

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«Воронежский государственный технический университет»

Кафедра систем автоматизированного проектирования и информационных систем

**МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ
ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ «ОБЪЕКТНО-
ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»**

*для обучающихся по направлению 09.03.02 «Информационные системы и
технологии» всех форм обучения*

Воронеж 2021

УДК 681.3

Составители: Б.Н. Тишуков, Д.В. Иванов

Методические рекомендации по выполнению лабораторных работ по дисциплине «Объектно-ориентированное программирование» для обучающихся по направлению 09.03.02 «Информационные системы и технологии» всех форм обучения / ФГБОУ ВО «Воронежский государственный технический университет»; сост.: Б.Н. Тишуков, Д.В. Иванов. – Воронеж: Изд-во ВГТУ, 2021. – 49 с.

Приводится задание и описание выполнения лабораторных работ по курсу «Объектно-ориентированное программирование» для студентов, обучающихся по направлению 09.03.02 «Информационные системы и технологии» всех форм обучения

Рецензент – к.т.н., доцент Е.Н. Королев

Рекомендовано методическим семинаром кафедры САПРИС и методической комиссией ФИТКБ Воронежского государственного технического университета в качестве методических материалов

ЛАБОРАТОРНАЯ РАБОТА № 1

«Базовые конструкции языка C++. Управляющие конструкции»

Задание 1

Найти значение алгебраического выражения, соответствующего варианту задания. Вывести результаты на печать. Все результаты выводить в развернутом виде (например: «Сумма чисел А и В равна 3.7854»).

Значения вводимых величин должны иметь не менее четырех значащих цифр и задаются студентом самостоятельно.

Варианты

Через а и b обозначены подлежащие вводу числа с плавающей точкой.

$$1) \sin^2(a+b^3) \sqrt{\frac{e^{a^2-9,4}}{(a+b)^3}}$$

$$2) \ln \left(\frac{\sin a^2 + \cos b}{\sqrt{1 + \frac{e}{a^3 + 3,4b}}} \right)$$

$$3) \frac{\operatorname{arctg} \left(\frac{\sin(a+\pi)}{\cos(b+2,87)} \right)}{\sqrt{a + \cos^2 b}}$$

$$4) \operatorname{arctg} \left(\frac{\sin a^2 + b^3}{\sqrt{1 + \frac{b}{a + \cos(\pi \cdot b^2)}}} \right)$$

$$5) \sqrt{\frac{\sin(\pi + b^2)}{a^2 (\sin a + b^2 \cos 7,2)}}$$

$$6) \frac{17,8a + \cos^2(13,2b^2 + 2,4b + 3,7)}{\sqrt{e + \frac{13,7 - \sin(a^2 + b)}{-17,478a + 13,2b}}}$$

$$7) \cos^3(a+b^2) \sqrt{\frac{e^{a^2-13,4b}}{(a+b)^3}}$$

$$8) \ln \left(\frac{\cos(a^2 + 8,72) + \sin^2(\pi \cdot b)}{\sqrt{1,2 + \frac{a}{a^3 + 3,47b}}} \right)$$

$$9) \operatorname{arctg} \left(\frac{\cos(2\pi \cdot b^2) + a^3}{\sqrt{1,87 + \frac{a^2}{b + \cos(3,42b^2)}}} \right)$$

$$10) \sqrt{\frac{\cos(\pi \cdot a) - 1,2b^2}{b^2 (\cos a^2 + b^2 \cos(13,4b))}}$$

$$11) \sqrt{\frac{\frac{a^2+e}{e^{a^3+b}} (1,7a^2 + b^3)}{\sin(b + a^2) + \cos(1,92b/a)}}$$

$$12) \cos a^2 \frac{(a+b)^3 - \cos^2(a^2 + b)}{\sqrt{\sin^2 b + \cos^2(\pi - a)^3}}$$

$$13) \cos^2(a^3 - \pi) \sqrt{\frac{e^{-b^2+4,9}}{(a-b)^3}}$$

$$14) e^{\frac{\cos b^2 + \sin a}{\sqrt{(a+b)^3}}} \operatorname{tg} \left(\frac{\sin(\pi + b^2)}{\cos(b-a)^2} \right)$$

$$15) \operatorname{ctg} \left(\frac{\sqrt{e + \frac{a+4,8}{b-a^2}}}{\sqrt{a + \cos^2(a+b)}} \right) \cdot e^{\frac{\sin a}{\cos b}}$$

$$19) \ln \left(\frac{\cos(a^2 + b) + \sin^2(7,6b)}{\sqrt{2,2e + \frac{b}{a - \cos^2(b^2 + a^2)}}} \right)$$

$$16) \sqrt{\frac{\cos(a - b^2 + 2ab)}{b^2(\cos(b - a)^2 + \pi^2 \sin 7,4)}} \times \\ \times \ln \frac{\cos(a - b)^3}{\sin(a + b)^3}$$

$$20) \operatorname{arctg} \left(\frac{\sin(2,48a^2) - b^3}{\sqrt{1,8 + \frac{\pi}{a + \cos(3,45b^2)}}} \right)$$

$$17) \frac{7,8b + \sin^2(8,5a^2 - 2ab + 4,7b)}{\sqrt{2,6 + \frac{1,3\pi - \cos(b^2 - a)}{18,87b + 7,85a}}}$$

$$18) \cos^3(b^2 - a^2) \sqrt{\frac{e^{(-a^2 + 2ab - b^2)}}{(\pi - b) + (a + \pi)^3}}$$

Задание 2

Составить программу вычисления функции $F(x)$, разложенной в ряд, сходящийся в заданной области.

Вычисления проводить до тех пор, пока модуль разности между последующим и предыдущим членами ряда не будет меньше или равен ε . Значение ε задается константой и для всех вариантов составляет 0,001.

Решить задачу для различных значений x . При решении задач массивы не использовать. Специальные функции для возведения в степень не использовать.

Варианты

№ п/п	Функция $F(x)$	Область сходимости	x
1	$1 - x + x^2 - \dots + (-1)^{n-1} x^{n-1} + \dots$	$ x < 1$	$x_1 = 0,07$ $x_2 = 0,95$ $x_3 = -0,5$
2	$1 - 2x + 3x^2 - \dots + (-1)^{n-1} nx^{n-1} + \dots$	$ x < 1$	$x_1 = 0,1$ $x_2 = 0,77$ $x_3 = -0,9$

№ п/п	Функция F(x)	Область сходимости	x
3	$x - \frac{x^2}{2} + \frac{x^3}{3} - \dots + (-1)^{n-1} \frac{x^n}{n} + \dots$	$-1 < x \leq 1$	$x_1 = 0,1$ $x_2 = 0,8$ $x_3 = -0,62$
4	$x - \frac{x^3}{3} + \frac{x^5}{5} - \dots + (-1)^{n-1} \frac{x^{2n-1}}{2n-1} + \dots$	$ x \leq 1$	$x_1 = 1$ $x_2 = 0,005$ $x_3 = -0,65$
5	$2x + \frac{(2x)^2}{2!} + \dots + \frac{(2x)^n}{n!} + \dots$	$ x < \infty$	$x_1 = 1$ $x_2 = 17,5$ $x_3 = -0,35$
6	$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots + (-1)^{n-1} \frac{x^{2n-1}}{(2n-1)!} + \dots$	$ x < \infty$	$x_1 = -2,2$ $x_2 = 0,75$ $x_3 = 5,6$
7	$1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots + (-1)^{n-1} \frac{x^{2n}}{(2n)!} + \dots$	$ x < \infty$	$x_1 = -2$ $x_2 = 0,98$ $x_3 = 15$
8	$\frac{\pi}{2} - \left[\frac{1}{x} - \frac{1}{3x^3} + \dots + (-1)^{n-1} \frac{1}{(2n-1)x^{2n-1}} + \dots \right]$	$ x > 1$	$x_1 = 10$ $x_2 = 1,5$ $x_3 = -5$
9	$x \left[\frac{x^2}{1!3} - \frac{x^4}{2!5} + \dots + (-1)^{n-1} \frac{x^{2n}}{n!(2n+1)} + \dots \right]$	$ x < \infty$	$x_1 = 1$ $x_2 = 7,5$ $x_3 = -2,1$
10	$\cos x + \frac{1}{2} \cos 2x + \dots + \frac{1}{n} \cos nx + \dots$	$0 < x \leq \pi$	$x_1 = 1$ $x_2 = \frac{\pi}{5}$ $x_3 = 2,5$
11	$\cos x + \frac{1}{3} \cos 3x + \dots + \frac{1}{2n-1} \cos(2n-1)x + \dots$	$0 < x \leq \pi$	$x_1 = \frac{\pi}{4}$ $x_2 = 1$ $x_3 = \frac{\pi}{2}$

12	$\frac{\sin x}{2} + \frac{\sin 2x}{2^2} + \dots + \frac{\sin nx}{2^n} + \dots$	$ x < \infty$	$x_1 = 1$ $x_2 = 3\pi$ $x_3 = -4,1$
13	$2 \left[x + \frac{x^3}{3} + \dots + \frac{x^{2n-1}}{2n-1} + \dots \right]$	$ x < 1$	$x_1 = 0,1$ $x_2 = -0,73$ $x_3 = 0,81$
14	$\frac{18}{\pi} \left(\sin x + \dots + \frac{1}{(2n-1)^3} \sin(2n-1)x + \dots \right)$	$0 < x \leq 2\pi$	$x_1 = 5$ $x_2 = \frac{2\pi}{3}$ $x_3 = \frac{\pi}{6}$
15	$2 \left[\frac{1}{x} + \frac{1}{3x^3} + \dots + \frac{1}{(2n-1)x^{2n-1}} + \dots \right]$	$ x > 1$	$x_1 = 2$ $x_2 = -3,15$ $x_3 = 1,45$
16	$\frac{4a}{\pi} \left(\sin x + \frac{\sin 3x}{3} + \dots + \frac{\sin(2n-1)x}{2n-1} + \dots \right)$	$0 < x \leq \pi$	$x_1 = \frac{\pi}{3}$ $x_2 = 2$ $x_3 = \frac{\pi}{4}$ $a = 12$
17	$\cos x - \frac{1}{2} \cos 2x + \dots + (-1)^{n-1} \frac{1}{n} \cos nx + \dots$	$0 \leq x < \pi$	$x_1 = 0,5$ $x_2 = \frac{\pi}{3}$ $x_3 = 1$
18	$\frac{\pi^2}{3} - 4 \left(\cos x - \frac{\cos 2x}{2^2} + \dots + (-1)^{n-1} \frac{\cos nx}{n^2} + \dots \right)$	$ x \leq \pi$	$x_1 = 0,1$ $x_2 = -3$ $x_3 = 1,5$
19	$\frac{4}{\pi} \left(\sin x + \frac{\sin 3x}{3^2} + \dots + (-1)^{n-1} \frac{\sin(2n-1)x}{(2n-1)^2} + \dots \right)$	$ x \leq \frac{\pi}{2}$	$x_1 = 0,15$ $x_2 = 1,25$ $x_3 = -0,85$
20	$(x-1) - \frac{(x-1)^2}{2} + \dots + (-1)^{n-1} \frac{(x-1)^n}{n} + \dots$	$0 < x \leq 2$	$x_1 = 1$ $x_2 = 0,13$ $x_3 = 1,85$

Задание 3

В задании всего 2 варианта: вариант 1 выполняют те, у кого основной вариант нечетный, вариант 2 выполняют те, у кого основной вариант четный. Каждый вариант содержит 15 задач, содержащих те или иные ошибки (логические, ошибки синтаксиса и т.п.). Задача состоит в следующем: используя

средства отладчика среды, обнаружить все ошибки и сделать программу работоспособной (т.е. выполняющей то, что от нее требуется).

