МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Институт прикладной математики и компьютерных наук
Кафедра «Прикладная математика и информатика»

Утверждено на заседании кафедры
«Прикладная математика и информатика»
24 января 2022 г., протокол № 5

Заведующий кафедрой

_____М.В. Грязев

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению лабораторных работ

по дисциплине (модулю)
«Системы программирования»

основной профессиональной образовательной программы
высшего образования – программы бакалавриата

по направлению подготовки
01.03.02 Прикладная математика и информатика

с направленностью (профилем)
Прикладная математика и информатика

Форма обучения: очная

Идентификационный номер образовательной программы: 010302-01-22

Тула 2022 год

Разработчик методических указаний

Скобельцын С.А., профессор каф. ПМиИ, д.ф.-м.н. 

(ФИО, должность, ученая степень, ученое звание) *(подпись)*

Содержание

Лабораторная работа № 1

Интегрированные среды разработки (Borland Pascal 7.0.и Borland C++)

Цель работы: Изучение способов описания элементов языка. Знакомство с интегрированной средой программирования программ Borland Pascal 7.0.и Borland C++

Теоретическая справка [6, лек 2-3]

Задание

В интегрированных средах Borland Pascal 7.0.и Borland C++ подготовить приведенные ниже программы. Изучить возможности сред по отладке программ.

Программа 1

```
program Example;
uses Crt;
var i: Integer;
   s1, s2: String;
begin
 ClrScr;
 Writeln('Введите строку символов');
 Readln(s1);
 s2:='';
 for i:=Length(s1) downto 1 do s2:=s2+s1[i];
 Writeln('Результирующая строка: ', s2);
end.
```

Программа 2

```
#include <iostream.h>
#include <conio.h>
void main()
{
 clrscr();
 int i=1, j=1, k=1;
 cout << (++i || ++j && ++k) << i << j << k;
 for (double x=0, p=1; x<5; p* = ++x);
 cout << '\n' << p;
}
```

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

1. Название работы.
2. Цель работы
3. Произвести сравнение работы в этих интегрированных средах.

Лабораторная работа № 2

Интегрированные среды разработки (Borland Delphi (Дельфи))

ОСНОВЫ РАБОТЫ В СРЕДЕ BORLAND DELPHI

Общие указания

Лабораторную работу можно выполнять в среде визуального программирования Borland Delphi любой версии.

Цель работы

Изучение основ работы в интегрированной визуальной среде разработки программ Borland Delphi.

Задание

Выполнить упражнения.

Упражнение 1

Создание простейшего Windows-приложения с заданным заголовком окна и цветом формы

1. Создать **папку** для сохранения разработанных приложений.
2. Запустить Delphi.
3. Изменить **заголовок** окна формы с Form1 на ***Привет*****:** в окне инспектора объектов (Object Inspector) установить для **свойства Caption** значение ***Привет***
4. Изменить **цвет** формы со стандартного на другой: в окне инспектора объектов установить для свойства **Color** значение **clAqua**.
5. **Выполнить** приложение:

5.1. Запустить приложение — меню Run, Run или F9 или кнопка на панели инструментов.

5.2. Изменить размеры окна.
5.3. Поэкспериментировать со стандартными кнопками минимизации и максимизации окна.
5.4 Закончить работу приложения, закрыв его окно.
6. **Сохранить** форму и проект на диске: Меню File, Save All, установить свою папку, создать новую папку (с именем **1**), открыть ее, ввести имя проекта.

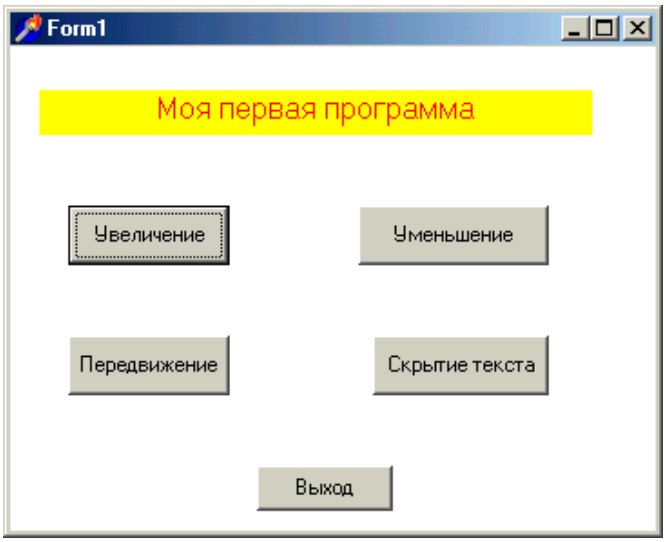
Упражнение 2

Создание Windows-приложения, которое содержит текст "Моя первая программа!" и кнопки, позволяющие изменять размер шрифта и двигать текст

1. Поместить **объект Label** в окно формы Form1.
2. **Переместить** объект Label1 на желаемое место в форме.
3. **Изменить свойства** объекта Label1: В окне инспектора объектов (Object Inspector) установить следующие значения для свойств объекта:

Объект	Свойство	Значение
Label1	Caption	Моя первая программа!
	Font	12 р., красный
	Alignment	taCenter
	Color	желтый (Yellow)
	AutoSize	False

4. **Выполнить** приложение: меню Run, Run или F9.
5. **Сохранить** форму и проект на диске: Меню File, Save All, установить свою папку, ввести имя **Приложение2**.
6. Поместить **объект Button** (командная кнопка) в окно Form1. Он по умолчанию получит имя Button1. Изменить его размеры.



7. Установить **свойство Caption** объекта Button1 в значение "Увеличение".
8. **Написать код** для события Click на объекте Button1: Два раза щелкнуть по объекту Button1 в форме. Между словами `Begin` и `End` написать следующий код:

```
Label1.Font.Size :=  Label1.Font.Size + 2;
```

9. **Выполнить** программу. Обратить внимание на то, что происходит при нажатии кнопки с надписью "Увеличение".
10. **Сохранить** форму и проект на диске: Меню File, Save.
11. **Создать объект** "командная кнопка" для уменьшения размера шрифта в тексте.
12. **Создать объект** "командная кнопка" для того, чтобы двигать текст. Код:

```
Label1.Left :=  Label1.Left + 10;
Label1.Top  :=  Label1.Top  + 10;
```

13. **Создать объект** "командная кнопка" для того, чтобы сделать текст невидимым.

Код:

```
Label1.visible :=  false;
```

14. **Создать объект** "командная кнопка" для выхода из работы программы. Код:

```
Close;
```

15. **Сохранить** форму и проект.

Упражнение 3

Создание Windows-приложения, в котором при щелчке на радио-кнопке с названием цвета на светофоре загорается соответствующий цвет

1. Поместить компоненты **Label, Panel, GroupBox, RadioButton (страница Standard)** в форму.
2. Установить следующие **свойства объектов**, используя Инспектор объектов:

Label1	Caption	Светофор
Panel1,2,3	Caption	Цвет
GroupBox1	Caption	Красный
RadioButton1	Caption	Желтый
RadioButton2	Caption	Зеленый
RadioButton3	Caption	

3. **Записать код** для процедуры обработки события Click (щелчок мыши) на объекте RadioButton1:

```
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
 Panel1.Color := clRed;
 Panel2.Color := clWhite;
 Panel3.Color := clWhite;
end;
```

Самостоятельно записать код для процедур:

```
TForm1.RadioButton2Click
```

и

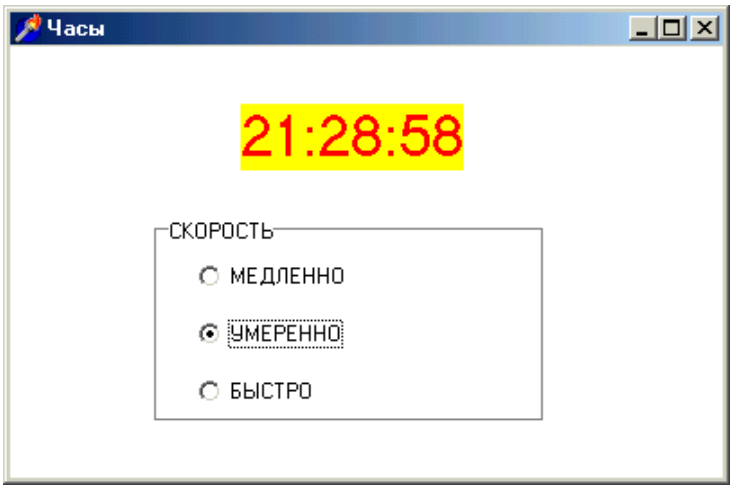
```
TForm1.RadioButton3Click
```

4. **Добавить** печать информации "Стойте", "Внимание", "Идите" на панели с соответствующим сигналом белым цветом шрифта жирным начертанием 12п.

