ЛАБОРАТОРНАЯ РАБОТА №1

Цель работы:

Приобрести умения и практические навыки для работы по созданию линейных программ и программ с разветвлениями; приобрести умения и практические навыки для работы по созданию программ с циклами.

Теоретическая часть:

Алфавит – основные неделимые знаки, с помощью которых пишутся все тексты программы (*0...9, a…z, A…Z,* специальные знаки – *(), [], {}, ‘’, “”,* +, -,*, /, %, точка, запятая, двоеточие, точка с запятой,* ~, ^, *!, ?, #, &,* ||, *пробел*).

Пробельные символы – знак пробел, знак табуляции, знак перехода на новую строку.

Из символов алфавита формируются *лексемы* – минимальные единицы языка, имеющие самостоятельный смысл. К ним относятся: идентификаторы, ключевые слова, знаки операций, константы, разделители. Границы лексем определяются другими лексемами.

Идентификатор – имя программных объектов. Идентификаторы состоят из букв, цифр и знака нижнего подчеркивания. Они не могут начинаться с цифры, не допускаются пробелы, теоретически их длина не ограничена. Нельзя выбирать в качестве идентификатора *ключевые слова* – зарезервированные идентификаторы, имеющие специальное назначение. Различаются строчные и заглавные символы. Идентификаторы могут быть, например, такими: *A1, Kol_vo, x*.

Знаки операции – один или более символ, которые определяют действие над операндом. Знаки пробела не допустимы.

Константы бывают следующих типов:

– *целые* – десятичные, восьмеричные (начинаются с 0), шестнадцатеричные (начинаются с 0x или 0X), двоичные (начинаются с 0b);

– *вещественные* – с десятичной точкой, экспоненциальная форма записи – $8e^{-13} = 8*10^{-13}$, $1.3E126 = 1.3*10^{126}$;

– *символьные* – один или более символов, заключенные в апострофы

(‘a’, ‘*’, ‘ф’). *Escape-последовательность* – последовательность, которая начинается с обратной косой черты. Такая последовательность рассматривается как один символ (‘\n’ для переноса на новую строку, ‘\t’ для табуляции, ‘\\’ для написания обратного слэша и т.д.);

– *строковые* – последовательность символов, заключенная в кавычки (“a”, “строка”). Если идут несколько строк подряд, они объединяются в одну. Длинную строковую константу можно записать на нескольких строчках, при этом обратная косая черта будет обозначать знак переноса.

Комментарий при компиляции кода не учитывается. Если комментарий располагается в одну строчку, то он должен начинаться с //. Если он состоит из нескольких строк, то синтаксис комментария таков: /* комментарий */, например:

```
/* это комментарий,
который располагается
на трех строчках */
```

Как правило, программа на языке Си состоит из следующих элементов:

– директивы препроцессора;

– глобальные объекты;

– функции.

Директивы препроцессора – директивы, используемые для присоединения специальных файлов, содержащих описание применяемых в программе функций. Подключаются к проекту с помощью директивы #include.

```
#include <stdio.h>
```

Глобальные объекты – в данном блоке описываются глобальные переменные и состояния, используемые в программе.

Функции – любая программа состоит из них – могут вызывать друг друга, но не могут содержаться одна в другой. Одна из функций обязательно имеет имя *main* – с нее начинается выполнение программы. Существует несколько форматов этой функции, один из которых выглядит следующим образом:

```
void main () {тело функции}
```

Тело функции содержит описания и операторы языка Си.

Тип данных определяет:

– внутреннее представление данных в памяти;

– множество значений, которые могут принимать переменные этого типа;

– операции, которые можно применять к переменным данного типа.

В языке Си существуют следующие базовые типы данных:

– *целый – int* – можно перед данным типом также писать *short* (тогда размер составит 2 байта) или *long* (4 байта). Целое число в двоичной системе счисления во внутреннем представлении, по умолчанию занимает 4 байта в памяти. Если *signed* (по умолчанию) – со знаком, *unsigned* – без знака;

– *символьный – char* – под него отводится 1 байт. Также может быть *signed* (по умолчанию) или *unsigned*. Данный тип может использоваться для представления целых чисел от -128 до 127 или от 0 до 255;

– *вещественный – float* (одинарной точности, занимает в памяти 4 байта), *double* (удвоенной точности, занимает в памяти 8 байтов), *long double* (максимальной точности, занимает в памяти 10 байт);

– тип *void* – тип, множество значений которого пусто. Используется для описания функций, которые не возвращают результаты в вызывающую программу и указания пустого списка параметров.