Для отчета по данному заданию необходимо:

- показать работающие программы;
- код программ должен быть структурированным;
- к каждой программе необходимо составить список обнаруженных ошибок (согласно сообщениям отладчика), а также пояснения того, как вы исправляли эти ошибки.

Вариант 1	Вариант 2
Задача 1. Дано натуральное число n. Получить все его натуральные делители.	
<pre> Main () { Int n, j; Cout>>'n'>>endl; Cin<<n; While j<n { if (n/i = 0) and (n<>j) cout>>j>>endl; } </pre>	<pre> #include <iostream> main() for (i=1, n div 2, i++); if n % i =0 cout<<i } </pre>
Задача 2. Дано 100 вещественных чисел. Вычислить разность между максимальным и минимальным из них. <pre> Main () {float a, min, max, res; i:int; cout<<'n=, a1= , a2= ' <<endl; cin>>n,a1,a2; if (a1>a2) {a1=max; a2=min }; else a2=max; a1=min; } i=3; while i<100 {cout<<'a', I, '=' <<endl; cin>>a; if (a<min) min=a; if (a>max) max=a; i++; res=max-min; cout<<res; } </pre>	Задача 2. Вычислить величину у, равную (n!!) $Y = \begin{cases} 1 \cdot 2 \cdot 3 \dots n, & \text{если } n - \text{нечетное} \\ 2 \cdot 4 \cdot 6 \dots n, & \text{если } n - \text{четное} \end{cases}$ <pre> main() {int I, fn, n, y; cout<<n; fn=1; y=1; if (n % 2 <> 0) {for (i=1; n;) { fn*=i; } } for (i=1, fn;) {y+=fn; } } else { for (i=2; n;) if I % 2 == 0 fn*=I else fn=fn*(i+1); } for (i=1; fn) { y+=fn } cout<<y; } </pre>

Задача 3. Вычислить величину y, равную $(n!!)$ $Y = \begin{cases} 1 \cdot 3 \cdot 5 \dots n, & \text{если } n - \text{нечетное} \\ 2 \cdot 4 \cdot 6 \dots n, & \text{если } n - \text{четное} \end{cases}$ <pre> if n%1 <> 0 for (i=1; n; i++;) y*=y*I; else for (i=1; n; i+=2;) y:=y*I; cout<<y<<endl; </pre>	Задача 3. Дано натуральное число n. Вычислить произведение первых n сомножителей: $\frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \dots$ <pre> (For (i=2; n, i+=2) { P=p*(i/(i+1)); } For (i=1; n; i+=2) {P=p*((i+1)/i); } cout<<p; </pre>
Задача 4. Вычислить $\sum_{k=1}^n \sum_{m=k}^n \frac{x+k}{m}$.	
<pre> Main() { int k, x, m, n; float s, rez=rezs=0; cout<<('n:', x:'); cin>>(n,x); for (k=1; n;) for (m=k; n;) { s:=(x+k)/m; rez+=s; } rezs:=rezs+rez; } cout<<rezs:8:3<<\n; } </pre>	<pre> int n, j; integer; { cout<<'n'; cin>>n; cout<<'x'; cin>>x; r=res=0; for (k=1; n) for (m=k, n) {s=(x+k)/m; r=r+s; } } res+=r; } cout<<res; } </pre>
Задача 5. Вычислить $\sum_{i=1}^n \sum_{j=1}^i \frac{1}{i+2j}$.	
<pre> Main() { int I, j, n, s=0; cin>>n; for (i=1; n) for (j=1; I) s:=1/(i+2*j); cout<<s; </pre>	<pre> { int i, j, n; float sum1, sum2; cout<<'n: '; cin>>n; sum1=0; sum2=0; for (i=1; n) for (j=1; i) sum1=sum1+(1/(i+2*j)); } sum2= sum1 + sum2; } cout<<sum2:8:3; } </pre>
Задача 6. Вычислить $\sum_{k=1}^{10} \frac{\sum_{n=1}^k \sin kn}{k!}$. <pre> fac=1; s=0; for (k=1; 10) { s=0; for (n=1, k) s=s+sin(1+k); fac*=k; zn=s/fac; s=s+zn;} </pre>	Задача 6. Дано 200 вещественных чисел. Определить, сколько из них больше своих «соседей», т.е. предыдущего и последующего чисел. <pre> { cout<<'a1='; cin>>b; cout<<'a2='; cin>>c n=0; for (i=1; 200) { cout<<'a', I, '='; cin>>(a); if (c>b)and(a<c) n=n++; b=c; c=a; } cout<<n; } </pre>

Задача 7. Дано натуральное число n. Получить сумму тех чисел вида $i^3 - 3*i*n^2 + n$ ($i = 1, 2, \dots, n$), которые являются утроенными нечетными. <pre> Main () { int n,i,k,l; S: float; Cout<<'Enter n' Cin>>n For (i=1, n) S=sqr(i)*i-3*i+sqr(n)+n { If s % 3 = 0 K=s % 3; If k % <>0 L=1+s; } cout<<'Result='; coutl;} </pre>	Задача 7. Вычислить $\prod_{i,j=1}^{20} \frac{1}{i+j^2}$. <pre> Main () { int i,j; float P; For (i=1; 20) { p=1; For (j=1; 20) P=p+1/(i+Sqr(j)); } cout<<p; } </pre>
--	---

Задание 4

Вариант	Условный оператор
1, 11	Написать программу вычисления модуля введенного числа
2, 12	Написать программу проверки попадания введенного числа в диапазон от -2 до 2
3, 13	Написать программу проверки не вхождения введенного числа в диапазон от 0 до 5
4, 14	Написать программу проверки на положительность введенного числа
5, 15	Написать программу проверки на отрицательность введенного числа
6, 16	Написать программу определения знака введенного числа
7, 17	Написать программу проверки попадания введенного числа в диапазон от -6 до -2
8, 18	Написать программу проверки не вхождения введенного числа в диапазон от -5 до -1
9, 19	Написать программу вычисления суммы модулей двух введенных чисел
10, 20	Написать программу вычисления $1/a$ с проверкой $a \neq 0$

С помощью оператора выбора реализовать меню (возможность выбора номера нужной задачи для ее проверки преподавателем).

ЛАБОРАТОРНАЯ РАБОТА № 2

«Работа с массивами»

Задание 1

Написать программу в соответствии с вариантом задания для обработки одномерного массива.

1. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный элемент.
 3. Вычислить среднеарифметическое элементов массива.
 4. Вывести массив на экран в обратном порядке.
2. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Заполнить массив случайными числами.
 2. Найти минимальный элемент.
 3. Вычислить сумму элементов массива.
 4. Вывести положительные элементы на экран.
3. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный элемент.
 3. Вычислить среднеарифметическое положительных элементов массива.
 4. Вывести отрицательные элементы на экран в обратном порядке.
4. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Заполнить массив случайными числами.
 2. Найти минимальный положительный элемент.
 3. Вычислить произведение не нулевых элементов массива.
 4. Вывести ненулевые элементы на экран в обратном порядке.
5. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный отрицательный элемент.
 3. Вычислить сумму отрицательных элементов массива.
 4. Вывести положительные элементы на экран.
6. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Заполнить массив случайными числами.
 2. Найти максимальный положительный элемент.
 3. Вычислить сумму элементов массива.
 4. Вывести ненулевые элементы на экран в обратном порядке.
7. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный элемент.
 3. Вычислить среднеарифметическое отрицательных элементов массива.
 4. Вывести массив на экран в обратном порядке.
8. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Заполнить массив случайными числами.
 2. Найти минимальный элемент.
 3. Вычислить произведение не нулевых элементов массива.
 4. Вывести положительные элементы на экран в обратном порядке.
9. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти минимальный положительный элемент.
 3. Вычислить сумму положительных элементов массива, кратных 3.
 4. Вывести не нулевые элементы на экран.
10. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Заполнить массив случайными числами.
 2. Найти максимальный положительный элемент.
 3. Вычислить произведение элементов массива.
 4. Вывести положительные элементы на экран.

11. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный элемент.
 3. Вычислить сумму четных элементов массива.
 4. Вывести отрицательные элементы на экран в обратном порядке.
12. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Заполнить массив случайными числами.
 2. Найти минимальный отрицательный элемент.
 3. Вычислить среднеарифметическое положительных элементов массива.
 4. Вывести положительные элементы на экран.
13. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти максимальный отрицательный элемент.
 3. Вычислить произведение отрицательных элементов массива.
 4. Вывести ненулевые элементы на экран в обратном порядке.
14. Дан одномерный массив, состоящий из N целочисленных элементов.
 1. Заполнить массив случайными числами.
 2. Найти максимальный элемент.
 3. Вычислить среднеарифметическое нечетных элементов массива.
 4. Вывести отрицательные элементы на экран.
15. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Ввести массив с клавиатуры.
 2. Найти минимальный положительный элемент.
 3. Вычислить сумму четных элементов массива.
 4. Вывести массив на экран в обратном порядке.
16. Дан одномерный массив, состоящий из N вещественных элементов.
 1. Заполнить массив случайными числами.
 2. Найти минимальный отрицательный элемент.
 3. Вычислить произведение ненулевых элементов массива, кратных 3.
 4. Вывести отрицательные элементы на экран в обратном порядке.

Задание 2

Написать программу в соответствии с вариантом задания для обработки двумерного массива.