Упражнение 4

Создание Windows-приложения, в котором работают цифровые часы с разной скоростью

1. Поместить компоненты **Label** (вкладка Standard) и **Timer** (System) в форму Form1.



2. Установить следующие **свойства** объектов

Объект	**Свойство**	**Значение**
Form1	Name	Clock
Label1	Caption	00:00:00
Label1	Color	clYellow
Label1	Font.Size	24
Label1	Font.Color	Красный

3. Записать **код обновления** времени для процедуры TClock.Timer1Timer:

```
Label1.Caption := TimeToStr(Time);
```

4. Добавление **кнопок регулирования скорости обновления времени.**

4.1. Добавить в форму компоненты **GroupBox** и **RadioButton**:

4.2. Установить следующие **свойства объектов**:

GroupBox1	Caption	Скорость
RadioButton1	Caption	Медленно
RadioButton2	Caption	Умеренно
RadioButton3	Caption	Быстро

4.3. Записать **код** для процедуры TForm1.RadioButton3Click:

```
Timer1.Interval := 1000;
```

Самостоятельно записать код для процедур:

```
TForm1.RadioButton1Click(3000)
```

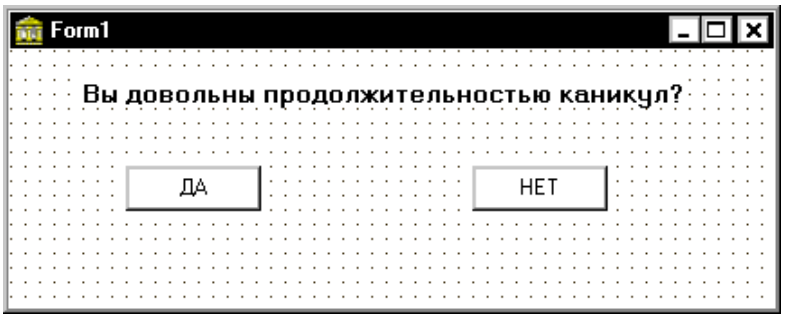
и

```
TForm1.RadioButton2Click(2000)
```

Упражнение 5

Программа-шутка

1. Поместить компоненты **Label** и **Button** в форму в соответствии с рисунком



2. Установить свойство объекта **Button2**: ***DragMode dmAutomatic***
3. Записать код для обработки события **MouseMove** на объекте **Button2**:

```
procedure TForm1.Button2MouseMove(Sender: TObject; Shift:
                    TShiftState; X,Y: Integer);
    begin
      Button2.Left := Button2.Left+10;
      Button2.Top  := Button2.Top+10;
    end;
```

4. Записать код для обработки события **Click** на объекте **Button1**:

```
    procedure TForm1.Button1Click(Sender: TObject);
    begin
      Label1.Caption := 'Мы были в этом уверены!'
    end;
```

5. Выполнить программу.
6. Развитие задачи: изменить программу так, чтобы при подводе курсора мыши к кнопке Button2 **кнопка исчезала**, а при отводе курсора — **появлялась**.

Упражнение 6

Программа с мультипликацией, видеоклипом и гиперссылкой

1. Создать в графическом редакторе Paint **новый рисунок** с размерами 500 на 100 пикселей. Вставив из файлов СОБАКА.BMP и СОБАКА 1.BMP рисунки, сформировать киноленту.

2. Разместить в форме **панель** с размерами 100x100.
3. Прямо на панель **точно** по левому краю разместить компонент **IMAGE1** с размерами 500x100 и вставить на него киноленту.

4. Разместить в форме **таймер** с интервалом 250 и для события **OnTimer** записать код

```
With  Image1  do Begin
       Left := Left-Width div 5;
       If Left = -Width Then Left:=0;
End;
```

5. Запустить и при успешной работе сохранить программу.
6. Разместить новую панель и компонент **MediaPlayer1**.
7. В процедуре создания формы записать код для проигрывания видеоклипа ***Cool.avi***

```
MediaPlayer1.Filename := 'Cool.avi';
MediaPlayer1.Open;
MediaPlayer1.Display := Panel2;
MediaPlayer1.Play;
```

8. Запустить и при успешной работе сохранить программу.
9. Разместить на форме **надпись** с текстом **гиперссылки**. Установить шрифт синего цвета с подчеркиванием. При подводе курсора сделать подсказку *"Сайт преподавателя"* и курсор в виде ручки. Записать код для события щелчка на гиперссылке, предварительно в инструкции USES добавив модуль **ShellAPI**

```
       ShellExecute(0,'Open',
'http://www.gor.h1.ru','','',SW_SHOW);
```

Упражнение 7

Построение графика отображения оценок учеников в зависимости от их номера по списку

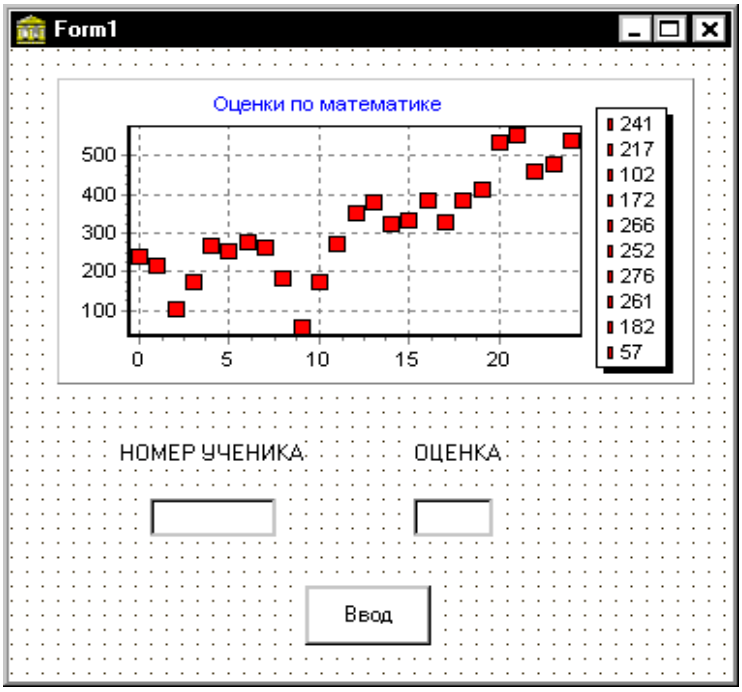
1. Вставьте в форму **компонент Chart** (страница Additional).
2. Вызовите редактор диаграмм: щелкните **правой кнопкой** мыши на объекте Chart1 и выберите опцию **Edit Chart**.
3. Щелкните по закладке **Series**, нажмите кнопку **Add** и выберите **тип графика Point** (точечный).
4. Во вкладке **Titles** (заголовки) укажите название диаграммы "Оценки по математике". Закройте редактор Edit Chart.
6. Добавьте объекты Edit, Button, Label и установите их свойства
7. Запишите **код** обработки события "щелчок" на кнопке Button1

```
Series1.AddXY (StrToFloat(Edit1.Text),
                StrToFloat(Edit2.Text),'', clRed);
```

8. Запустите программу.
9. **Развитие задачи**.

9.1. Замените точечный график на **линейный**: вызовите опцию Edit Chart. Удалите набор данных Series1, вставьте новый набор Series1 и выберите для него линейный график (Line), запустите программу.