Оператор sizeof возвращает размер памяти в байтах, которую занимает выражение или тип. Чтобы получить размер типа данных, этот тип помещается в скобки после sizeof:

```
sizeof(int);
```

Также sizeof может вычислять размер какого-нибудь объекта, например, переменной. В этом случае название объекта передается в скобки, либо может идти после оператора без скобок.

Спецификации преобразования позволяют задать формат вывода различных типов данных. Определение спецификаций преобразования выглядит следующим образом:

```
% [флаги ширина_поля.точность модификатор] спецификатор
```

Основные спецификации преобразования для базовых типов данных:

– %c: для отдельных символов (тип char);

– %s: для строк;

– %d: для целых чисел со знаком (тип int);

– %i: для целых чисел со знаком (тип int);

– %u: для целых положительных чисел (тип unsigned);

– %f: для чисел с плавающей точкой (float, double);

– %e: для экспоненциального представления чисел с плавающей точкой (float, double);

– %o: для восьмеричных чисел без знака;

– %x: для шестнадцатеричных чисел;

– %%: для знака процента.

Для вывода числовых значений с плавающей точкой мы можем использовать ширину поля и точность. Ширина поля представляет целое положительное число, которое определяет длину выводимого значения в символах. Точность — это также целое положительное число, которое определяет количество цифр в дробной части.

Модификаторы позволяют конкретизировать выводимое значение. Используются следующие модификаторы:

– h: для значений short int;

– l: для значений long int и unsigned long int;

– L: для значений long double.

Дополнительно можно использовать флаги, которые позволяют управлять форматированием при выводе:

– **-** указывает, что выравнивание будет идти от левого края (по умолчанию используется выравнивание справа).

– **+** если выводимое значение имеет знак (+ или -), то оно выводится. Без данного флага знак выводится только в случае отрицательного значения.

– *пробел* вставляет пробел на месте знака перед положительными числами

– **#** при использовании со спецификаторами "o", "x", "X" значение числа выводится с предшествующими символами 0, 0x или 0X. При использовании со спецификаторами "f", "g", "G" десятичная точка будет выводиться, даже если в числе нет дробной части.

Операторы делятся на *исполняемые* и *неисполняемые*.

Исполняемые операторы задают действие над данными. *Неисполняемые* являются описаниями данных (описания). Каждый оператор завершается *точкой с запятой*.

Переменные – поименованная область памяти, в которой хранятся данные определенного типа. Имеют имя и значение. Бывают *локальные* и *глобальные*.

Локальные переменные объявлены внутри блока, их область действия – от точки объявления и до конца блока (включая внутренние блоки).

Глобальные переменные объявлены вне блока, действуют от точки описания и до конца файла. Любая переменная должна быть описана перед использованием.

Оператор описания переменной имеет следующий синтаксис:

```
const тип имя [размерность] = инициализатор;
```

– *имя* – идентификатор;

– *тип* – любой тип данных в языке Си;

– *инициализатор* – позволяет присвоить переменной начальное значение при ее объявлении;

– *const* – указывает, что объявлена константа. Константы не разрешается изменять, следовательно, они должны сразу иметь

инициализатор;

– *размерность* – необходима для описания массивов;

Можно объявлять несколько переменных, разделяя их знаком запятой.

Для вывода на консоль в языке Си используется функция printf():

```
printf(строка_форматирования, список_аргументов);
printf("Имя: %s \t Возраст: %d \t Рост: %3.1f", "Иван", 22, 178.557);
```

Для ввода данных с консоли используется функция scanf_s():

```
scanf_s(форматная_строка, аргументы);
scanf_s("%d", &age);
```

Выражения состоят из операндов и знаков операций. Они используются для вычисления значения определенного типа.

Операции бывают *унарные* (один операнд), *бинарные* (два операнда) и *тернарные* (три операнда). Каждый операнд может быть выражением или, в частном случае, константой или переменной. Если рядом расположены операции одного приоритета, то:

– справа налево вычисляются унарные, тернарная (условная), операция присваивания;

– слева направо вычисляются все остальные операции.

Операции *инкремента (++)* и *декремента (- -)* имеют две формы записи:

– *префиксная* – операнд изменяется, новое значение является результатом выражения;

```
int a = 2, b;
b = ++a;  // a = 3, b = 3
```

– *постфиксная* – результат выражения – исходное значение операнда, после вычисления выражения операнд изменяется.

```
b = a++;  // b = 2, a = 3
```

Операция *деления* и *остатка от деления (%)* применяется только к целым, результат также будет целым.