1. Дан двумерный массив размерностью 4×6 , заполненный целыми числами с клавиатуры. Сформировать одномерный массив, каждый элемент которого равен количеству элементов соответствующей строки, больших данного числа.
2. Дан двумерный массив размерностью 6×5 , заполненный целыми числами с клавиатуры. Сформировать одномерный массив, каждый элемент которого равен первому четному элементу соответствующего столбца, если такого нет, то равен нулю.
3. Дана матрица размером 5×4 . Поменять местами первую строку и строчку, в которой находится первый нулевой элемент.
4. Дан двумерный массив размером $n \times m$, заполненный случайными числами. Определить, есть ли в данном массиве столбец, в котором равно количество положительных и отрицательных элементов.
5. Дана матрица A размерностью $n \times m$. Сформировать одномерный массив B , элементами которого являются номера первых отрицательных элементов каждой строки массива A . (0 – отрицательный элемент отсутствует).
6. Дан двумерный массив размерностью 5×6 , заполненный целыми числами с клавиатуры. Сформировать одномерный массив, каждый элемент которого равен наибольшему по модулю элементу соответствующего столбца.
7. Найти среднее арифметическое элементов каждой строки матрицы $Q(l, m)$ и вычесть его из элементов этой строки.

8. Дан двумерный массив размером $n \times m$, заполненный случайными числами. Определить, есть ли в данном массиве строка, содержащая больше положительных элементов, чем отрицательных.
9. Дана матрица $K(n, m)$. Сформировать одномерный массив $L(m)$, элементами которого являются суммы элементов j -ого столбца.
10. Матрица $K(m, m)$ состоит из нулей и единиц. Найти в ней номера строк и столбцов, не содержащих единицы, либо сообщить, что таких нет.
11. Целочисленный массив $K(n, n)$ заполнить нулями и единицами, расположив их в шахматном порядке.
12. Дана матрица $A(n, m)$. Сформировать одномерный массив $B(n)$, элементами которого являются суммы элементов i -ой строки.
13. Дан двумерный массив размерностью 5×6 , заполненный целыми числами с клавиатуры. Сформировать одномерный массив, каждый элемент которого равен произведению четных положительных элементов соответствующего столбца.
14. Дан двумерный массив размером 8×7 , заполненный случайным образом. Заменить все элементы первых трех столбцов на их квадраты, в остальных столбцах изменить знак каждого элемента на противоположный.
15. Дана матрица размером 8×7 , заполненная случайным образом. Поменять местами две средние строки с первой и последней.
16. В каждой строке, заполненной случайным образом, матрицы размером $n \times m$ поменять местами первый элемент и максимальный по модулю.

Задание 3

Выполнить сортировку элементов одномерного массива согласно варианту.

Метод простого выбора – варианты 1, 7,

Метод простого обмена – варианты 2, 8

Метод прямого включения – варианты 3, 9, 13

Сортировка слияниями – варианты – 4, 10, 14

Метод быстрой сортировки – варианты 5, 11, 15

Пирамидальная сортировка – варианты 6, 12, 16

ЛАБОРАТОРНАЯ РАБОТА № 3

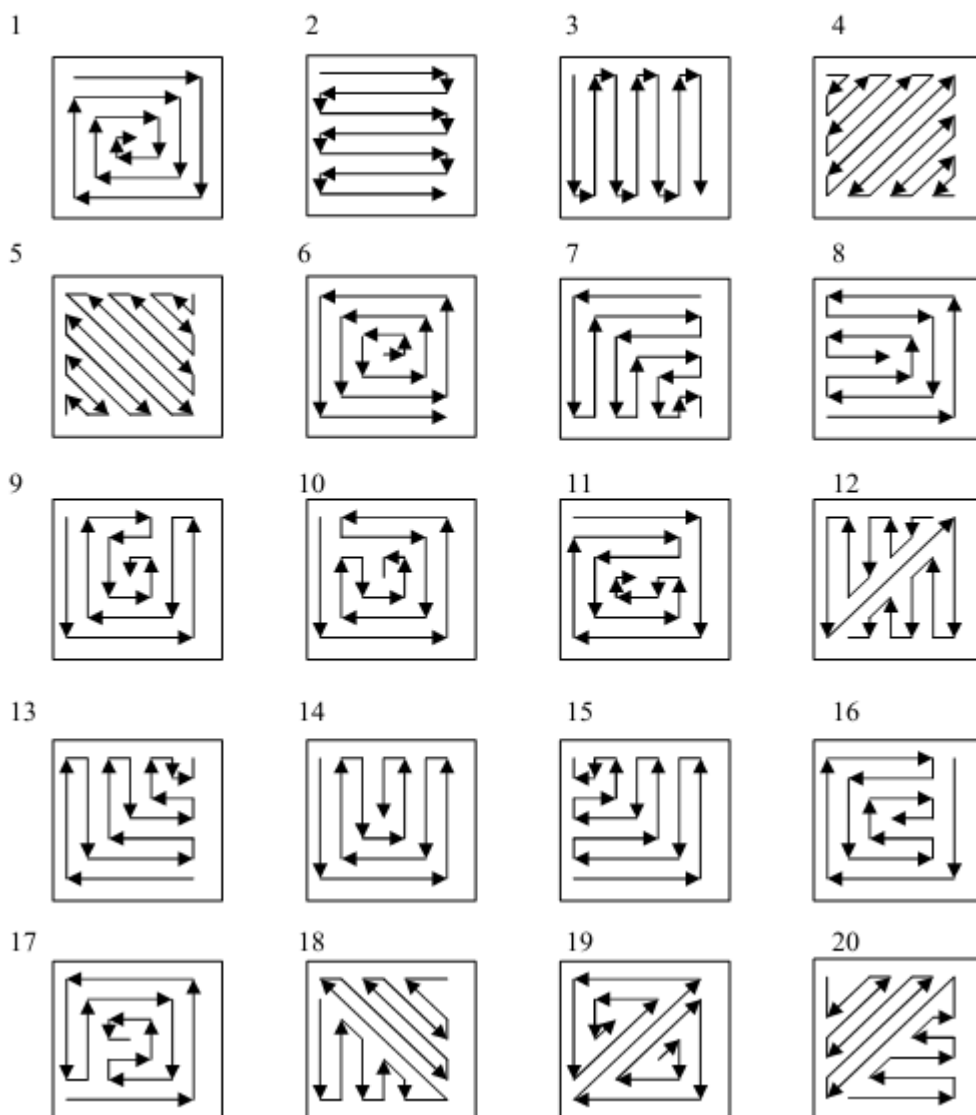
«Подпрограммы»

Написать функцию для заполнения квадратной матрицы размерностью $n \times n$ возрастающей последовательностью целых чисел по заданной схеме. Размерность матрицы задается константой и должна находиться в пределах $4 \leq n \leq 16$. Написать также функции вывода матрицы на экран и функцию нахождения обратной матрицы, функцию умножения матриц. Показать, что найдена обратная матрица, перемножив ее с исходной. Работу всех функций проиллюстрировать в программе.

Пример заполненной матрицы размером 4×4 по схеме варианта 1:

1	2	3	4
12	13	14	5
11	16	15	6
10	9	8	7

Варианты



ЛАБОРАТОРНАЯ РАБОТА № 4

«Работа со строками и файлами»

Задание	Текст задания
Вариант 1	
1.	Получить от пользователя строку текста и вставить вместо каждого пробела восклицательный знак. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Подсчитать количество слов в каждой строке текста. Записать в новый файл все строки, закончив их полученным числом. Например: Это пример строки текста в старом файле (7)
3.	Описать структуру с именем AEROFLOT, содержащую следующие поля: NUMR – номер рейса (целое число); COST – стоимость билета (число с плавающей точкой); NAZN – название пункта назначения рейса; TIP – тип самолета. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив AIRPORT, состоящий из семи элементов типа AEROFLOT; • вывод на экран номеров рейсов и типов самолетов, вылетающих в пункт назначения, название которого совпало с названием, введенным с клавиатуры; • если таких рейсов нет, выдать на дисплей соответствующее сообщение.
Вариант 2	
1.	Получить от пользователя строку текста и подсчитать количество символов в тексте. Вывести полученное значение на экран. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Вставить перед каждым восклицательным знаком вопросительный. Записать измененные строки в новый файл.
3.	Описать структуру с именем WORKER, содержащую следующие поля: SALARY – оклад; YEAR – год поступления на работу. NAME – фамилия и инициалы работника; POS – название занимаемой должности; Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив TABL, состоящий из десяти структур типа WORKER; • вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры; • если таких работников нет, вывести на дисплей соответствующее сообщение.

Вариант 3	
1.	Получить от пользователя строку текста и подсчитать количество точек в строке. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Променять местами четные и нечетные строки текста. Записать измененные строки в новый файл.
3.	Описать структуру с именем TRAIN, содержащую следующие поля: NUMR — номер поезда; TIME — время отправления; NAZN — название пункта назначения; OTPR - название пункта отправления. Написать программу, выполняющую следующие действия; • ввод с клавиатуры данных в массив RASP, состоящий из шести элементов типа TRAIN; • вывод на экран информации о поездах, направляющихся в пункт, название которого введено с клавиатуры; • если таких поездов нет, выдать на дисплей соответствующее сообщение.
Вариант 4	
1.	Получить от пользователя строку текста и Подсчитать количество пробелов в строке. Вывести полученное значение на экран. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Найти самую короткое предложение текста и заменить её фразой "С новым годом!". Записать измененный текст в новый файл.
3.	Описать структуру с именем MARSH, содержащую следующие поля: NUMER – номер маршрута; TIME – среднее время в пути; BEGST – название начального пункта маршрута; TERM – название конечного пункта маршрута; Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив TRAFIC, состоящий из восьми элементов типа MARSH; • вывод на экран информации о маршрутах, которые начинаются или кончаются в пункте, название которого введено с клавиатуры; • если таких маршрутов нет, выдать на дисплей соответствующее сообщение.

Вариант 5	
1.	Получить от пользователя строку текста и подсчитать количество двойных пробелов. Вывести полученное значение на экран. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Найти самую длинную строку текста и заменить её пустой строкой. Записать измененный текст в новый файл.
3.	Описать структуру с именем NOTE, содержащую следующие поля: NAME — фамилия, имя; TELE — номер телефона; GROUP - категория, к которой относится контакт (семья, работа и т.д.) BDAY — день рождения (массив из трех чисел). Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив BLOCKNOTE, состоящий из восьми элементов типа NOTE; • вывод на экран информации о человеке, номер телефона которого введен с клавиатуры; • если такого нет, выдать на дисплей соответствующее сообщение.
Вариант 6	
1.	Получить от пользователя строку текста и выделить из строки все числа. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Поменять местами самую длинную и самую короткую строки текста. Записать измененный текст в новый файл.
3.	Описать структуру с именем ZNAK, содержащую следующие поля: TIME – время рождения (массив из двух чисел); BDAY – день рождения (массив из трех чисел) NAME – фамилия, имя; ZODIAC – знак Зодиака. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив BOOK, состоящий из восьми элементов типа ZNAK; • вывод на экран информации о человеке, чья фамилия введена с клавиатуры; • если такого нет, выдать на дисплей соответствующее сообщение.
Вариант 7	
1.	Получить от пользователя строку текста и найти самое длинное слово текста. Повторять до тех пор, пока не встретится строка завершающаяся словом quit

2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Добавить в начало каждой строки текста её номер и пробел. Записать измененный текст в новый файл.
3.	Описать структуру с именем ORDER, содержащую следующие поля: PLAT – расчетный счет плательщика (); POL – расчетный счет получателя; DESCRP – описание назначения платежа; SUMMA – перечисляемая сумма в руб. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа ORDER; • вывод на экран информации о сумме, снятой с расчетного счета плательщика, введенного с клавиатуры; • если такого расчетного счета нет, выдать на дисплей соответствующее сообщение.