9.2. Сделайте параллельный ввод данных для двух наборов (серий), например, для оценок **по двум предметам.**



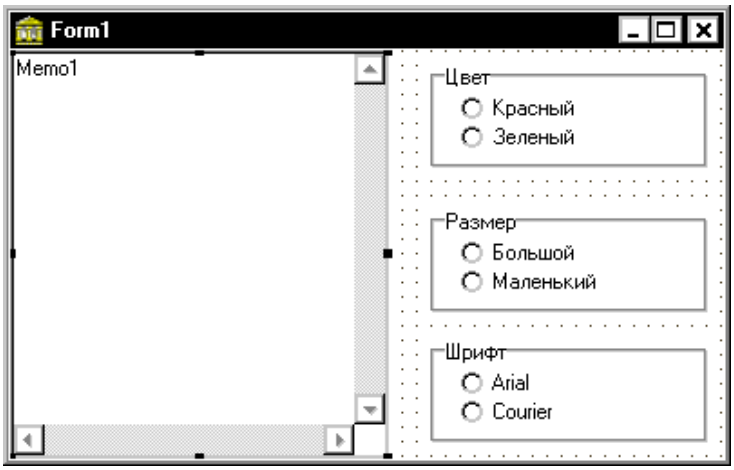
Упражнение 8

Создание текстового редактора, в который можно загрузить файл, отредактировать его и сохранить

1. Вызвать текстовый редактор **Блокнот** и создать в нем текстовый файл **My_text.txt** с содержанием:

Button - кнопка
RadioButton - переключатель
Label - надпись
Edit - строка текста
Memo - редактор текста

2. Сохранить файл в папку **MEMO**.
3. Начать новый проект и сразу сохранить его **в папке Memo**.
4. Поместить компонент **Memo** в форму и установить для свойства **ScrollBars** (линейки прокрутки) значение **ssBorth**, а для свойства **Align** (размещение) значение **alLeft** (левая часть формы).



5. Записать код для загрузки файла **при создании формы**:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  Memo1.lines.LoadFromFile('My_text.txt');
```

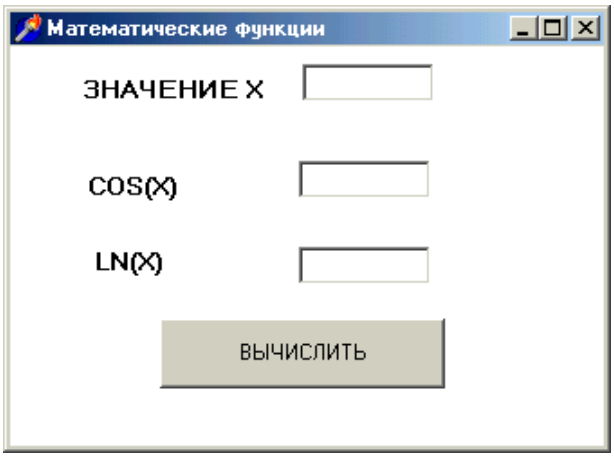
6. Записать код, позволяющий сохранить файл при закрытии формы:

```
procedure TForm1.FormClose…
begin
  Memo1.lines.SaveToFile('My_text.txt');
```

7. Запустить программу. Добавить что-нибудь в появившийся текст. Закрыть программу.
8. Запустить ее еще раз и убедиться, что загружается **откорректированный текст.**
9. Добавить группу переключателей (RadioButton) для выбора **цвета шрифта** и записать соответствующий код.
10. Добавить группу переключателей для выбора **размера** шрифта.
11. Добавить группу переключателей для выбора **вида** шрифта.

Упражнение 9

Создание Windows-приложения для вычисления различных математических функций



Ввести значение **x** и вычислить значения следующих математических функций:
Cos(x) Ln(x) Power(x,k) Sqr(x) Sqrt(x) Sin(x) Log10(x) Exp(x)
1. Поместить компоненты **Label** и **Edit** в окно формы Form1 и установить их свойства.
2. Записать код для обработки события **OnClick** на объекте Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
Var  x, y1, y2 : real;
begin
x := StrToFloat(Edit1.Text);
y1 := cos(x);
y2 := ln(x);
Edit2.text := FloatToStr(y1);
 Edit3.text := FloatToStr(y2);
    end;
```

4. Добавить возможность вычисления **остальных функций**. Учесть, что для использования функций Log10 и Power необходимо в строке Uses добавить в перечень модуль **Math**.

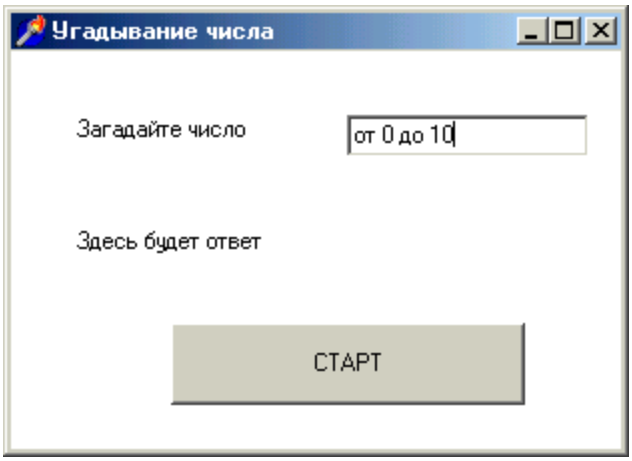
Упражнение 10

Создание Windows-приложения для вычисления случайного числа и сравнения его с числом, введенным пользователем. По результатам сравнения выдается сообщения: "Вы угадали", "Число меньше искомого", "Число больше искомого", "Число вне диапазона"

Код:

```
 var
    Form1: TForm1;    m : integer;                    {Случайное число}
 ...
 procedure TForm1.Button1Click(Sender: TObject);
 Var n : integer;  s : string;
 begin
    n := StrToInt(Edit1.Text);
    If (n < 0) Or (n > 10) Then s := 'Число вне диапазона'
     Else if n > m Then s := 'Число больше искомого'
     Else If n < m Then s := 'Число меньше искомогоl'
     Else If n = m Then s := 'Вы угадали';
    Label2.Caption := s;
 end;

 procedure TForm1.FormCreate(Sender: TObject);
 begin
    Randomize; m := Random (10);
  end;
```

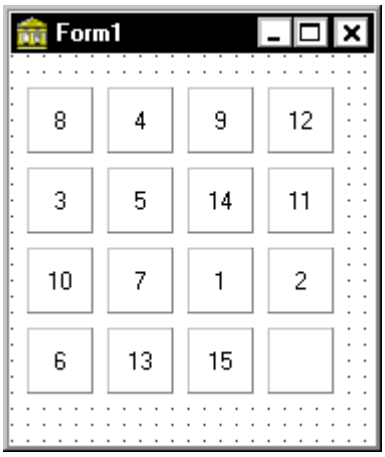


Упражнение 11

Игра "15"

На экране находятся 16 клеток по 4 в ряд. В начале игры в 15 клетках высвечено случайное число в диапазоне 1...15. Последняя клетка не заполнена. Игрок должен за минимальное количество ходов расположить числа в порядке возрастания. Каждый ход заключается в **щелчке** по одной панели с цифрами, после чего она перемещается **на пустую панель**.

1. Поместить на форму объекты Panel1,..., Panel16 и выделить их.



2. В окне Object Inspector выбрать страницу **Events**. Дважды щелкнуть на событии **OnClick**. Будет создан обработчик событий, общий для всех выделенных объектов.

3. Записать код:

```
procedure TForm1.Panel1Click(Sender: TObject);
 var T,L : integer;
 begin
    With ( Sender as TPanel ) do
       begin
       L := Panel16.Left;
       T := Panel16.Top;
       Panel16.Left := Left;
       Panel16.Top  := Top;
       Left := L;    Top  := T;      end;
 end;
```

4. Добавить подсчет **количества** сделанных игроком ходов.

