацией о названии города, средней ночной и средней дневной температурах для апреля. Вывести на экран самый холодный город.

8. Сформировать текстовый файл с информацией о фамилии фигуриста и его оценках, полученных у трех судей. Вывести на экран фамилию лучшей.

9. Сформировать текстовый файл, записав в него фамилии студентов, год их рождения, пол. Вывести на экран всех мужчин моложе 20 лет.

10. Сформировать текстовый файл, содержащий информацию о марке компьютера, его тактовой частоте, объеме оперативной и внешней памяти. Вывести на экран сведения о компьютерах, удовлетворяющих заданным требованиям (вводятся с клавиатуры).

11. Сформировать текстовый файл с информацией об автомобилях: марке, максимальной скорости, количестве пассажиров. Вывести на экран информацию о марках автомобилей, чья максимальная скорость выше 150 км/ч.

3-1

12. Сформировать текстовый файл с информацией о фамилии студента, и его оценках (всего три). Вывести на экран всех студентов без двоек и троек.

13. Сформировать текстовый файл с информацией об индексе учебной группы, количестве студентов в ней и количестве студентов, успешно сдавших сессию. Вывести на экран все группы, чей процент успеваемости выше заданной величины.

15. Сформировать текстовый файл, в который записать фамилии студентов и баллы, набранные при выполнении четырех заданий. Вывести на экран фамилии студентов, которые преодолели заданный порог по сумме полученных баллов (порог вводится с клавиатуры).

16. Сформировать текстовый файл, в который записать фамилии студентов и баллы, набранные при выполнении трех заданий. Вывести на экран фамилию лучшего студента по сумме набранных баллов.

18. Сформировать текстовый файл, в котором для группы спортсменов указаны сведения о фамилии и результате в беге на дистанцию 5000м (минуты, секунды). Вывести на экран информацию о худщем спортсмене.

так же как 3-1 только через минЭлемент