Вариант 8

1.	Получить от пользователя строку текста и Подсчитать количество восклицательных знаков в строке. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Записать в конец этого же файла таблицу с указанием трех наиболее часто встречающихся слов и количества их повторений в строке.
3.	Описать структуру с именем WIZARD, содержащую следующие поля: MANA – количество магической маны; LEADER – переключатель о статусе героя: является ли лидером группы; HITPOINT – запас жизненной силы; NAME – имя героя. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив TEAM, состоящий из восьми элементов типа WIZARD; • вывод на экран информации о герое с максимальным уровнем маны; • вывод на экран информации о герое с именем, введенным с клавиатуры. Если героя с такими именем нет, то выводится информация о лидере группы.

Вариант 9

1.	Получить от пользователя строку текста и Определить, содержит ли строка хотя бы пару соседних одинаковых слов. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
----	--

2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Записать в новый файл этот же текст и таблицу с указанием, сколько раз встретилась в каждой строке указанная пользователем буква латинского алфавита и пробел
3.	Описать структуру с именем TRACK, содержащую следующие поля: MODEL – марка боевой машины; SEATS – количество мест для размещения солдат; GUN_Size – калибр главного орудия; ARMOR – толщина брони в мм. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив SQUADRON, состоящий из восьми элементов типа TRACK; • вывод на экран информации о боевой машине с минимальной толщиной брони; • вывод на экран информации о боевой машине с введенным с клавиатуры калибром главного орудия. Если такой машины нет, то выводится информация о машине с наиболее близким к введенному значению калибром орудия.
Вариант 10	
1.	Получить от пользователя строку текста и Определить общую длину слов на нечётных позициях в строке. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Вывести в конец этого же файла таблицу с указанием количества слов, начинающихся с одной и той же буквы.
3.	Описать структуру с именем CHESSPEICE, содержащую следующие поля: COST – "стоимость" фигур в пешках; POSITION – номер клетки, в которой находится фигура; COLOR – цвет фигуры; NAME – название фигуры. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив CHESSBOARD, состоящий из 32 элементов типа CHESSPEICE; (ввести не менее 10 фигур с соблюдением правил игры в шахматы) • вывод на экран информации о всех фигурах указанного пользователем цвета; • вывод на экран информации о фигуре, расположенной в указанной пользователем клетке. Если в указанной клетке доски фигуры нет, то выводится информация о фигуре из ближайшей занятой клетки.

Вариант 11	
1.	Получить от пользователя строку текста и Определить количество знаков, не являющихся буквами или числами. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Перевести в верхний регистр все слова четных строк. И записать результат в новый файл.
3.	Описать структуру с именем PLANET, содержащую следующие поля: NAME – название планеты; RADIUS_PL– Радиус планеты; RADIUS_ORB – Радиус орбиты; SATELLITES_QNT – количество спутников, как естественных, так и искусственных; SAT_NAMES – массив с названиями спутников, как естественных, так и искусственных; Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив STAR_SYSTEM из N элементов типа PLANET, описывающий состав звездной системы; (ввести не менее 5 планет с указанием спутников как минимум для трех из них) • вывод на экран информации о всех планетах с одинаковым количеством спутников; • вывод на экран информации о планетах, радиус орбиты которых находится в указанном пользователем диапазоне. Если в указанной диапозоне нет планет, то выводится информация о всей звездной системе.
Вариант 12	
1.	Получить от пользователя строку текста и Перевести в верхний РЕГИСТР символы всех слов на четных позициях. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Определить количество предложений в тексте, учитывая, что предложение заканчивается точкой, вопросительным или восклицательным знаком. Вывести сводную таблицу в этот же файл.
3.	Описать структуру с именем BUTTON, содержащую следующие поля: TEXT – Текст, расположенный на кнопке; WIDTH – Ширина кнопки; HEIGHT – Высота кнопки; RESULT – модальный результат нажатия кнопки, то есть каким действиям приводит: Close (закрыть), Click (щелчок), Pressed (Нажата), Released (отжата) и т.д.; Написать программу, выполняющую следующие действия:

	• ввод с клавиатуры данных в массив WINDOW из N элементов типа BUTTON, описывающий диалоговое окно с кнопками; (ввести описание не менее 10 кнопок с указанием всех параметров) • вывод на экран информации о всех кнопках одинаковой ширины или длины (в зависимости от выбора пользователя); • вывод на экран информации о кнопках, модальный результат которых соответствует введенному пользователем значению.
--	--

Вариант 13

1.	Получить от пользователя строку текста и определить, сколько вопросительных знаков, в строке Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Добавить в начало каждого предложения текста его номер и пробел. Вывести результат в новый файл.
3.	Описать структуру с именем PLANET, содержащую следующие поля: NAME – название планеты; RADIUS_PL– Радиус планеты; RADIUS_ORB – Радиус орбиты; SATELLITES_QNT – количество спутников, как естественных, так и искусственных; SAT_NAMES – массив с названиями спутников, как естественных, так и искусственных; Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив STAR_SYSTEM из N элементов типа PLANET, описывающий состав звездной системы; (ввести не менее 5 планет с указанием спутников как минимум для трех из них) • вывод на экран информации о всех планетах с одинаковым количеством спутников; • вывод на экран информации о планетах, радиус орбиты которых находится в указанном пользователем диапазоне. Если в указанной диапозоне нет планет, то выводится информация о всей звездной системе.

Вариант 14

1.	Получить от пользователя строку текста и Променять местами первое и последнее слова в строке. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Определить количество пробелов в нечётных строках текста Вывести сводную таблицу в конце этого же файла.
3.	Описать структуру с именем ZNAK, содержащую следующие поля: TIME – время рождения (массив из двух чисел); BDAY – день рождения (массив из трех чисел)

	NAME – фамилия, имя; ZODIAC – знак Зодиака. Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив BOOK, состоящий из восьми элементов типа ZNAK; • вывод на экран информации о человеке, чья фамилия введена с клавиатуры; • если такого нет, выдать на дисплей соответствующее сообщение.
Вариант 15	
1.	Получить от пользователя строку текста и Променять местами первое слово, начинающееся с гласной буквы и последнее слово, начинающееся с согласной буквы. Повторять до тех пор, пока не встретится строка завершающаяся словом quit
2.	Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). (В файле должно быть не менее 30 строк). Выделить все числа из текста и подсчитать их сумму и среднее. Вывести результат в конце этого же файла.
3.	Описать структуру с именем WORKER, содержащую следующие поля: SALARY – оклад; YEAR – год поступления на работу. NAME – фамилия и инициалы работника; POS – название занимаемой должности; Написать программу, выполняющую следующие действия: • ввод с клавиатуры данных в массив TABL, состоящий из десяти структур типа WORKER; • вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры; • если таких работников нет, вывести на дисплей соответствующее сообщение.

ЛАБОРАТОРНАЯ РАБОТА № 5

«Классы, свойства, индексаторы»

Цель работы: получить навыки создания простейших классов с использованием принципа инкапсуляции, свойств, индексаторов, перегрузки операторов.

Лабораторное задание

Спроектировать класс согласно варианту. Продемонстрировать работу класса в консольном приложении. В программе должна выполняться проверка всех разработанных элементов класса.

Вариант 1

Создать класс Point, содержащий следующие члены класса:

- поля: `int x, y`;
- конструкторы, позволяющие создать экземпляр класса с нулевыми координатами, с заданными координатами;
- методы, позволяющие вывести координаты точки на экран, рассчитать расстояние от начала координат до точки, переместить точку на плоскости на вектор (a, b) ;
- свойства, позволяющие получить/установить координаты точки (доступное для чтения и записи), умножить координаты точки на скаляр (доступное только для записи);
- индексатор, позволяющий по индексу 0 обращаться к полю `x`, по индексу 1 – к полю `y`, при других значениях индекса выдается сообщение об ошибке;
- перегрузку: операции `++` (`--`) – одновременно увеличивает (уменьшает) значение полей `x` и `y` на 1; констант `true` и `false` – обращение к экземпляру класса дает значение `true`, если значение полей `x` и `y` совпадает, иначе `false`;
- операции бинарный `+` – одновременно добавляет к полям `x` и `y` значение скаляра.

Вариант 2

Создать класс Triangle, содержащий следующие члены класса:

- поля: `int a, b, c`;
- конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;
- методы, позволяющие вывести длины сторон треугольника на экран, рассчитать периметр треугольника, рассчитать площадь треугольника;

- свойства, позволяющие получить/установить длины сторон треугольника (доступное для чтения и записи), установить, существует ли треугольник с данными длинами сторон (доступное только для чтения);
- индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, по индексу 2 – к полю c, при других значениях индекса выдается сообщение об ошибке;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a, b и c на 1; констант true и false – обращение к экземпляру класса дает значение true, если треугольник с заданными длинами сторон существует, иначе – false; операции * – одновременно умножает поля a, b и c на скаляр.

Вариант 3

Создать класс Rectangle, содержащий следующие члены класса:

- поля: int a, b;
- конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;
- методы, позволяющие вывести длины сторон прямоугольника на экран, рассчитать периметр прямоугольника, рассчитать площадь прямоугольника;
- свойства, позволяющие получить/установить длины сторон прямоугольника (доступное для чтения и записи), установить, является ли данный прямоугольник квадратом (доступное только для чтения);
- индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, при других значениях индекса выдается сообщение об ошибке;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a и b; констант true и false – обращение к экземпляру класса дает значение true, если прямоугольник с заданными длинами сторон является квадратом, иначе – false; операции * – одновременно умножает поля a и b на скаляр.

Вариант 4

Создать класс Money, содержащий следующие члены класса:

- поля: int first (номинал купюры); int second (количество купюр);
- конструктор, позволяющий создать экземпляр класса с заданными значениям полей;
- методы, позволяющие вывести номинал и количество купюр, определить, хватит ли денежных средств на покупку товара на сумму N рублей, определить, сколько штук товара стоимости n рублей можно купить на имеющиеся денежные средства;

- свойства, позволяющие получить/установить значение полей (доступное для чтения и записи), рассчитать сумму денег (доступное только для чтения);
- индексатор, позволяющий по индексу 0 обращаться к полю first, по индексу 1 – к полю second, при других значениях индекса выдается сообщение об ошибке;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей first и second; операции ! – возвращает значение true, если поле second не нулевое, иначе false; операции бинарный + – добавляет к значению поля second значение скаляра.