Упражнение 12

Программа, которая вводит одномерный массив действительных чисел и вычисляет двумерный массив

Даны действительные числа $x_1, ..., x_8$. Получить **квадратную матрицу**

$$\begin{bmatrix} x_1 & x_2 & \dots & x_8 \\ x_1^2 & x_2^2 & \dots & x_2^8 \\ \dots & \dots & \dots & \dots \\ x_1^8 & x_2^8 & \dots & x_8^8 \end{bmatrix}$$

Для **ввода** исходного массива и **вывода** полученной матрицы использовать компоненты **StringGrid.**

Развитие задачи:

1. Ввести исходный массив из файла.
2. Полученный массив и исходную информацию вывести в **файл**.

Оформление отчета

Отчет по данной работе не оформляется. Результаты выполнения упражнений должны быть продемонстрированы преподавателю

Лабораторная работа № 3

Интегрированные среды разработки Microsoft Visual Basic

Цель работы Изучение основ работы в интегрированной визуальной среде разработки программ VBA для Exel.

Задание Выполнить упражнения 1-4.

Упражнение 1 Использование макпрорекордера

Для разработки программ используются два способа - макрорекордер и редактор *VB (Visual Basic Editor).* Инструментальное средство *Macrorecorder (Макрорекордер)* отслеживает действия пользователя, выполняемые при помощи команд меню или клавиатурных эквивалентов («горячих» клавиш), генерирует и записывает соответствующие макрокоманды до тех пор, пока пользователь не остановит процесс записи. В результате создается программа (макрос), которая дает тот же эффект, что и выполненные действия. Это позволяет автоматизировать процессы обработки данных, проводимые командами меню, неоднократно выполняя записанную программу.

Пусть требуется рассчитать размер вклада через год при условии, что первоначальная сумма составляла 10 тыс. рублей и вклад был открыт под

10,5% годовых с ежемесячным начислением процентов.

Включите запись макроса командой *Сервис-Макрос-Начать запись (Tools-Record Macro-Record New Macro).*

В поле *Имя макроса (Macro Name)* введите имя будущей процедуры. По умолчанию имя состоит из слова «Макрос» *(«Macro»)* и порядкового номера создаваемого макроса . Нажатие на *OK* включает запись.

В статусной строке появится сообщение *Запись (Recording),* а на экране возникнет панель инструментов , первая кнопка которой — *Остановить запись (Stop Macro).* Если этой панели на экране нет, то необходимо включить ее с помощью команды *Сервис-Настройка (Tools Options),* вкладка *Панели инструментов.* В списке возможных панелей инструментов пометьте *Остановить запись (Stop Macro).*

Введите данные и формулы в соответствии с правилами Exel и остановите запись макроса. Перейдите в редактор *VB,* нажав клавиши *Alt+F11.*

	A	B	C
10	S	10000	
11	T	12	
12	r	10,50%	
13	P	=B10*(1+B12/12)^B11	
14			
15			

Созданная процедура записана в модуле *Module1* с именем *Макрос1.* В ячейке *B13* стоит формула расчета нового размера вклада по формуле сложных процентов. Ссылки на ячейки с данными записаны в стиле *R^1C1* и являются относительными по отношению к ячейке *B13.* Активизируйте другой рабочий лист - *Лист2* и проверьте правильность выполнения тех же расчетов на новом листе. Для этого выполните команду *Сервис-Макрос-Макросы (Tools-Macro-Macros).*

Упражнение 2 Редактор VBA

Редактор *Visual Basic* позволяет записывать, сохранять и модифицировать программные модули, выполнять и отлаживать процедуры. В редактор *VB* можно перейти:

командой *Сервис-Макрос-Редактор Visual Basic;*

нажав кнопку ***h на панели инструментов*** *Visual Basic;*

с помощью функциональных клавиш *Alt-F11*. Возврат в документ MS Excel выполняется:

командой *Microsoft Excel* — последней командой меню *Вид;*

нажатием на кнопку **1^**стандартной панели инструментов *Visual Basic;*

клавишами *Alt-F11* или *Alt-Q*.

Документ MS Excel представляет собой рабочую книгу, содержащую рабочие листы с данными различных типов и формулами, листы диаграмм и программы для обработки данн^хх.

Программные компоненты документа (модули, процедуры, формы) объединяются в проект, который сохраняется на диске вместе с документом.

Проект на *VBA* нельзя создать независимо от документа. Каждой рабочей книге соответствует проект с именем, состоящим из двух частей: *VBAProject* и названия книги в скобках. Такое имя присваивается проекту по умолчанию.

Окна редактора VB

Основные компоненты разработки, отладки и запуска программ - это окна редактора *VB,* часть из которых рассматривается в лекции 4. *Отладка программ*.

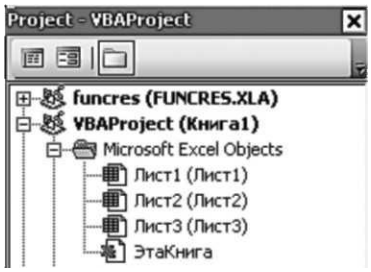
Project Window (окно проекта)

Структура проекта видна в окне проекта редактора *Visual Basic* (рис.3.1). Основное предназначение окна проекта - обеспечение навигации в проекте.

Структура проекта имеет вид дерева. Элементы структуры могут быть свернуты или раскрыты нажатием на пиктограммы «+» или «-», которые расположены слева от папок, составляющих проект.

Показать окно проекта можно при помощи:

- нажатия на кнопку стандартной панели инструментов;



. Окно проекта

команды *Project Explorer* меню *View;*

клавиш *Ctrl+R.*

Первоначально проект содержит только одну папку с объектами документа — *Microsoft Excel Objects.*

По мере разработки пользовательского приложения добавляются:

папка *Формы (Forms),* которая содержит строки для каждой формы, созданной пользователем. Это диалоговые окна пользователя — объекты *Userform;*

папка *Модули (Modules)* со стандартными пользовательскими модулями, не связанными с конкретными объектами или формами;

папка *Модули класса (Classes)* с модулями, определяющими пользовательские классы;

- папка *Ссылки (References)* со ссылками на другие проекты.

Объекты, соответствующие рабочей книге, ее листам и пользовательским формам, состоят из двух компонент — видимая часть (например, непосредственно рабочий лист) и кодовая часть (процедурный лист).

На процедурном листе, связанном с объектом, размещаются процедуры обработки событий этого объекта и дополнительно могут размещаться процедуры общего типа и процедуры-функции, необходимые для выполнения событийных процедур.

Переход на процедурный лист любого объекта проекта производится:

двойным щелчком на выбранный объект проекта;

нажатием клавиши *F7* на выделенный объект проекта;

• выбором команды *Code* из меню *View* или из контекстного меню для выделенного объекта проекта.

Стандартные модули содержат:

процедуры общего типа;

процедуры-функции, разработанные пользователем;

процедуры, записанные макрорекордером.

В дальнейшем будем использовать термин «модуль», имея в виду, что все сказанное распространяется на процедурный лист и на стандартный модуль, если иное не оговорено особо.

Дополнительно к процедурам в начале модуля могут располагаться инструкции компилятору, объявления глобальных и модульных переменных или переменных пользовательских типов.

Процедура - это последовательность команд (операторов языка), начинающаяся с оператора *Sub* и заканчивающаяся оператором *End Sub.*

Все операторы, которые заключены между этими двумя операторами, составляют тело процедуры.

После записи оператора начала процедуры вручную и нажатия клавиши *Enter* конец процедуры заисывается автоматически.

Для вставки процедуры установите точку вставки в тело модуля и выполните команду *Procedure (Процедура)* из меню *Insert (Вставка).*

ВАЖНО

Любую процедуру, расположенную в стандартном модуле, можно запускать из диалогового окна *Макрос,* содержащего перечень доступных для запуска процедур (рис. 1.5).

Процедуры, которые расположены на процедурных листах, связанных с объектами рабочей книги или пользовательскими формами, не доступны в этом окне.

Properties Window (Окно свойств)

Это окно показывает и позволяет изменять свойства объекта, выделенного в окне проекта или в окне формы. Для вызова окна свойств используйте клавишу *F4* или пиктограмму **i# .** Если выделен стандартный модуль или проект , то высвечивается его единственное свойство — *Name* (имя). Если же вы**1**делен, например, объект *Userform,* то в окне свойств показана таблица с перечнем его свойств.