Результатом выполнения *операций сравнения* является значение «истина» или «ложь». Существуют следующие операции сравнения: ==, !=, >,

>=, <, <=).

Логические операции – «и» (&&), «или» (||) – выполняются над всем аргументом в соответствии с таблицей истинности каждой из операций.

Операция *отрицания* в языке Си выглядит следующим образом:

```
int x = 5, y;
y = -x;   // y = -5
y = !x;   // y = 0 (возвращает 1, если x = 0; 0 в остальных случаях)
```

Преобразование типа (type) производится следующим образом:

```
a = x / (float) y;  // a = 3,5 при x = 7 и y = 2
```

Результат вычисления выражения имеет значение и тип. Если операнды одного типа, то результат имеет тот же тип. Если операнды имеют различный тип, то перед вычислением выполняется преобразование от более коротких к более длинным типам для сохранения значимости и точности, при этом может измениться внутреннее представление переменных.

В таблице представлены основные операции языка Си и их приоритет выполнения:

Приоритет	**Действие**
Унарные операции	
1	++ икремент -- декремент
1	& взятие адреса
1	* разыменование
1	*(type)* преобразование типа
1	*sizeof* размер
Бинарные и тернарные операции	
2	* умножение / деление % остаток от деления
3	+ сложение

	- вычитание
4	< меньше <= меньше или равно > больше >= больше или равно
5	= = равно ! = не равно
6	&& логическое И
7	| | логическое ИЛИ
8	? : условная операция (тернарная)
9	= присваивание * = умножение с присваиванием / = деление с присваиванием % = остаток от деления с присваиванием + = сложение с присваиванием - = вычитание с присваиванием
10	, последовательное вычисление

В языке Си существуют различные операторы.

выражение; – оператор вычисления выражений, частный случай выражения – пустой оператор, он используется, когда по синтаксису требуется оператор, а по смыслу – нет.

Операторы ветвлений – условный оператор

```
if (выражение) оператор1;
else оператор2;
```

Выражение должно иметь логический тип. Если нужно выполнить несколько операторов, то их заключают в фигурные скобки. Типичной ошибкой при использовании операторов ветвления является неправильное использование операций сравнения, например, использование операции присваивания вместо операции сравнения и т.д., а также запятая с точкой,

поставленная после оператора *if ()*.

Также к операторам ветвления относится *оператор ветвления* switch. Он имеет следующий синтаксис:

```
switch (выражение) {
case константное выражение 1: оператор1;
…
case константное выражение N: оператор N;
default: оператор;
}
```

Выражение должно быть целочисленным или символьным. Данный оператор предназначен для разветвления процесса вычисления на несколько направлений. Выполнение оператора начинается с вычисления выражения, результат сравнивается с константными выражениями. Управление передается первому оператору из списка, помеченного константным выражением, совпавшим с результатом выражения. Затем выполняются операторы из всех ветвей, включая *default* – необязательную ветвь, которая получает управление, если не одно константное выражение не совпало с результатом.

Оператор *break* применяется для того, чтобы избежать выполнение операторов и других ветвлений. Предназначен для выхода из оператора *switch* или операторов цикла. Этот оператор передает управление оператору, следующему за оператором *switch* или циклом.

Цикл – многократное выполнение одного или группы операторов. *Тело цикла* – выполняемые операторы. Однократное выполнение тела называется *итерацией*.

Существуют различные операторы цикла. Рассмотрим их далее.

```
while (выражение) оператор;
```

Этот оператор называется *оператором цикла с предусловием*. Если выражение не выполняется, происходит выход из цикла. Выражение должно быть логического типа.

```
#include "stdafx.h"
```

```
void main() {
int x;
scanf_s(“%d”, &x);
while (x) {
printf (“%d”, x);
scanf (“%d”, &x);
}
```

или:

```
while (1) {
scanf_s (“%d”, &x);
if (x) break;
printf (“%d”, x);
}
```

Тело цикла при использовании этого оператора может ни разу не выполниться.

```
do оператор; while (выражение)
```

Этот оператор называется *оператором цикла с постусловием*. Выражение также должно быть логического типа.

```
do {
scanf_s (“%d”, &x);
printf (“%d”, x);
} while (x);
```

```
for (инициализация; выражение; модификация) оператор;
```

Этот оператор цикла называется *оператором цикла с параметром*. *Инициализация* – объявление и присвоение начальных значений величинам в цикле. Выражений может быть несколько. Их результатом должен быть логический тип. *Модификацией* называется изменение параметров цикла после каждой итерации. Инициализация производится один раз в начале цикла, причем переменные из нее действуют только в самом цикле. Любая часть может быть пустой, но точки с запятой сохраняются.