Вариант 5

Создать класс для работы с одномерным массивом целых чисел. Разработать следующие члены класса:

- поля: int [] IntArray;
- конструктор, позволяющий создать массив размерности n;
- методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы массива в порядке возрастания;
- свойство, возвращающее размерность массива (доступное только для чтения), и свойство, позволяющее умножить все элементы массива на скаляр (доступное только для записи);
- индексатор, позволяющий по индексу обращаться к соответствующему элементу массива;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; операции ! – возвращает значение true, если элементы массива не упорядочены по возрастанию, иначе – false; операции бинарный * – умножить все элементы массива на скаляр; операции преобразования класса массив в одномерный массив (и наоборот).

Вариант 6

Создать класс для работы с двумерным массивом целых чисел. Разработать следующие члены класса:

- поля: int [,] intArray;
- конструктор, позволяющий создать массив размерности $n \times m$;
- методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, вычислить сумму элементов i -го столбца;
- свойства, позволяющие вычислить количество нулевых элементов в массиве (доступное только для чтения), установить значение всех элементов главной диагонали массива, равное скаляру (доступное только для записи);

- двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если двумерный массив является квадратным; операции бинарный + – сложить два массива соответствующих размерностей; операции преобразования класса массив в двумерный массив (и наоборот).

Вариант 7

Создать класс для работы с двумерным массивом вещественных чисел. Разработать следующие функциональные члены класса:

- поля: double [][] doubleArray;
- конструктор, позволяющий создать ступенчатый массив;
- методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы каждой строки массива в порядке убывания;
- свойство, возвращающее общее количество элементов в массиве (доступное только для чтения), и свойство, позволяющее увеличить значение всех элементов массива на скаляр (доступное только для записи);
- двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если каждая строка массива упорядочена по возрастанию, иначе – false; операции преобразования класса массив в ступенчатый массив (и наоборот).

Вариант 8

Создать класс для работы со строками. Разработать следующие члены класса:

- поле: string line;
- конструктор, позволяющий создать строку на основе заданного строкового литерала;
- методы, позволяющие подсчитать количество цифр в строке, выводить на экран все символы строки, встречающиеся в ней ровно один раз, вывести на экран самую длинную последовательность повторяющихся символов в строке;
- свойство, возвращающее общее количество символов в строке (доступное только для чтения);

- индексатор, позволяющий по индексу обращаться к соответствующему символу строки (доступный только для чтения);

- перегрузку: операции унарного ! – возвращает значение true, если строка не пустая, иначе – false; констант true и false – обращение к экземпляру класса дает значение true, если строка является палиндромом, false – в противном случае; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования класса строка в тип string (и наоборот).

Вариант 9

Создать класс для работы со строками. Разработать следующие члены класса:

- поле: `StringBuilder line`;
- конструктор, позволяющий создать строку на основе заданного строкового литерала, и конструктор, позволяющий создавать пустую строку;
- методы, позволяющие подсчитать количество пробелов в строке, заменить в строке все прописные символы на строчные, удалить из строки все знаки препинания;
- свойство, возвращающее общее количество элементов в строке (доступное только для чтения), и свойство, позволяющее установить значение поля в соответствии с введенным значением строки с клавиатуры, а также получить значение данного поля (доступно для чтения и записи);
- индексатор, позволяющий по индексу обращаться к соответствующему символу строки;
- перегрузку: операции унарного + (-) – преобразующей строку к строчным (прописным) символам; констант true и false – обращение к экземпляру класса дает значение true, если строка не пустая, иначе – false; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования строки в тип `StringBuilder` (и наоборот).

Вариант 10

Самостоятельно изучите тип данных `DateTime`, на основе которого необходимо создать класс для работы с датой. Данный класс должен содержать следующие члены класса:

- поле `DateTime data`;
- конструкторы, позволяющие установить заданную дату, дату 1.01.2000;
- методы, позволяющие вычислить дату предыдущего дня, вычислить дату следующего дня, определить сколько дней осталось до конца месяца;

- свойства, позволяющие установить или получить значение поле класса (доступно для чтения и записи), определить, является ли год високосным (доступно только для чтения);

- индексатор, позволяющий определить дату i-го по счету дня относительно установленной даты (при отрицательных значениях индекса отсчет ведется в обратном порядке);

- перегрузку: операции ! – возвращает значение true, если установленная дата не является последним днем месяца, иначе – false; констант true и false – обращение к экземпляру класса дает значение true, если установленная дата является началом года, иначе – false; операции & – возвращает значение true, если поля двух объектов равны, иначе false.

Вариант 11

Создать класс Triangle, содержащий следующие члены класса:

- поля: int a, b, c;

- конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

- методы, позволяющие вывести длины сторон треугольника на экран, рассчитать периметр треугольника, рассчитать площадь треугольника;

- свойства, позволяющие получить/установить длины сторон треугольника (доступное для чтения и записи), установить, существует ли треугольник с данными длинами сторон (доступное только для чтения);

- индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, по индексу 2 – к полю c, при других значениях индекса выдается сообщение об ошибке;

- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a, b и c на 1; констант true и false – обращение к экземпляру класса дает значение true, если треугольник с заданными длинами сторон существует, иначе – false; операции * – одновременно умножает поля a, b и c на скаляр.

Вариант 12

Создать класс Rectangle, содержащий следующие члены класса:

- поля: int a, b;

- конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

- методы, позволяющие вывести длины сторон прямоугольника на экран, рассчитать периметр прямоугольника, рассчитать площадь прямоугольника;

- свойства, позволяющие получить/установить длины сторон прямоугольника (доступное для чтения и записи), установить, является ли данный прямоугольник квадратом (доступное только для чтения);
- индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, при других значениях индекса выдается сообщение об ошибке;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a и b; констант true и false – обращение к экземпляру класса дает значение true, если прямоугольник с заданными длинами сторон является квадратом, иначе – false; операции * – одновременно умножает поля a и b на скаляр.

Вариант 13

Создать класс для работы с двумерным массивом целых чисел. Разработать следующие члены класса:

- поля: int [,] intArray;
- конструктор, позволяющий создать массив размерности $n \times m$;
- методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, вычислить сумму элементов i-го столбца;
- свойства, позволяющие вычислить количество нулевых элементов в массиве (доступное только для чтения), установить значение всех элементов главной диагонали массива, равное скаляру (доступное только для записи);
- двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;
- перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если двумерный массив является квадратным; операции бинарный + – сложить два массива соответствующих размерностей; операции преобразования класса массив в двумерный массив (и наоборот).

Вариант 14

Создать класс для работы с двумерным массивом вещественных чисел. Разработать следующие функциональные члены класса:

- поля: double [][] doubleArray;
- конструктор, позволяющий создать ступенчатый массив;
- методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы каждой строки массива в порядке убывания;
- свойство, возвращающее общее количество элементов в массиве (доступное только для чтения), и свойство, позволяющее увеличить значение всех элементов массива на скаляр (доступное только для записи);

– двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если каждая строка массива упорядочена по возрастанию, иначе – false; операции преобразования класса массив в ступенчатый массив (и наоборот).

Вариант 15

Создать класс для работы со строками. Разработать следующие члены класса:

– поле: `StringBuilder line`;

– конструктор, позволяющий создать строку на основе заданного строкового литерала, и конструктор, позволяющий создавать пустую строку;

– методы, позволяющие подсчитать количество пробелов в строке, заменить в строке все прописные символы на строчные, удалить из строки все знаки препинания;

– свойство, возвращающее общее количество элементов в строке (доступное только для чтения), и свойство, позволяющее установить значение поля в соответствии с введенным значением строки с клавиатуры, а также получить значение данного поля (доступно для чтения и записи);

– индексатор, позволяющий по индексу обращаться к соответствующему символу строки;

– перегрузку: операции унарного + (-) – преобразующей строку к строчным (прописным) символам; констант true и false – обращение к экземпляру класса дает значение true, если строка не пустая, иначе – false; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования строки в тип `StringBuilder` (и наоборот).

Лабораторная работа № 6

«Наследование»

Создайте иерархию классов, используя наследование. В каждой программе необходимо соблюсти принцип разделения интерфейса и реализации класса (иными словами, не забывайте выделять заголовочные файлы). В каждом варианте необходимо написать программу, иллюстрирующую применение всех методов ваших классов. Прежде чем приступить к написанию программ, продумайте, какие необходимы функции в каждом из классов (может, где-то необходимо считать координаты, где-то площади и объемы, а где-то хранить фамилии и года поступления): как в базовом, так и в классах-наследниках. Также продумайте, что следует поместить в закрытые (а, возможно, защищенные) переменные. Предусмотрите возможность переопределения методов базового класса в производном. Приветствуется демонстрация наследования перегруженных операций. Возможно, в некоторых вариантах лучше воспользоваться композицией, чем наследованием.

Варианты

1. Точка -> Квадрат -> Куб.
2. Четырехугольник -> Трапеция -> Параллелограмм.
3. Точка -> Четырехугольник -> Параллелепипед.
4. Точка -> Треугольник -> Треугольная призма.
5. Точка -> Круг -> Сфера.
6. Факультет:
 - а) администрация;
 - б) преподаватель;
 - с) студент.
7. Учащийся в университете:
 - а) студент;
 - б) аспирант;
 - с) слушатель.
8. Студент:
 - а) первокурсник;

б) студент 2-4 курса;

с) дипломник.

9. Круг -> Конус -> Усеченный конус.

10. Треугольник -> Треугольная пирамида -> Усеченная треугольная пирамида.

11. Прямолинейное движение делится на равномерное и равноускоренное. Равноускоренное в свою очередь может реализовываться свободным падением по вертикали.

12. Криволинейное движение можно разбить на движение по окружности и падение тела, брошенного под углом к горизонту. Из движения тела, брошенного под углом к горизонту, можно выделить свободное падение тела, брошенного горизонтально.

13. Спортсмен -> Легкоатлет -> Спринтер.

14. Точка -> Треугольник -> Равнобедренный треугольник -> Равносторонний треугольник.

15. Транспортное средство:

а) трамвай;

б) троллейбус;

с) автобус.

16. Чемпионат по программированию -> Университетский тур -> Городской тур -> Областной тур.

17. Студент -> Математик -> Математик-программист.

18. Среди накопителей информации можно выделить такие классы, как жесткий диск и флеш-карта. Среди жестких дисков, в свою очередь, можно выделить класс съемных дисков.