Изменение свойства *Name* приводит к изменению имени объекта, под которым этот объект известен всем процедурам проекта.

Свойства объектов можно изменять в режиме конструктора *(Design time)* или в режиме выполнения процедуры *(Run time).* Некоторые свойства объектов можно менять в любом режиме, а отдельные свойства подлежат изменению только в режиме конструктора или только в режиме выполнения процедуры. Если свойства объектов меняются в режиме выполнения процедуры, то они никогда не видны в окне свойств объекта.

Code (окно программы)

Окно открывается при выполнении команды *Code (Программа)* из меню *View* или при нажатии клавиши *F7.* Можно нажать кнопку *View Code* —I (первая на панели инструментов

в окне проекта) или сделать двойной щелчок на имени модуля, объекта рабочей книги или формы.

Меню и панели инструментов Visual Basic

При переходе в редактор *VB* меняется меню и появляется стандартная панель инструментов *Visual Basic*.

Пиктограммы отладки программ расположены на отдельной панели инструментов, которая высвечивается командой *View-Toolbars-Debug.*

Упражнение 3 Работа с диалоговыми окнами ввода-вывода данных

Для ввода/вывода данных или для высвечивания служебн^хх сообщений используются встроенные функции *InputBox* и *MsgBox.*

Функции создают на экране диалоговое окно с сообщением и одной или несколькими кнопками. Программа приостанавливается до нажатия одной из кнопок. Подробно эти функции рассматриваются в разделе Встроенные функции.

Встроенная функция *InputBox* высвечивает диалоговое окно для ввода значения.

Оператор ActiveCell.FormulaRlCl = "10000" в процедуре *Макрос1* можно записать в виде ActiveCell.FormulaRlCl = lnputBox("BBeflM сумму вклада"). Тогда при выполнении процедуры возникнет диалоговое окно, в поле которого можно ввести число. После нажатия клавиши *Enter* введенное число попадает в активную ячейку.

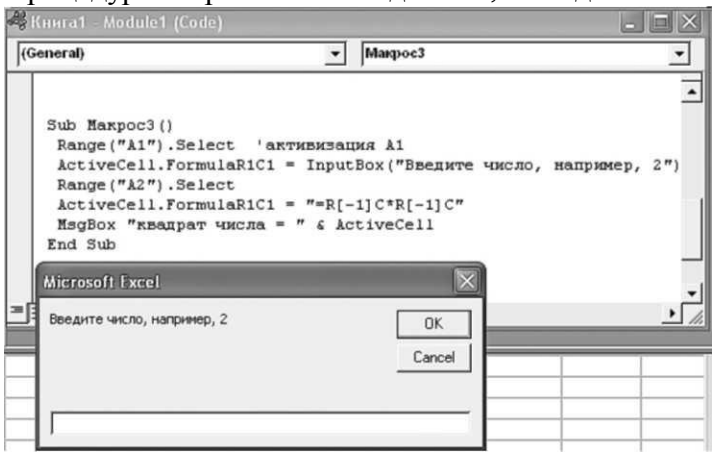
Диалоговое окно функции *InputBox*

• Если не введено никакого значения (пустой ввод) или нажата кнопка *Cancel*, то активная ячейка будет пустой, что приведет к ошибке выполнения процедуры.

Встроенная функция *MsgBox* используется для вывода сообщений. В простейшей форме записи функции *MsgBox* после ключевого слова следует только текст сообщения. Текст сообщения является строкой символов. В качестве текста сообщения могут применяться сцепленные строки символов (оператор сцепления &).

Пример

Процедура запрашивает ввод числа, выводит его квадрат.



Запрос на ввод числа функцией *InputBo*

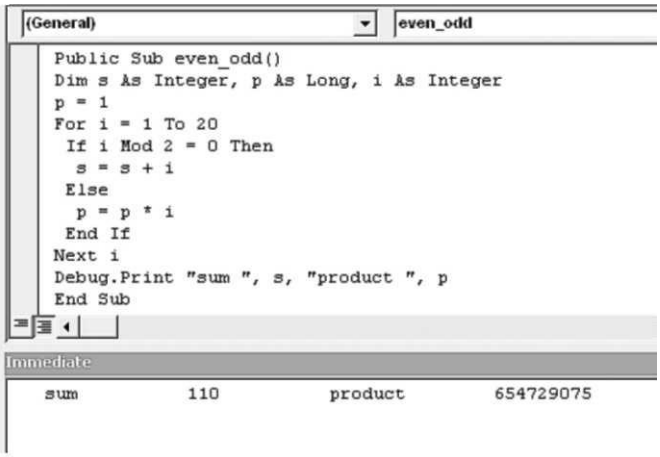
Упражнение 4 Простейшие программы

Примеры

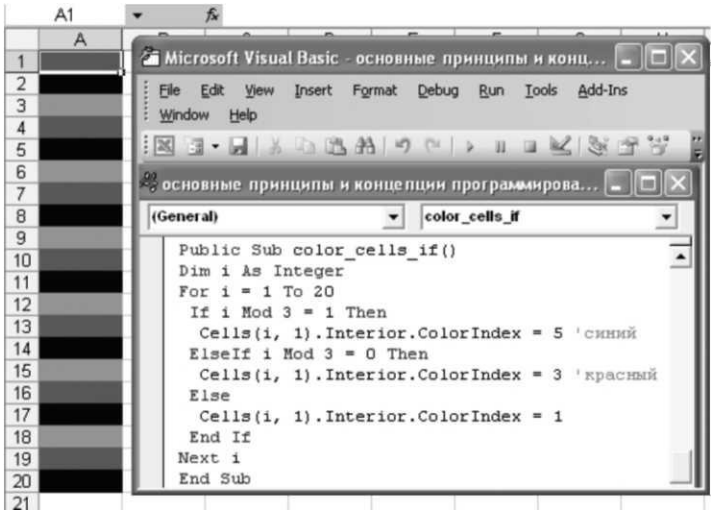
1. В^**х**числяется сумма четных чисел в интервале значений от 0 до 20 и произведение нечетных чисел в том же интервале.

Переменная*р* описана как *Long*, т.к. значение произведения превосходит число 32767. Начальное значение переменной устанавливается в 1.

В оператор цикла *For* вложен условный оператор *If*. Четность числа проверяется как нулевой остаток от деления на два.



Пример вычисления суммы и произведения четных чисел в интервале значений



Пример раскраски ячеек, номера строк которых кратны 5, 3 и 1

2. Первые двадцать ячеек столбца *A* меняют свой цвет. Если номер ячейки кратен трем, то цвет заливки красный. Для первой ячейки в тройке ячеек устанавливается синий цвет заливки. Остальные ячейки окрашиваются черным цветом (рис. 8.2). Запись *Cells(i,1)* представляет объект *Range* — ячейку первого столбца i-ой строки. Объект *Interior* — цвет заливки ячейки. Свойство *ColorIndex* этого объекта — номер цвета из цветовой палитры.

При использовании в условии числового выражения нуль интерпретируется как *False*, а любое значение, отличное от нуля, — как *True*. В условии можно проверить тип выражения или значение переменной, используя специальные функции.

Оформление отчета

Отчет по данной работе не оформляется. Результаты выполнения упражнений должны быть продемонстрированы преподавателю

Лабораторная работа № 4

Формальные способы описания языков программирования

Цель. Освоить способы описания языков программирования

Задание. Описать в виде БНФ или диаграммы Вирта конструкцию (понятие) языка С++.

Теоретическая справка [6, лек 6],[1-5]

Варианты задания:

1. Идентификаторы.
2. Число.
3. Строка.
4. Программа.
5. Раздел описаний.
6. Строковый тип.
7. Функция.
8. Структура.
9. Описание констант.
10. Описание массивов.
11. Описание переменных.
12. Описание функций.
13. Выражение.
14. Оператор присваивания.
15. Условный оператор.
16. Оператор case.
17. Оператор do…….while.
18. Оператор for.
19. Описание модуля.
20. Описание файла

Лабораторная работа № 5

Преобразование формальных грамматик.