Оператор *continue;* передает управление на начало следующей итерации, пропуская все операторы до конца тела цикла.

Если в теле одного цикла содержится другой оператор цикла, то говорят о *вложенных циклах*.

При использовании кириллических символов в языке Си можно столкнуться с ситуацией, когда вместо кириллических символов отображаются непонятные знаки. В этом случае необходимо явным образом задать текущую среду для вывода символов. Это делается с помощью функции setlocale(), определение которой имеется в заголовочном файле locale.h.

```
char *locale = setlocale(LC_ALL, "");
```

Задание 1.

Варианты:

1. Дано четырехзначное целое число. Вычислить и вывести на экран сумму цифр этого числа. Число вводится с клавиатуры.

2. Дано четырехзначное целое число. Определить, равна ли сумма старших цифр сумме младших. Число вводится с клавиатуры.

3. Определить число, полученное выписыванием в обратном порядке цифр заданного трехзначного числа, и вывести его на экран. Число вводится с клавиатуры.

4. Найти и вывести на экран произведение цифр заданного четырехзначного числа. Число вводится с клавиатуры.

5. Ввести с клавиатуры значение x, вычислить и вывести на экран значение y:

$$y = \begin{cases} x, \text{если } x \leq 0 \\ x^2, \text{если } 0 < x \leq 1 \\ x^3, \text{если } x > 1 \end{cases}$$

6. Ввести с клавиатуры значение x, вычислить и вывести на экран значение y:

$$y = \begin{cases} x, \text{если } x > 1 \\ x^4, \text{если } 0 < x \leq 1 \\ x^2, \text{если } x \leq 0 \end{cases}$$

7. Ввести число x – возраст человека. Определить: ходит ли он в ясли (от 1 до 3 лет), ходит ли он в детский сад (от 4 до 6 лет), ходит ли он в школу (от 7 до 18 лет), уже окончил школу (19 лет и старше).

8. Ввести число t от 0 до 24 – час дня. Определить, какому времени суток соответствует этот час: от 0 до 3 – ночь, от 4 до 10 – утро, от 11 до 17 – день, от 18 до 21 – вечер, от 22 до 24 – ночь. Число вводится с клавиатуры.

9. Дано число t – температура воздуха. Выводить на экран сообщение о погоде: холодно, прохладно, тепло, жарко. Число вводится с клавиатуры.

10. Даны a, b и c – углы треугольника в градусах. Определить, можно ли построить такой треугольник, и если можно, то определить тип треугольника.

Значения углов вводятся с клавиатуры.

11. Даны *a, b* и *c* – стороны треугольника. Проверить, можно ли построить треугольник с такими сторонами, и если можно, то определить его тип (равносторонний, равнобедренный, разносторонний). Значения сторон треугольника вводятся с клавиатуры.

12. Даны 4 числа: оценки студента на экзаменах. Если он получил хотя бы одну оценку «2» – его отправят на дополнительную сессию. Если он получил хотя бы одну оценку «3» – его лишат стипендии. Если он получил все оценки «5» – он получит повышенную стипендию. Вывести на экран сообщение о решении деканата. Оценки вводятся с клавиатуры.

13. Дано 3 целых числа. Вывести на экран «Да» или «Нет» в зависимости от того, имеют 3 числа одинаковую четность или нет. Числа вводятся с клавиатуры.

14. Даны целые числа *a, b, c* и *d*. Найти и вывести на экран наибольшее из них. Числа вводятся с клавиатуры

15. Даны целые числа *a, b, c* и *d*. Найти и вывести на экран число, которое ближе всего к среднему арифметическому этих чисел. Числа вводятся с клавиатуры.

16. Дано пятизначное целое число. Определить, является ли это число палиндромом (читается одинаково слева направо и справа налево). Число вводится с клавиатуры.

17. Дано трехзначное целое число. Найти и вывести на экран сумму его четных цифр. Число вводится с клавиатуры.

18. Вводится число x – количество дней. Определить, сколько это недель и дней (например, 10 дней = 1 неделя и 3 дня).

19. Вводится число x – температура в градусах Цельсия. Преобразовать его в градусы Фаренгейта и вывести результат на экран по формуле: F = C * 9/5 + 32.