19. Точка -> Треугольник -> Прямоугольный треугольник -> Равнобедренный прямоугольный треугольник.

20. Точка -> Четырехугольник -> Ромб -> Квадрат.

Лабораторная работа № 7. Виртуальные функции и полиморфизм

Часть 1 (до 3 баллов)

Вам даны классы BinaryOperation (бинарный оператор) и Number (число), которые наследуются от базового класса Expression (выражение). Ваша задача реализовать базовый класс Expression так, чтобы не было утечек памяти. Кроме этого подумайте, какие методы стоит сделать виртуальными.

```
#include <cassert> // assert
using namespace std;

struct Expression
{
    // здесь должен быть ваш код
};

struct Number : Expression
{
    Number(double value) : value_(value) {}
    double value() const { return value_; }
    double evaluate() const { return value_; }
private:
    double value_;
};

struct BinaryOperation : Expression
{
    enum {
        PLUS = '+',
        MINUS = '-',
        DIV = '/',
        MUL = '*'
    };

    BinaryOperation(Expression const *left, int op,
                    Expression const *right):left_(left), op_(op), right_(right)
    {
        assert(left_ && right_);
    }

    ~BinaryOperation()
    {
        delete left_;
        delete right_;
    }

    Expression const *left() const { return left_; }
    Expression const *right() const { return right_; }
    int operation() const { return op_; }
};
```

```

double evaluate() const
{
    double left = left_>evaluate();
    double right = right_>evaluate();
    switch (op_)
    {
        case PLUS: return left + right;
        case MINUS: return left - right;
        case DIV: return left / right;
        case MUL: return left * right;
    }
    assert(0);
    return 0.0;
}

private:
    Expression const *left_;
    Expression const *right_;
    int op_;
};

```

Проверьте полученную иерархию на следующем фрагменте кода:

```

//-----
Expression * e1 = new Number(1.234);
Expression * e2 = new Number(-1.234);
Expression * e3 = new BinaryOperation(e1,
                                     BinaryOperation::DIV, e2);
cout<<e3->evaluate()<<endl;
//-----

```

Часть 2 (до 5 баллов)

Добавьте к иерархии из предыдущего упражнения класс-наследник FunctionCall. FunctionCall должен представлять вызов одной из двух предопределенных математических функций: sqrt — извлечение квадратного корня и abs — вычисление модуля числа. Функция идентифицируется строкой, переданной в качестве параметра в конструктор. Не забудьте, что у функции должен быть аргумент (которым может быть любое выражение Expression)!

```

#include <string> // std::string
#include <cassert>
#include <cmath> // sqrt и fabs
// эти классы из предыдущего упражнения
struct Expression;
struct BinaryOperation;
struct Number;
struct FunctionCall : Expression
{
    /**
     * @name - это имя функции, возможные варианты
     * "sqrt" и "abs".
     *
     * Объекты, std::string можно
     */
};

```

```

        * сравнивать с C-строками используя
        * обычный синтаксис ==.
        *
        * @arg - выражение-аргумент функции
        */

FunctionCall(/*аргументы*/)
{
    //реализация
}

// реализуйте оставшиеся методы из
// интерфейса структуры Expression и не забудьте
// удалить arg_, как это сделано в классе BinaryOperation
//также реализуйте предложенные ниже методы

std::string const & name() const
{
    // реализация
}

Expression const *arg() const
{
    //реализация
}

~FunctionCall(){/*реализация*/}

private:
    //а тут должны быть поля для FunctionCall
};

```

Проверьте полученную иерархию на следующем фрагменте кода:

```

//-----
Expression* n32 = new Number(32.0);
Expression* n16 = new Number(16.0);
Expression* minus = new BinaryOperation(n32, BinaryOperation::MINUS, n16);
Expression* callSqrt = new FunctionCall("sqrt", minus);
Expression* n2= new Number(2.0);
Expression* mult = new BinaryOperation(n2, BinaryOperation::MUL, callSqrt);
Expression* callAbs = new FunctionCall("abs", mult);
cout<<callAbs->evaluate()<<endl;
//-----

```

Часть 3 (до 7 баллов)

Создайте иерархию классов, используя наследование от абстрактного базового класса. В каждой программе необходимо соблюсти принцип разделения интерфейса и реализации класса. В каждом варианте необходимо написать программу, иллюстрирующую применение всех методов ваших классов. Прежде чем приступить к написанию программ, продумайте (или посоветуйтесь с преподавателем), какие необходимы функции в каждом из классов. Также продумайте, что следует поместить в закрытые или защищенные переменные. Особенно обратите внимание на то, какие функции следует сделать чистыми виртуальными. Обязательно предусмотрите виртуальный деструктор! В основной программе обязательно используйте динамическое связывание (без него работа не принимается).

Варианты

1. Координаты точки на плоскости – это способ описания ее положения в двумерном пространстве. Способов описания может быть много, но мы ограничимся декартовыми, полярными и естественными координатами. Рассмотрите класс «Координата» как базовый, а перечисленные способы описания положения на плоскости реализуйте конкретными классами. Можете также поэкспериментировать с функциями – друзьями классов, которые могут переводить один способ описания в другой.
2. Усложним задачу №1 – перейдем в пространство. Здесь остановимся на декартовых, сферических и цилиндрических координатах. Далее сделайте то же, что и в первой задаче.
3. Пока с утра вы едете в университет, вы являетесь пассажиром. Реализуйте абстрактный базовый класс «Пассажир» и конкретные классы – пассажир с билетом, льготной сезонкой, транспортной картой (помните, что их бывает несколько видов), «заяц». Попробуйте подсчитать, сколько пассажиров каждого «вида» едет в час пик и какова выручка кондуктора с них.
4. Реализуйте иерархию Форма -> Точка -> Круг -> Сфера. Можете попробовать написать для этих классов функции рисования, а не только расчета площади, периметра и т.п.
5. То же, что и в задаче №5, но иерархия Форма -> Точка -> Многоугольник -> Многогранник.
6. Все мы любим отдыхать. Но абстрактное понятие «нерабочий день» может на самом деле оказаться конкретным выходным, праздником или отпуском. Реализуйте такую иерархию. Разумеется, самый большой приоритет имеют выходные – на них могут попасть и праздники, и отпуск.
7. Вспомним основы механики. Наиболее общее понятие «движение» конкретизировалось сначала понятиями прямолинейное и криволинейное движение. Прямолинейное движение в свою очередь еще более сильно конкретизировали: равномерное и неравномерное движение. А уж из неравномерного движения выделяли еще равноускоренное движение. С криволинейным все немного проще – достаточно рассмотреть движение по окружности как частный случай более общего криволинейного движения. Реализуйте иерархию классов, используя абстрактный базовый класс «Движение». Не забудьте про основные характеристики движения!
8. Напишите абстрактный класс «Студент», которому наследуют конкретные классы отличник, хорошист, троечник, должник. Не забудьте успевающим студентам начислить стипендию, а неуспевающим – дату пересдачи долгов. Естественно, количество пересдач ограничено – не более 3. Предусмотрите возможность отчисления неуспевающих студентов или ухода их в академический отпуск.
9. При слове «Авиабилет» у вас наверняка возникают самые разные ассоциации – ведь это может быть билет на простой междугородний рейс или на международный. Это может быть билет первого класса, бизнес-класса, второго класса... Реализуйте иерархию «Авиабилет» -> «Междугородний» -> «Международный». Помните, что требования на приобретение международного авиабилета более жесткие, значит данных в этом классе (или функций) будет больше.
10. Представьте, что вы только что выиграли чемпионат по программированию. Вы стали абстрактным победителем, чтобы о вас знали как о конкретном человеке, победившем в конкретном чемпионате, надо определить несколько вещей. Во-первых, свои персональные данные (хотя бы ФИО). Во-вторых, вид и тур соревнования: университетский тур, городской, областной, федеральный или международный. Начиная с абстрактного класса «Победитель чемпионата», реализуйте приведенную выше иерархию классов.

11. Попробуйте немного себя в роли бухгалтера, начисляющего зарплату. Само по себе понятие «Зарплата» не особенно конкретное: оно включает и почасовую, и ставочную зарплату, и комиссионные,
и процент с продаж (если работа связана с продажей). Реализуйте данные классы.
12. Вычислительное средство само по себе является понятием абстрактным. Оно может представлять из себя бухгалтерские счёты, арифмометр, калькулятор или современную ПЭВМ. Реализуйте такую иерархию.
13. Поговорим о транспортных средствах. На дороге можно встретить грузовой, легковой и пассажирский транспорт. Каждый из этих видов транспорта обладает общими, а также специфическими характеристиками (подумайте, какими). Напишите иерархию классов «Автомобиль» (абстрактный базовый) и далее 3 производных конкретных класса. Подумайте, какие характеристики нужны для вывода на печать, попробуйте также рассчитать средний тормозной путь, грузоподъемность, вместимость соответствующих транспортных средств.
14. Вся компьютерная графика делится на векторную и растровую. Так как сильно отличаются принципы построения рисунков этих видов, можно выделить программы, поддерживающие тот или иной вид графики и сохраняющие рисунки в том или ином формате. Попробуйте написать программу с использованием полиморфизма и абстрактного класса, реализующую данную схему.
15. Не за горами сессия, и скоро такое абстрактное понятие, как «Контроль успеваемости», превратится во вполне реальное понятие зачет или экзамен (в устной или письменной форме, или, быть может, в форме теста). Подумайте, какие элементы и функции входят в абстрактный базовый класс «Контроль успеваемости» и как их реализовать для конкретных производных классов. Напишите программу.
16. Попробуйте выстроить логическую цепочку: Студент университета N -> Студент факультета N -> Студент специальности N -> Студент курса N. Вместо N каждый раз будет нужно подставить название вуза, факультета, специальности, номер курса. Реализуйте цепочку через абстрактные и конкретные классы. Возможно, для конкретной специальности придется ввести новые предметы, а для конкретного курса – количество часов на изучение определенных дисциплин.
17. Вы уже не первый год изучаете языки программирования. Наверняка вы знаете, что их можно подразделить на языки структурного программирования и языки объектно-ориентированного программирования. К первым можно отнести, например, C и Pascal, ко вторым – C++, Java... Реализуйте эту структуру через абстрактные и конкретные классы.
18. Понятие "здание" относится к абстрактным. Чтобы его конкретизировать, необходимо знать, является ли оно жилым домом, или торговым центром, или административным зданием, либо спортивным (крытый каток, бассейн) или культурным (театр) объектом... Разумеется, у всех них будут какие-то общие характеристики (например, адрес), и у каждого из них – свои особенные характеристики (скажем, у жилого дома – этажность, количество подъездов, наличие газовых коммуникаций, ГВС, наличие и количество лифтов, наличие мусоропровода и т.д., у бассейна – количество акваторий и их размер (например, две 50-метровые акватории по 9 дорожек), количество мест в раздевалке, душ...). Реализуйте иерархию классов.
19. Все основные неорганические вещества на нашей планете можно подразделить на соли, кислоты, основания и оксиды. Естественно, все категории веществ состоят из химических элементов. Сделав класс «Химический элемент» абстрактным базовым, опишите приведенную здесь иерархию.