Цель Изучить алгоритмы преобразования КС-грамматик

Задание Для заданного варианта грамматики произвести следующие преобразования:

1. устранение бесполезных символов
2. преобразование КС-грамматики с ε-правилами в эквивалентную НКС-грамматику
3. исключение цепных правил
4. удаление произвольного правила вывода
5. устранение прямой левой рекурсии
6. произвести левую факторизация
7. преобразование к нормальной форме Хомского.
8. преобразование к нормальной форме Грейбах
9. преобразование к нормальной форме Грейбах с использованием систем уравнений

Теоретическая справка [6, лек 7-9],[1-5]

Варианты заданий

№1

S→b	A->Ab	C->Bf
S→C	B->Bb	C->d
S->cCB	B->cB	
A->E	C->cA	

№2

S→aA	B→a
A→aA	B→cB
A→b	C→bAC
A→cC	

№3

S→aABC	B→вFD	E→aCE
S→aE	C→aE	E→a
A→SCD	D→aD	F→AB
A→c	B→b	

№4

S→aC	C→bA
S→bA	C→d
A→cAB	
B→aC	

№5

S→aAB A→aAB B→c

S→bA A→E
S→E B→bB

№6
S→aAB A→E
S→bBS B→dB
S→E B→E
A→cBS

№7
S→Aa A→E
S→bB B→cBdd
A→cAdA B→E
A→a

№8
S→bA A→E
A→bA B→bB
A→aB B→E

№9
S→AB B→b
A→SA B→aA
A→BB B→E
A→bA

№10
S→AC C→D
A→B C→DaC
A→AaB D→i
B→i

№11
S→1A A→C C→1c0
S→B0 B→B0 C→E
A→1A B→C

№12
S→SS A→1A0
S→1A0 A→E

№13
S→aC B→aC
S→bA C→bA
A→cAB C→d

№14
S→abSa A→baAb
S→aaAb A→b
S→b

№15

S→LA P→i F→Q(i)
S→LB A→F
L→P : = Q→i
L→Q : = B→F

№16
S→AC C→D
A→B C→DaC
A→AaB D→i
B→i

№17
S→1A B→B0
S→B0 B→C
A→1A C→1C0
A→C C→E

№18
S→T+P P→C
S→T C→1C
T→T*P C→1
T→P

№19
S→A A→1a0
S→B B→1B00
A→1A0 B→1b00

№20
S→Ba A→a
S→Ab B→Sb
A→Sa B→BBa
A→AAb B→b

№21
S→AB B→SA
S→a B→BB
A→BS B→a
A→Sb

№22
S→Ab B→bS
A→Sa B→c
A→cB

№23
S→AB B→b
A→SA B→aA
A→BB B→E
A→bB

№24

S→SaA A→ASa
S→AA A→Ad
S→b A→c

№25

S→AB B→b
A→SA B→aA
A→BB B→E
A→bB

№26

S→Aa A→E
S→bB B→cBdd
A→cAdA B→E
A→a

№27

S→SS A→1A0
S→1A0 A→E

№28

S→aC B→aC
S→bA C→bA
A→cAB C→d

№29

S→aAB A→a
S→AB B→AS
A→BBB B→b

№30

S→A A→1a0
S→B B→1B00
A→1A0 B→1b00

№31

S→Aa A→a B→E
S→bB A→E
A→cAdA B→cBdd

№32

S→abSa A→baAb
S→aaAb A→b
S→b

№33

S→AB B→b

A→SA B→aA
A→BB B→E
A→bB

№34
S→0AB A→0
S→AB B→AS
A→BBB B→1

№35

S→aP M→aP
S→bF P→bF
F→cFM P→f

№36
S→P+ A→E
S→ -F F→|F[[
A→|P[P F→E
A→ +

№37
S→SS A→aAb
S→aAb A→E

№38
S→bA A→fC ` C→aAC
A→bA B→b
A→d B→fB

№39

S→ + B→B+
S→P B→-B
S→ -PB P→ Pa
A→E P→Bb
A→ A+ P→d

№40
S→Tøp p→C
S→T C→aC
T→T1p C→a
T→p

№41
S→A0 B→0S
A→S1 B→+
A→+B

№42
S→0S1 A→a

S→01 B→bS
S→aB B→aBB
S→bA B→b
A→aS
A→bAA

№43
A→AaB B→BAa
A→BB B→Bd
A→b B→c
B→aA
№44
S→Ba A→a
S→Ab B→Sb
A→Sa B→BBa
A→AAb B→b

№45

S→A B→AB
S→B B→Ba
A→aB B→AS
A→bS B→b
A→b

№46
S→A C→a
S→B C→E
A→C D→S
A→D D→b
B→D E→S
B→E E→c
C→S E→E

№47
A→AaB B→BAa
A→BB B→Bd
A→b B→c
B→aA

№48
A→BC C→AB
A→a C→cc
B→CA C→a
B→Ab

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

4. Название работы.
5. Цель работы

6. Описание хода выполнения работы.
    1. исходная грамматика
    2. полученная грамматика после каждого из 9 преобразований.

Лабораторная работа № 6

Конечные автоматы и распознаватели

Цель Ознакомиться с правилами построения управляющих таблиц и автоматов.

Задание По своему варианту выполнить задание.

Теоретическая справка [6, лек 10],[1-5]

Варианты заданий

№1

Построить управляющую таблицу для LL(1)-грамматики с правилами

S→Ab|Bd

A→aA|E

B→cB|E

№2

Построить управляемую таблицу для LL(1)-грамматики с правилами

S→aAA|bSA|cA

A→aAS|bSS|cS|d

№3

Построить управляемую таблицу для LL(1)-грамматики с правилами

S→aR|(S)R

R→^aR|E

№4

Построить управляемую таблицу для LL(1)-грамматики с правилами

S→aAbBbS B→E

S→E C→cC

A→aBC C→E

A→bΔ

B→aB

№5

Построить управляемую таблицу для LL(1)-грамматики с правилами

S→ZC D→E

Z→ + D→.dΔ

Z→ - A→dΔ

Z→E A→E

C→dAD

№6

Постройте для заданной грамматики правый анализатор и приведите всевозможные такты его работы для входной цепочки

E→E+T

E→T
T→(E)
T→i

№7
Постройте для заданной грамматики правый анализатор и приведите всевозможные такты его работы для входной цепочки
S→aAb A→Bb
S→c B→AΔ
A→bS B→c

№8
Постройте для заданной грамматики левый анализатор и приведите всевозможные такты его работы для входной цепочки
S→aSbS
S→aS
S→c

№9
Постройте для заданной грамматики левый анализатор и приведите всевозможные такты его работы для входной цепочки
S→(AS) A→(SaΔ)
S→(b) A→(a)

№10
Построить правый и левый разбор для заданных грамматик и входных цепочек
S→bAb A→a
A→cB B→Aad

№11
Построить правый и левый разбор для заданных грамматик и входных цепочек
S →0S11
S→011

№12
Построить МП автомат по КС-грамматике G=(N,∑,P,S)
N={S,L,B}; ∑={i,=,*}
S→L=B
S→B
L→*B
L→i
B→L}

№13
Построить МП автомат по грамматике
N={S,T,P}; ∑={i,¬,^,v,(,)}

P={S→SvT
S→T
T→T^P
T→P
P→ ¬P
P→i
P→(S)}

№14

Построить МП автомат по КС-грамматике

N={S,A,B,C,P,Q} ∑={m,p,q,x,y}

P={S→AB A→m C→E Q→qQ
S→PQx B→bC P→pP Q→E
A→xy C→bbC P→E

№15

Построить МП автомат по грамматике G={N,∑,P,S}

N={S,A} ∑={a,b,c}

P={S→SaA
S→AΔ
S→в
A→ASa
A→Ad
A→c}

№16

Построить ДМП - преобразователь осуществляющий перевод произвольной цепочки из множества $\{a^n b^m c^n\}$, где n>0, m≥0} в цепочку вида 1^{n+m}

№17

Постройте ДМП преобразователь, осуществляющий перевод произвольной цепочки, состоящей из нулей и единиц в цепочку вида $1^n 0^m$, где n и m - соответственно число единиц и нулей в данной цепочке.