20. Вводится число x – количество баллов за тест. Определить, какой оценке соответствует этот балл (меньше 51 – 2, от 51 до 70 – 3, от 71 до 85 –

4, от 86 до 100 – 5).

21. Вводятся два целых числа x и y – координаты точки в пространстве. Определить, в какой четверти находится точка на плоскости.

Задание 2.

Варианты:

1. Вычислить и вывести на экран значения функции:

$$y = \begin{cases} 3x^2 + 2, \text{если } 1 \leq x < 2 \\ x, \text{если } 2 \leq x < 4 \\ \dfrac{5}{x-8}, \text{если } 4 \leq x \leq 9 \end{cases}$$

при x, который изменяется на интервале от 1 до 9 с шагом 1.

2. Найти на интервале от 9,1 до 10 корень уравнения $x^3 + x = 1000$ с точностью до 0,0001.

3. Дана непустая последовательность положительных целых чисел, за которыми следует 0 (признак конца последовательности). Вычислить среднее арифметическое этих чисел. Последовательность должна вводится с клавиатуры.

4. Дана непустая последовательность положительных целых чисел, за которыми следует 0 (признак конца последовательности). Вычислить сумму этих чисел. Последовательность должна вводится с клавиатуры.

5. Дана непустая последовательность положительных целых чисел, за которыми следует 0 (признак конца последовательности). Вычислить произведение этих чисел. Последовательность должна вводится с клавиатуры.

6. Вывести на экран латинские буквы, коды которых четны.

7. Вывести на экран латинские буквы, коды которых нечетны.

8. Вывести на экран латинские буквы, коды которых кратны введенному с клавиатуры числу от 1 до 10.

9. Дана дата в виде: день, месяц. Напечатать, сколько дней прошло от начала года. Дата вводится с клавиатуры.

10. Дано целое $n > 0$, за которым следует n вещественных чисел. Определить, сколько среди них отрицательных. Значение n и вещественные числа вводятся с клавиатуры.

11. Дано целое $n > 0$, за которым следует n вещественных чисел. Определить, сколько среди них положительных. Значение n и вещественные

числа вводятся с клавиатуры.

12. Дано целое $n > 0$, за которым следует n вещественных чисел. Определить, сколько среди них кратных 2. Значение n и вещественные числа вводятся с клавиатуры.

13. Дано целое $n > 0$, за которым следует n вещественных чисел. Определить, сколько среди них кратных числу, введенному с клавиатуры. Значение n и вещественные числа также вводятся с клавиатуры.

14. Дана последовательность из N вещественных чисел. Определить, каких значений больше – положительных или отрицательных. Значение N и числа вводятся с клавиатуры.

15. Дана последовательность из N целых чисел. Найти среднее арифметическое максимального и минимального элементов последовательности. Значение N и числа вводятся с клавиатуры.

16. Дана последовательность из N целых чисел. Определить, сколько элементов последовательности находится в интервале от -1 до 45. Значение N и числа вводятся с клавиатуры.

17. Найти значения 10 членов ряда $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{6}, \ldots$.

18. Найти значения 40 членов ряда $1, \frac{3}{4}, \frac{4}{8}, \frac{5}{16}, \ldots$.

19. Дана последовательность из n целых чисел. Определить, является ли она невозрастающей. Значение n и числа вводятся с клавиатуры.

20. Дана последовательность из n целых чисел. Определить, является ли она неубывающей. Значение n и числа вводятся с клавиатуры.

21. Дана последовательность из целых чисел, которая заканчивается нулем. Определить, является ли она невозрастающей. Числа вводятся с клавиатуры.

22. Дана последовательность из целых чисел, которая заканчивается нулем. Определить, является ли она неубывающей. Числа вводятся с клавиатуры.

23. Дана последовательность из n целых чисел. Определить, является ли

она возрастающей. Значение n и числа вводятся с клавиатуры.

24. Дана последовательность из n целых чисел. Определить, является ли она убывающей. Значение n и числа вводятся с клавиатуры.

25. Дана последовательность из целых чисел, которая заканчивается нулем. Определить, является ли она возрастающей. Числа вводятся с клавиатуры.

26. Дана последовательность из целых чисел, которая заканчивается нулем. Определить, является ли она убывающей. Числа вводятся с клавиатуры.

27. Дано x – целое число. Определить, является ли оно простым. Число вводится с клавиатуры.