Вспомните, что все эти вещества могут взаимодействовать друг с другом, превращаясь друг в друга по определенным законам.

20. Напишите абстрактный класс «Учащийся». Его конкретными реализациями будут ученик, студент, аспирант. Подумайте, как выстроить подобную иерархию и какими общими и специфическими характеристиками будет обладать каждый класс. Напишите реализацию данной иерархии.

Лабораторная работа № 8.

Обработка исключений

В данной лабораторной работе вам нужно написать класс исключения и программу, способную генерировать и обрабатывать определенный вид исключения (программа должна содержать блоки `try`, `catch`, точку `throw`). Позаботьтесь о том, чтобы исключение действительно могло возникнуть, продемонстрируйте работу перехватчика и обработчика исключений.

Варианты

Обработать следующие исключения:

1. Взятие квадратного корня из отрицательного числа.
2. Нехватка памяти при динамическом ее выделении.
3. Попытка записи в файл, открытый только для чтения.
4. Ввод пользователем вещественного числа вместо целого.
5. Ввод пользователем строки вместо числа.
6. Ввод пользователем специального символа вместо целого числа.
7. Взятие натурального логарифма от нуля.
8. Взятие натурального логарифма от отрицательного числа.
9. Ввод пользователем несуществующей даты, например, «31 февраля».
10. Ввод несуществующего времени, например, 56 час. 335 мин. 98 сек.
11. Ввод пользователем отрицательного возраста.
12. Ввод пользователем отрицательной зарплаты.
13. Попытка чтения из несуществующего файла.
14. Попытка записи в несуществующий файл.
15. Неопределенность вида «0/0».
16. Неопределенность вида «0*¥».
17. Ввод числа, спецсимволов или русских букв вместо символов латинского алфавита.
18. Ввод числа, спецсимволов или латинских букв вместо символов русского алфавита.
19. Превышение при вводе максимально возможной длины строки.
20. Выход индекса за пределы массива.

ЛАБОРАТОРНАЯ РАБОТА № 9

Регулярные выражения. Шаблоны

Задание 1

- 1 В файле с телефонными переговорами клиента определить по какому номеру выполнено максимальное количество звонков.
- 2 Выполнить анализ кода программы на наличие в нем всех циклов `for` языка C++. Напечатать результаты анализа на экране монитора.
- 3 Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество знаков пунктуации.
- 4 В файле с телефонными переговорами клиента выполнить сортировку переговоров по их стоимости.
- 5 Выполнить анализ кода программы на наличие в нем комментариев языка C++. Напечатать результаты анализа на экране монитора.
- 6 Считать текст из файла и вывести его на экран монитора, заменив цифры словами, например, «0» на слово «ноль»; «1» на слово «один» и т. д.
- 7 В файле с телефонными переговорами клиента определить, по какому номеру выполнено максимальное количество звонков.
- 8 Выполнить анализ кода программы на наличие в нем всех операторов присваивания языка C++ (не учитывать записи типа `for (i=0; i<=10; i++) ...`).
- 9 Напечатать результаты анализа на экране монитора. Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество чисел (не цифр). В файле с телефонными переговорами клиента определить, по какому номеру был самый продолжительный по времени разговор.
- 10 Выполнить анализ кода программы на наличие в нем всех операторов вывода информации на экран (консоль) языка C++. Напечатать результаты анализа на экране монитора.
- 11 Считать текст из файла и вывести его на экран монитора только цитаты текста, т. е. предложения, заключенные в кавычки.
- 12 В файле с телефонными переговорами клиента определить, по какому номеру выполнены подряд (2 и более соединений) звонки.
- 13 Выполнить анализ кода программы на наличие в нем всех операторов ввода информации с клавиатуры (консоли) языка C++. Напечатать результаты анализа на экране монитора.
- 14 Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество повторяющихся слов.
- 15 В файле с телефонными переговорами клиента определить все номера, время связи с которыми было менее 5 с.

Примечание

В некоторых вариантах индивидуальных заданий необходимо выполнить анализ файла телефонных переговоров клиента. В этом файле, кроме прочей информации, должны содержаться записи об абонентах, включающих следующие сведения: – номер абонента, с которым выполнялось соединение; – дату и время соединения; – продолжительность соединения; – стоимость переговоров.

Задание 2 находится в файле «ЛАБОРАТОРНАЯ РАБОТА № 9_2»

Упражнение 1. (1 балл) Реализуйте шаблонную версию класса Array. Список всех операций, которые должен поддерживать класс Array, приведен в шаблоне кода. Проверьте написанный код на типах int, double, string, char*, пользовательском типе Student (для студента храните имя и номер зачетки, а также определите необходимые методы — подумайте над конструкторами и деструкторами). Шаблон класса Array должен успешно работать с данными любого типа.

```
#include <cstddef>
using namespace std;

template <typename T>
class Array
```

```

{ private:
    T* myArray;
    size_t n;
public:
    // Список операций:
explicit Array(size_t size = 0, const T& value = T())
    // конструктор класса, который создает
    // Array размера size, заполненный значениями
    // value типа T. Считайте что у типа T есть
    // конструктор, который можно вызвать без
    // без параметров, либо он ему не нужен.
    {}
Array(const Array & mas)
    // конструктор копирования, который создает
    // копию параметра. Считайте, что для типа
    // T определен оператор присваивания.
    { }
~Array()
    // деструктор, если он вам необходим.
    {}
Array& operator=(const Array& mas)
    // оператор присваивания.
    {}
    //
    // две версии оператора доступа по индексу.
T& operator[](size_t idx)
    {}
const T& operator[](size_t idx) const
    {}
size_t size() const {}
};

```

Упражнение 2. (1 балл) В предыдущей версии предполагается, что для типа T определен оператор присваивания или он ему не нужен (например, для примитивных типов он не нужен). При создании шаблонных классов контейнеров (вроде Array и не только) разумно стараться минимизировать требования к типам шаблонных параметров. Поэтому усложним задачу, реализуйте класс Array не полагаясь на то, что для типа T определен оператор присваивания.

Подсказка: возможно, понадобится placement new и явный вызов деструктора, чтобы создавать и уничтожать объекты, аллоцировать правильно выровненную память можно с помощью new char[N * sizeof(T)], где N – количество элементов массива.

```

#include <cstddef>

template <typename T>
class Array
{ private:
    T* myArray;
    size_t n;
public:
    // Список операций:
explicit Array(size_t size=0, const T& value=T())
    // конструктор класса, который создает

```

```

        // Array размера size, заполненный значениями
        // value типа T. Считайте что у типа T есть
        // конструктор, который можно вызвать без
        // без параметров, либо он ему не нужен.
        {}
Array(const Array & mas)
    // конструктор копирования, который создает
    // копию параметра. Считайте, что для типа
    // T определен оператор присваивания.
    { }
~Array()
    // деструктор, если он вам необходим.
    { }
Array& operator=(const Array& mas)
    // оператор присваивания.
    { }
    // две версии оператора доступа по индексу.
T& operator[](size_t idx)
    { }
const T& operator[](size_t idx) const
    { }
size_t size() const {}
};

```

Упражнение 3. (1 балл) Шаблонные классы можно наследовать. Реализуйте шаблонную структуру ValueHolder с одним типовым параметром T, унаследованную от интерфейса ICloneable. Интерфейс ICloneable содержит всего один виртуальный метод ICloneable* clone() const, который должен вернуть указатель на копию объекта, на котором он был вызван (объект должен быть создан в куче). Шаблон ValueHolder, как говорит его название, хранит всего одно значение (назовите его data_) типа T (для типа T определен конструктор копирования). Не делайте поле data_ закрытым (поэтому в данном случае мы явно пишем, что ValueHolder должна быть структурой).

```

struct ICloneable
{
    virtual ICloneable* clone() const = 0;
    virtual ~ICloneable() { }
};

// Шаблон ValueHolder с типовым параметром T,
// должен содержать одно открытое поле data_
// типа T.

// В шаблоне ValueHolder должен быть определен
// конструктор от одного параметра типа T,
// который инициализирует поле data_.
//
// Шаблон ValueHolder должен реализовывать
// интерфейс ICloneable, и возвращать указатель
// на копию объекта созданную в куче из метода
// clone.

```

Упражнение 4. (1 балл) Реализуйте функцию копирования элементов сору_n из массива источника типа U* в целевой массив типа T*, где T и U произвольные типы, для которых определено преобразование из

U в T. На вход функция принимает два указателя и количество элементов, которые необходимо скопировать.

Пример вызова функции `copy_n`:

```
int ints[] = {1, 2, 3, 4};
double doubles[4] = {};
copy_n(doubles, ints, 4);
// теперь в массиве doubles содержатся
// элементы 1.0, 2.0, 3.0 и 4.0

#include <cstdint>
// Параметры функции copy_n идут в следующем
// порядке:
// 1. целевой массив
// 2. массив источник
// 3. количество элементов, которые нужно
// скопировать
//
// Вам нужно реализовать только функцию copy_n,
// чтобы ее можно было вызвать так, как показано
// в примере.
```

Упражнение 5. (1 балл) Реализуйте шаблонную функцию `minimum`, которая находит минимальный элемент, который хранится в экземпляре шаблонного класса `Array`, при этом типовой параметр шаблона `Array` может быть произвольным. Чтобы сравнивать объекты произвольного типа, на вход функции также будет передаваться компаратор, в качестве компаратора может выступать функция или объект класса с перегруженным оператором «круглые скобки» `()`.

Примеры вызова функции `minimum`:

```
bool less(int a, int b) { return a < b; }

struct Greater
{ bool operator()(int a, int b) { return b < a; } };

Array<int> ints(3);
ints[0] = 10;
ints[1] = 2;
ints[2] = 15;

int min = minimum(ints, less); // в min должно попасть 2
int max = minimum(ints, Greater()); // в max должно попасть 15

#include <cstdint>
template <typename T>
class Array
{
public:
    explicit Array(size_t size = 0, const T& value = T());
    Array(const Array& other);
    ~Array();
```

```

    Array& operator=(Array other);
    void swap(Array &other);
    size_t size() const;
    T& operator[](size_t idx);
    const T& operator[](size_t idx) const;
private:
    size_t size_;
    T *data_;
};
// Ваш код

```

Упражнение 6. (2 балла) Шаблонный класс `Array` может хранить объекты любого типа, для которого определён конструктор копирования, в том числе и другой `Array`, например, `Array< Array<int> >`. Глубина вложенности может быть произвольной. Напишите шаблонную функцию (или несколько) `flatten`, которая принимает на вход такой "многомерный" `Array` неизвестной заранее глубины вложенности и выводит в поток `out` через пробел все элементы, хранящиеся на самом нижнем уровне.