№18

Постройте ДМП преобразователь осуществляющий перевод цепочки b_i в цепочку $(bi+1)^2$, где bi - цепочка из нулей и единиц, являющаяся двоичным представителем числа i

№19

Постройте ДМП-преобразователь осуществляющий перевод произвольной цепочки из множества
$\{1^n 0^m, n,m>0\}$ в цепочку вида $0^n 1^{2n}$

№20

Построить ДМП преобразователь, осуществляющий перевод произвольной цепочки из множества
$\{1^{3n+2} 0^n, n\geq 0\}$, в цепочку вида $1^n 0^n$

№21

Построить ДМП преобразователь осуществляющий перевод цепочки из множества $\{1^m 0^n, m,n>0, m\neq n\}$ в цепочку вида 1^{m-n} , если m>n или в цепочку 1^{n-m} , если n>m

№22

Построить ДМП преобразователь осуществляющий перевод произвольной цепочки из множества

$\{0^n 1^n, n>0\} \cup \{1^n 0^n, n>0\}$ в цепочку вида $1^{2n}\ 0^{2n}$

№23

Построить ДМП преобразователь осуществляющий перевод произвольной цепочки из множества

$\{0^n\ 1^n, n>0\}$ в цепочку вида a^{2n}

№24

Построить ДМП преобразователь осуществляющий перевод произвольной цепочки из множества

$\{ 0^n 1^n 0^m 1^m. . . ,$ где $n,m>0\}$ в цепочку вида $1^{n+m+...}$

№25

Построить ДМП преобразователь. Осуществляющий перевод произвольной цепочки из множества

$\{ a^n b^m c^m d^n,$ где $n>0, m\geq 0\}$ в цепочку вида $1^n 0^{m+n}$

№26

Построить расширенный МП автомат по грамматике

N={S,A,L} ∑={a,b,(,)}

P={S→bAb

A→(L

A→a

L→Aa) }

№27

Построить расширенный МП автомат по грамматике

N={S,D,R,X,Y} ∑={begin, end, d,r, ; , _ , ,}

P={S→ begin _ D; R end

D→dX Y→_ , чY

X→_dX Y→E

X→E

R→rY

№28

Построить расширенный МП автомат по грамматике

N={S,T,P,C} ∑={+, x, /, .}

P={S→S+T C→/C

S→T C→/

T→T*P

T→P

P→. T .

№29

Построить расширенный МП автомат по грамматике

N={S,A,B} ∑={a,b,c,d}
P={S→Aa A→E
S→bB B→cBdd
A→cAdA B→E }
A→a

№30
Постройте ДМП автомат распознающий цепочки из множества $\{a^n b^m c^n \mid n > 0, m \geq 0\}$

№31
Постройте ДМП автомат распознающий цепочки из множества $\{ a^n b^m c^m d^n \mid n > 0, m \geq 0 \}$

№32
Постройте ДМП автомат, распознающий цепочки в алфавите {0,1} с одинаковым количеством нулей и единиц.

№33
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 1^n 0^m 1^m \ldots \mid n, m. . > 0\}$

№34
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 1^n \mid n > 0\}$

№35
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 1^n \mid n>0\} \cup \{1^n 0^n \mid n>0\}$

№36
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 10^n \mid n > 0\}$

№37
Построить ДМП автомат, распознающий цепочки из множества $\{1^{3n+2} 0^n \mid n \geq 0\}$

№38
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 1^m \mid n > m > 0\}$

№39
Построить ДМП автомат, распознающий цепочки из множества $\{0^n 1^m \mid n \geq m > 0\}$

№40
Построить расширенный МП автомат по грамматике
N={S,L,B} ∑={i,+,*}
P={S→L+B
S→B
L→*B
L→i
B→L}

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

7. Название работы.
8. Цель работы
9. Описание хода выполнения работы.

Лабораторная работа № 9

Лексический анализ

Цель Ознакомиться с правилами построения лексического анализатора и работающих с ним таблиц.

Задание. Написать лексический анализатор для фрагмента программы по вашему варианту (на языке Паскаль или Си++). Недостающие объявления дописать. Вставить в текст комментарии. В результате данный текст должен быть переведен в лексемы и соответствующие таблицы должны быть построены.

Теоретическая справка [6, лек 11],[1-5]

Варианты заданий

№1

```
int main ()
{ int x
for (x=0; x<=100; x++)
{ if (x%2) continue;
Cout <<x<< ‘ ‘ ;
}
return 0;
}
```

№2

```
#include<iostream>
using namespace std;
int main ()
{ char let
for (let=’Z’; let>=’A’; let--)
cout<<let;
return 0;
}
```

№3

```
begin
  sum:=0;
for i:=0 to n do
  begin
    sum:=sum+A[i]*A[i+2];
```

```
end;
end.
```

№4

```
begin
clrscr;
top i:=nil;
for i:=1 to 10 do push(i);
end.
```

№5

```
for i:=filesize(f)-1 downto 1 do
for j:=0 to i-1 do
begin
seek(f,j);
read(f,X,Y);
if X>Y then
begin
seek(f,j);
write(f,Y,X);
end;
end;
close(f);
```

№6

```
#include<iostream>
#include<cmath>
{ int main ()
int num;
double sq_r;
for (num=1; num<100; num++) {
sq_r=sqrt((double)num);
cout<<num<<” ”<<sq_r<<’\n’;
}
return 0;
```

```
}
```

№7

```
int main ()
{ int n
do {
cout<<”Число ≠100:”;
cin>>n
} while (n:=100);
return 0;
}
```

№8

```
function Fact (i:integer):longint;
begin
if i=1 then Fact:=1
else Fact:=i*Fact(i-1)
end;
```

№9

```
bool password()
{ char s[80];
cout<<”Введите пароль:”;
gets(s);
if  (!strcmp(s,”пароль”))
return false;
} return true;
}
```

№10

```
begin
i:=nk
while (i>0) and (Name<>kw[i].word) do
i:=i-1;
if i>0 then
Testkw:=kw[i].lex
```

```
else
Testkw:= lexName;
end;
```

№11

```
if eof(f) then
ch:=chEoT
else if eoln(f) then begin
line:=line+1;
pos:=0;
ch:=chEoL;
end
else begin
read(f,ch);
pos=pos+1;
end
```

№12

```
int main ()
{
int t,i,n[3][4];
for(t=0; t<3; ++t) {
for(i=0; i<3; ++i) {
n[t][i]=(t*4)+i+1;
cout<< n[t][i]<<’ ’;
}
cout<<’\n’;
}
return 0;
}
```

№13

```
s:=0;
for i:=1 to n do
s:=s+fa(i);
```

№14

```
begin
i:=0;
repeat
if i<NL then begin
i:=i+1;
N[i]:=ch;
end
else Error (‘Ошибка’);
Next ch
until not ( ch in [‘A’..’Z’]);
end.
```

№15

```
if eof(f) then
ch:=chEoT
else if eoln(f) then begin
readln(f);
end
```

№16

```
type Tint=1…100
var PX: ^char
    PY: ^Tint
begin
New(PX); New(PY);
PX^:=’*’;
PY^”=3;
Dispose(PX);
Dispose(PY);
end.
```

№17

```
procedure GC (C:integer);
begin
Gen(abs(c));
if C>0 then
Gen(cmNeq);
end;
```

№18

```
int main ()
{ int i,l
for(t=2; t<1000; i++) {
for(j=2; j<=(i/j); j++)
if(!(i%j)) break;
if (j>(i/j)) cout<<i<<”-простое число\n”;
}
return 0;
}
```

№19

```
#include<iostream>
#incluse<cstdlib>
using namespace std;
int main()
{ int magic;
  int guess;
magic=rand()
cin>>guess;
if (guess==magic) {
cout<<”**да**\n”;}
else
cout<<”вы ошиблись”;
return 0;
}
```

№20

```
type Rec=record
A: char
B: byte
end;
var X:Rec;
begin
X.A:=’*’;
X.B:=7;
end.
```