28. Для каждого целого числа из интервала a до b, вывести на экран все его делители. Числа a и b вводятся с клавиатуры.

29. Дано целое число x. Вывести на экран все его делители. Число вводится с клавиатуры.

30. Дано целое число x. Вывести на экран все числа, на которые оно не делится. Число вводится с клавиатуры.

Вопросы к теоретическому материалу

1. Что называется алфавитом?
2. Что входит в пробельные символы?
3. Что называется лексемой? Что относится к лексемам?
4. Что называется идентификатором?
5. Каким должен быть правильный идентификатор?
6. Назовите типы констант.
7. Каким образом описывается комментарий в языке Си?
8. Из каким элементов как правило состоит программа на языке Си?
9. Что называется директивами препроцессора? Как они объявляются?
10. Что определяет тип данных?
11. Перечислите базовые типы данных в языке Си.
12. Какой оператор позволяет вычислить размер объекта в байтах? Укажите его синтаксис.
13. Для чего нужны спецификации преобразования? Как они определяются?
14. Перечислите основные спецификации преобразования для базовых типов данных.
15. Чем различаются исполняемые и неисполняемые операторы?
16. Что называется переменной?
17. В чем разница между локальными и глобальными переменными?
18. Укажите синтаксис оператора описания переменной.
19. Какая функция используется для вывода на консоль в языке Си? Назовите ее синтаксис.
20. Какая функция используется для ввода с консоли в языке Си? Назовите ее синтаксис?
21. Чем различаются префиксная и постфиксная формы записи инкремента и декремента?
22. Перечислите операции сравнения. Укажите их синтаксис.

23. Перечислите логические операции. Укажите их синтаксис.

24. Какой оператор используется для преобразования типов? Укажите его синтаксис.

25. Укажите синтаксис условного оператора ветвлений.

26. Укажите синтаксис оператора ветвления *switch*.

27. Для чего применяется оператор *break*?

28. Что называется циклом?

29. Укажите синтаксис оператора цикла с предусловием.

30. Укажите синтаксис оператора цикла с постусловием.

31. Укажите синтаксис оператора цикла с параметром.

32. Для чего используется оператор *continue*?

ПРОЦЕСС СДАЧИ ЛАБОРАТОРНОЙ РАБОТЫ

По итогам выполнения каждой лабораторной работы студент:

1. Демонстрирует преподавателю правильно работающие программы;
2. Демонстрирует приобретенные теоретические знания, отвечая на пять вопросов по лабораторной работе;
3. Демонстрирует отчет по выполненной лабораторной работе, соответствующий всем требованиям.

Отчет по лабораторной работе оформляется по шаблону, представленному в приложении 1. Требования к отчету представлены в приложении 2.

ПРИЛОЖЕНИЕ 1. ШАБЛОН ОТЧЕТА

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ»

Институт компьютерных технологий и защиты информации Отделение СПО ИКТЗИ (Колледж информационных технологий)

ЛАБОРАТОРНАЯ РАБОТА №

по дисциплине

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

Работу выполнил

Студент гр.43__

Фамилия И.О.

Принял

Преподаватель Григорьева В.В.

Казань, 2025 г.

1. **Цель работы.**

2. **Задание на лабораторную работу** – вставляется задание на лабораторную работу, соответствующее индивидуальному, выданному преподавателем, варианту студента.

3. **Результат выполнения работы** – формируется описание хода выполнения работы и вставляются скриншоты с результатами работы разработанных программ (скриншоты должны быть подписаны, например, *Рисунок 1. Начальное состояние программы* и т.п.).

4. **Листинг программы** – вставляется код разработанной программы

ПРИЛОЖЕНИЕ 2. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА

Лист документа должен иметь книжную ориентацию, поля документа должны составлять: левое – 3 см, правое – 1,5 см, верхнее – 2 см, нижнее 2 см.

Нумерация страниц – внизу страницы по центру, первая страница не нумеруется

Междустрочный интервал – 1,5 (полуторный), отступ первой строки – 1,25.

Текст документа должен быть выполнен с использованием шрифта Times New Roman, размер – 14, выравнивание – по ширине. Заголовки выполняются тем же шрифтом, но размера 16, полужирное начертание, размещение – по центру.

Рисунки должны размещаться по центру, они нумеруются по порядку. Перед рисунком в тексте на него должна быть ссылка. Подпись рисунка должна располагаться по центру и быть выполнена шрифтом Times New Roman, размер – 12. Сначала происходит нумерация рисунка, а затем пишется его название.