Примеры работы функции `flatten`:

```

Array<int> ints(2, 0);
ints[0] = 10;
ints[1] = 20;
flatten(ints, std::cout); // выводит на экран строку "10 20"
Array< Array<int> > array_of_ints(2, ints);
flatten(array_of_ints, std::cout);
// выводит на экран строку "10 20 10 20"
Array<double> doubles(10, 0.0);
flatten(doubles, std::cout);
// работать должно не только для типа int

```

Примечание: лидирующие и завершающие пробельные символы будут игнорироваться, т. е. там где ожидается "10 20" будет так же принят, например, вариант " 10 20 ", но не вывод "1020".

Подсказка: шаблонные функции тоже можно перегружать, из нескольких шаблонных функций будет выбрана наиболее специфичная.

```

#include <iostream>
// Весь вывод должен осуществляться в поток out,
// переданный в качестве параметра.
//
// Можно заводить любые вспомогательные функции,
// структуры или даже изменять сигнатуру flatten,
// но при этом все примеры вызова из задания должны
// компилироваться и работать.

```

Упражнение 7. (1 балл) Выше вы реализовали простой шаблон `ValueHolder`, в этом задании мы используем его чтобы написать класс `Any` (интересно, что не шаблонный), который позволяет хранить значения любого типа! Например, вы сможете создать массив объектов типа `Any`, и сохранять в них `int`-ы, `double`-ы или даже объекты `Array`. Подробности в шаблоне кода.

Подсказка: в нешаблонном классе `Any` могут быть шаблонные методы, например, шаблонный конструктор.

```

struct ICloneable;
// Поле data_ типа T в классе ValueHolder
// открыто, к нему можно обращаться

template <typename T>
struct ValueHolder;
// Это класс, который вам нужно реализовать

class Any
{
public:
// В классе Any должен быть конструктор, который можно вызвать
// без параметров, чтобы работал следующий код:
// Any empty; // empty ничего не хранит

// В классе Any должен быть шаблонный конструктор от одного
// параметра, чтобы можно было создавать объекты типа Any,
// например, следующим образом:
// Any i(10); // i хранит значение 10

// Деструктор: выделенные ресурсы нужно освободить.

// В классе Any также должен быть конструктор копирования (вам
// поможет метод clone интерфейса ICloneable)

// В классе должен быть оператор присваивания и/или шаблонный
// оператор присваивания, чтобы работал следующий код:
// Any copy(i); // copy хранит 10, как и i
// empty = copy; // empty хранит 10, как и copy
// empty = 0; // а теперь empty хранит 0

// Чтобы получать хранимое значение, определите в классе Any
// шаблонный метод cast, который возвращает указатель на
// хранимое значение, или нулевой указатель в случае
// несоответствия типов или если объект Any ничего не хранит:
// int *iptr = i.cast<int>(); // *iptr == 10
// char *cptr = i.cast<char>(); // cptr == 0,
// // потому что i хранит int, а не char
// Any empty2;
// int *p = empty2.cast<int>(); // p == 0
// При реализации используйте dynamic_cast.
};

```

Упражнение 8. (1 балл) В качестве упражнения на частичную специализацию шаблонов классов вам предлагается реализовать простой шаблон SameType. Этот шаблон не содержит никаких методов, а только одно **статическое константное поле типа bool** с именем value. Шаблон принимает два типовых параметра, и если два типовых параметра шаблона являются одним и тем же типом, то статическое поле value должно хранить значение true, в противном случае – значение false.

Примеры:

```
struct Dummy { };
typedef int type;
std::cout << SameType<int, int>::value << std::endl;
// выведет 1, т. е. true
std::cout << SameType<int, type>::value << std::endl;
// 1, type == int
std::cout << SameType<int, int&>::value << std::endl;
// 0, int и ссылка на int - различные типы
std::cout << SameType<Dummy, Dummy>::value << std::endl; // 1
std::cout << SameType<int, const int>::value << std::endl;
// 0, const - часть типа

// Определите шаблон SameType с двумя типовыми параметрами.
// В шаблоне должна быть определена одна статическая константа // типа bool с именем
value
```

Упражнение 9. (1 балл) Реализуйте функцию `array_size`, которая возвращает размер массива, переданного в качестве параметра. Функция должна работать только для массивов! Т. е. если функции передать указатель, должна произойти ошибка компиляции.

Примеры:

```
int ints[] = {1, 2, 3, 4};
int *iptr = ints;
double doubles[] = {3.14};
array_size(ints); // вернет 4
array_size(doubles); // вернет 1
array_size(iptr); // тут должна произойти ошибка компиляции
```

Подсказка: можно организовать передачу в функцию массивов только заданного размера (передача массива по ссылке), совместите его с вашими знаниями о шаблонах.

ЛАБОРАТОРНАЯ РАБОТА № 10

«Многопоточное программирование»

Цель работы: ознакомиться с организацией многопоточной обработки данных, получить основные навыки программирования с использованием потоков и работы с визуальными компонентами.

Лабораторное задание

Разработать консольное и визуальное приложение согласно варианту. Требования к программе: реализовать возможность задавать приоритет каждого из порожденных потоков; использовать символы псевдографики для визуализации потоков на экране монитора.

1 Умножение матрицы на вектор. Обработку одной строки матрицы производить в порожденном потоке.

2 Поиск всех простых чисел (простым называется число, которое является своим наибольшим делителем) в указанном интервале чисел, разделенном на несколько диапазонов. Обработка каждого диапазона производится в порожденном потоке. Классический алгоритм Евклида определения наибольшего общего делителя двух целых чисел (x , y) может применяться при следующих условиях:

- оба числа x и y неотрицательные;
- оба числа x и y отличны от нуля.

На каждом шаге алгоритма выполняются сравнения:

- если $x == y$, то ответ найден;
- если $x < y$, то y заменяется значением $y - x$;
- если $x > y$, то x заменяется значением $x - y$.

3 Винни-Пух и пчелы. Заданное количество пчел добывают мед равными порциями, задерживаясь в пути на случайное время. Винни-Пух потребляет мед порциями заданной величины за заданное время и столько же времени может прожить без питания. Работа каждой пчелы реализуется в порожденном потоке.

4 Шарики. Координаты заданного количества шариков изменяются на случайную величину по вертикали и горизонтали. При выпадении шарика за нижнюю границу допустимой области шарик исчезает. Изменение координат каждого шарика в отдельном потоке.

5 Противостояние нескольких команд. Каждая команда увеличивается на случайное количество бойцов и убивает случайное количество бойцов участника. Борьба каждой команды реализуется в отдельном потоке.

6 Контрольная сумма. Для нескольких файлов (разного размера) требуется вычислить контрольную сумму (сумму кодов всех символов файла). Обработка каждого файла выполняется в отдельном потоке.

7 Бег с препятствиями. Создается условная карта трассы в виде матрицы, ширина которой соответствует количеству бегунов, а высота фиксирована, которая содержит произвольное количество единиц (препятствий) в произвольных ячейках. Стартующие бегуны (потоки) перемещаются по трассе и при встрече с препятствием задерживаются на фиксированное время. По достижении финиша бегуны сообщают свой номер.

8 Создать два потока. Первый ищет числа Фибоначчи (каждое последующее число равно сумме двух предыдущих чисел), второй – простые числа. Результат работы каждого потока сохраняется в отдельный файл. После остановки потока программа производит анализ файлов, выводит их на экран, а так же показывает количество найденных чисел Фибоначчи и простых чисел.

9 Создать два потока. Первый поток производит запись в файл случайных данных, второй производит чтение данных из этого файла и вывод их на экран.

10 Создать приложение, выполняющее вычисление значений функции $y = 23 \cdot x^2 - 33$, с шагом $x=0.01$. Первый поток выполняет расчёт функции и добавляет результаты расчёта в конец массива. Второй поток извлекает из массива значения x и y и выводит их на экран.

11 Создать приложение, которое выполняет сортировку массива данных и отображает процесс сортировки на экране. Первый поток производит сортировку по возрастанию, второй – по убыванию. После каждого перемещения элементов производится вывод на экран текущего состояния сортировки. Каждый поток работает с отдельным экземпляром массива данных.

12 Создать игру, где будут 2...3 барана и волк. При совпадении координат волка с бараном баран исчезает. При совпадении координат баранов появляется новый баран. Все движутся хаотически.

13 Создать три потока, генерирующих случайным образом целые числа от 0 до 9. При нажатии на клавишу «Enter» потоки останавливаются и результат анализируется. Цель анализа – выявить наличие следующих комбинаций цифр в потоках: три одинаковых числа, два одинаковых числа, три единицы, три семерки, две единицы.

14 Создать три потока, каждый из которых управляет перемещением псевдосимвола на экране вдоль оси X, и устроить «тараканьи бега» среди них.

15 Поиск указанной строки в указанном файле. Обработка одной строки в порожденном потоке.

Лабораторная работа № 11

АБСТРАКЦИЯ И ИНКАПСУЛЯЦИЯ

Цель работы: изучить основы объектно-ориентированного программирования на алгоритмическом языке C++. Получить представление о реализации принципов абстракции и инкапсуляции.

Порядок выполнения работы

1. Изучить семантику и синтаксис описания классов на языке C++.
2. Реализовать класс Complex для работы с комплексными числами. Класс должен обеспечивать инкапсуляцию информации о действительной и мнимой частях комплексного числа, а также о его модуле и амплитуде (в представлении Эйлера). Соответствующие поля должны являться закрытыми (private), а для доступа к их значениям должны быть описаны методы записи и чтения (например, для работы с действительной частью комплексного числа это могут быть методы setRe() и getRe()). Между двумя представлениями комплексного числа (и) должна осуществляться синхронизация при любых изменениях значения комплексного числа в том или ином представлении. Конструкторы класса должны обеспечивать возможность создания объекта тремя способами: со значениями полей по умолчанию (действительная и мнимая части равны нулю), создания объекта с заданием действительной и мнимой частей, а также создания объекта в представлении Эйлера. Для класса должны быть реализованы арифметические методы (сложение, вычитание, умножение, деление) и специальные методы ввода/вывода.
3. Разработать и реализовать тестовое приложение для класса Complex, демонстрирующее использование его методов.
4. Выполнение работы продемонстрировать преподавателю.
5. После приёмки работы преподавателем оформить отчёт.
6. Защитить работу преподавателю, ответив на заданные вопросы.