№21

```
begin
Bottom^.Name:=Name;
Obj;Top;
while Obj^.Name<>Name do
Obj:=Obj^.Prev;
```

№22

```
S:=0;
i:=1;
repeat
S:=S+A[i]*B+C*A[i+1];
i:=i+1;
until i>n
```

№23

```
procedure Vect_Max;
var Max: integer;
i: byte;
begin
MAX:=Vect[1];
for i:=2 to N do
if Vect[i]>Max then
Max:=Vect[i];
end;
```

№24

```
type T_Rec=record
A: record
B: integer
C: real
end
var Rec: T_Rec
with Rec do begin
A.B:=1;
A.C:=3.14
end
```

№25

```
int main()
{ int x;
for (x=0; x<5; x++) {
if (x==1 )cout<<”1\n”
else if (x==2 )cout<<”2\n”
else if (x==3 )cout<<”3\n”
else if (x==4 )cout<<”4\n”
else cout<<”Ошибка\n”; }
return 0;
}
```

№26

```
const m=30; b=50
var A:array[1..m, 1..n] of real;
begin
for i:=1 to m do begin
for j:=1 to n do
read (A[i,j]);
readln;
end;
end.
```

№27

```
tObj*p;
while ((p=Top)→Cat:=CatGuard) {
Top=Top→Prev;
free(p);
```

№28

```
#include<iostream>
using namespace std;
int main ()
{ int i
for (i='1'; i>=100; i++)
i<<i<<" ";
return 0;
}
```

№29

```
int main ()
{
unsigned char ch;
ch:=32;
while(ch)
{ ch++ }
return 0;
}
```

№30

```
type Vect=array [1..3] of Byte;
var x: Vect
i: byte
begin
for i:=1 to 3 do
read(x[i]);
end.
```

№31

```
procedure Error (M:string);
```

```
var EL:= line;
begin
while ( ch<>chEoL ) and ( ch<>chEoT) do
NextCh;
end.
```

№32

```
begin
while ch in ( chS, chT, chE ) do Next ch;
```

№33

```
#include<iostream>
using namespace std;
int main ()
{ int a, b;
cout<<”Введите число”ж
cin>>a;
if (a>b) cout<<”a<b”;
return 0;
}
```

№34

```
begin
statment;
while Lex=LexSemi do begin
NextLex;
statement;
end;
end;
```

№35

```
if ( x→Cat= - CatVar)
AssStat ();
else if ( x→Cat= =CatStProc&&x→Typ==TypWoNe )
CallStat (x→Val) else
Expected(“ Обозначения переменной ”);
```

№36

```
switch(ch) {
case‘;’:
```

```
Next Ch (); lex=lexSemi;
break;
case’.’:
NextCh (); lexDot;
break;
}
```

№37

```
S:= 0;
i:= 1;
while i< = n do
begin
S:=S+A[i]*fa(i);
i:=i+1;
end
```

№38

```
switch(ch) {
case 0 cout<<”<1\n”;
case 1 cout<<”<2\n”;
}
return 0;
```

№39

```
begin
while Top<> nil do
begin
pop( Value );
writeln( ‘Value=’, Value:5:2 );
end;
end.
```

№40

```
begin
assign ( Finp,’F.dat’);
reset(Finp);
```

```
for i:=1 to m do
begin
for j:=1 to n do
read (Finp, A[I,j]);
readln(Finp);
end;
```

№41

```
for i:= 1 to m do
begin
B:=A[i,j];
A[i,j]:= A[i, n-j+1];
A[i, n-j+1];=B
end;
```

№42

```
Case Ch of
‘A’…’Z’, ’a’…’z’: Ident;
‘;’ begin
NextCh;
Lex:=LexSemi;
end;
```

№43

```
if Ch < > ChTab then begin
write(Ch);
pos:=pos*2+pos1+1;
end
```

№44

```
int main ( )
{ int *i, j [10];
double *i, g[10];
int x;
```

```
i=j;
f=g;
for (x=0; x<10; x++)
cout<<i+x;
return 0;
}
```

№45

```
static tLex TestKW(void) {
int i
for (i=nk-1; i>=0; i--)
if ( strcmp (Name, KWTable[i], Word ) = =0 )
return KWTable[i].Lex;
return LexName;
}
```

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

- Название работы.
- Цель работы
- Описание хода выполнения работы:
    1. Исходный текст
    2. Текст после дооформления
    3. функциональную схему программы
    4. листинг программы
    5. результат работы- массив найденных лексем, таблицу ключевых слов, таблицу символов

Лабораторная работа № 7

Восходящий синтаксический анализ (SLR-анализатор)

Цель Ознакомиться с правилами построения SLR анализатора.

Задание. Построить SRL-анализатор для своего **варианта грамматики из лабораторной работы №5** (или) показать, что для данной грамматики это нельзя сделать. При необходимости исправить грамматику так, чтобы анализатор можно было построить.

Теоретическая справка [6, лек 13],[1-5]

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

- Название работы.
- Цель работы
- Описание хода выполнения работы:
    1. Вариант задания
    2. уравляющую таблицу
    3. исправленный вариант задания
    4. управляющую таблицу
    5. потактовую работу анализатора

Лабораторная работа № 8

Перевод выражения в польскую запись

Цель Ознакомиться со способами перевода арифметического выражения в польскую запись

Задание. Написать программу, которая переводит выражение (**вариант взять у преподавателя)** в польскую запись. Перевод должен быть осуществлен либо с помощью построения дерева (рекурсивный алгоритм) (оценивается d баллами), либо по алгоритму Дейкстры (оценивается в d/2 , баллов).

Теоретическая справка [6, лек 16],[1-5]

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

- Название работы.
- Цель работы
- Описание хода выполнения работы:
    1. Вариант задания
    2. функциональную схему программы
    3. листинг программы
    4. Контрольный пример (печать содержимого любого из узлов дерева должна быть предусмотрена)

Лабораторная работа № 10

Синтаксический и контекстный анализ

Цель Ознакомиться с правилами построения синтаксического и контекстного анализатора и работающих с ним таблиц.

Задание. Написать в добавление к лексическому анализатору для фрагмента программы, **по вашему варианту из лабораторной работы №7,** синтаксический и контекстный анализатор. Программа должна определять синтаксические ошибки, строить необходимые таблицы и работать сними.

Теоретическая справка [6, лек 13-14],[1-5]

Оформление отчета

Отчет оформляется на стандартных листа формата А4 . В нем необходимо привести:

- Название работы.
- Цель работы
- Описание хода выполнения работы:
    1. Вариант задания
    2. функциональную схему программы
    3. листинг программы
    4. Контрольный пример

Библиографический список

1. Опалева, Э.А. Языки программирования и методы трансляции : учеб. пособие для вузов / Э.А.Опалева, В.П.Самойленко .— СПб. : БХВ-Петербург, 2005 .— 480 с.
2. С.З.Свердлов Языки программирования и методы трансляции. СПб.-Изд-во «Питер»-2007.- 924 с.
3. Ахо, А. Компиляторы: принципы, технологии, инструменты / А.Ахо, Р.Сети, Д.Ульман.— М.и др. : Вильямс, 2001,2008 .— 768 с.
4. Марченко, А.И. Программирование в среде Turbo Pascal 7.0 / А.И.Марченко, Л.А.Марченко; Под ред. Тарасенко В.П. — 6-е изд., стер., юбил. — Киев : Век, 2000, 2004.— 464 с.
5. Шилдт, Schildt G. C+ : базовый курс / Г.Шилдт; пер.с англ.и ред. Н.М.Ручко .— 4-е изд. — М. : Вильямс, 2005,2007 .— 624 с.
6. Родионова Г.А. Системы программирования. Конспект лекций.- Изд-во ТулГУ, 2011, (электронный вариант).
7. Родионова Г.А. Системы программирования .Методические указания к самостоятельной работе студентов. - Изд-во ТулГУ, 2011, (электронный вариант).
8. Белоусова С.Н.Основные принципы и концепции программирования на языке VBA в Excel: Учебное пособие / С.Н. Белоусова, И.А. Бессонова — М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2010. — 200 с.: ил., табл. — (Основы информационных технологий