МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Рубцовский институт (филиал) федерального государственного бюджетного образовательного учреждения высшего образования «Алтайский государственный университет»

Утверждено решением Ученого совета Рубцовского института (филиала) АлтГУ протокол №1 от 20.09.2024 г.

РАБОЧАС ПРОГРАММА ДИСЦИПЛИНЫ «Основы алгоритмизации и программирование»

ПРОГРАММЫ ПРОФЕССИОНАЛЬНОЙ ПЕРЕПОДГОТОВКИ

«ПЕДАГОГИКА И МЕТОДИКА ПРЕПОДАВАНИЯ ИНФОРМАТИКИ И ИКТ В ШКОЛЕ»

Рубцовск
2024

Программа рассмотрена и одобрена на заседании Ученого совета Рубцовского института (филиала) АлтГУ от 20.09.2024 г., протокол № 1.

Председатель методической комиссии института:

Заместитель директора по учебной работе _____ О. Г. Голева

Руководитель центра:

Преподаватель _____ И.С. Краснослободцева

Разработчик:

преподаватель кафедры математики и прикладной информатики _____ Е.И. Кирибаев

Содержание

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью реализации программы является: научить слушателей максимально использовать языки программирования, строить логически правильные и эффективные программы.

2. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

В результате освоения модуля слушатели должны знать:

– общие принципы построения алгоритмов, основные алгоритмические конструкции;

– понятие системы программирования;

– основные элементы процедурного языка программирования, структуру

– программы, операторы и операции, управляющие структуры, структуры

– данных, файлы, кассы памяти;

– подпрограммы, составление библиотек программ;

должен **уметь**:

– использовать языки программирования, строить логически правильные и эффективные программы

3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

3.1. Учебно-тематический план

№ п/п	Наименование разделов и тем	Максимальная нагрузка слушателей, час.	Количество аудиторных часов			Самостоятельная работа слушателей, час.
			Лекции	Практические (семинарские) занятия	Лабораторные работы	
1	**2**	**3**	**4**	**5**	**6**	**7**
Раздел 1. Понятие алгоритмизации						
Раздел 1	Тема 1. Понятие алгоритмизации, свойства алгоритмов, общие принципы построения алгоритмов. Основные алгоритмические конструкции	10	2			8
	Тема 2. Эволюция языков программирования	8				8
Раздел 2. Составление программ						
Раздел 2	Тема 3. Составление программ на алгоритмическом языке	10			2	8
	Тема 4. Операторы и операции	10			2	8
	Тема 5. Подпрограммы. Составление библиотек подпрограмм	10			2	8
ИТОГО		48	2		6	40

4. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература

1. Черпаков, И. В. Основы программирования : учебник и практикум для вузов / И. В. Черпаков. — 2-е изд. — Москва : Издательство Юрайт, 2024. — 196 с. — (Высшее образование). — ISBN 978-5-534-18759-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/545506.

2. Якимов, С. П. Структурное программирование : учебное пособие для вузов / С. П. Якимов. — Москва : Издательство Юрайт, 2024. — 342 с. — (Высшее образование). — ISBN 978-5-534-14885-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/544321.

Дополнительная литература

1. Трофимов, В. В. Алгоритмизация и программирование : учебник для вузов / В. В. Трофимов, Т. А. Павловская. — 4-е изд. — Москва : Издательство Юрайт, 2024. — 108 с. — (Высшее образование). — ISBN 978-5-534-20430-8. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/558138.

Базы данных, Интернет-ресурсы, информационно-справочные и поисковые системы

1. Электронная библиотечная система «Консультант студента» [Электронный ресурс]. – М.: ООО «Политехресурс». – Режим доступа: http://www.studentlibrary.ru/.

2. Электронно-библиотечная система «Университетская библиотека On-line» [Электронный ресурс]. – М.: Издательство «Директ-Медиа». – Режим доступа: http://www.biblioclub.ru.

3. Электронная библиотечная система Алтайского государственного университета [Электронный ресурс]. – Барнаул. – Режим доступа: http://elibrary.asu.ru/.

4. Образовательная платформа «Юрайт» [Электронный ресурс]. – М.: ООО «Электронное изд-во Юрайт». – Режим доступа: https://www.biblioonline.ru/about.

5. Электронно-библиотечная система «Znanium.com» [Электронный

ресурс]. – М.: ООО «Научно-издательский центр Инфра-М». – Режим доступа: http://znanium.com/.

6. Поисковые системы: Google, Yandex, Rambler.

7. Научная электронная библиотека eLIBRARY.RU [Электронный ресурс]: информационно-аналитический портал в области науки, технологии, медицины и образования. – М.: ООО Научная электронная библиотека. – Режим доступа: https://elibrary.ru/projects/subscription/rus_titles_open.asp.

8. Электронно-библиотечная система Издательство «Лань» [Электронный ресурс]. – СПб.: Издательство Лань. – Режим доступа: https://e.lanbook.com/.

5. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Учебные аудитории для проведения занятий всех видов (дисциплинарной подготовки); групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации. Для самостоятельной работы и подготовки к занятиям используются помещения, оснащенные компьютерной техникой с доступом к сети «Интернет» и электронной информационно образовательной среде института.

Специальные аудитории укомплектованы специализированной мебелью и техническими средствами обучения, служащими для предоставления учебной информации. Лабораторные занятия проводятся в компьютерных классах, а также в кабинете программирования и баз данных.

Требования к программному обеспечению учебного процесса:

- Windows 7 Professional Service Pack 1;
- Microsoft Office Professional Plus 2010;
- 7-Zip;
- Windows 10 Education;
- Foxit Reader;

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

1. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ, НЕОБХОДИМЫЕ ДЛЯ ОЦЕНКИ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

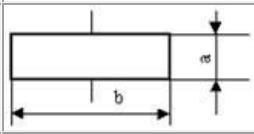
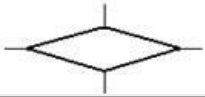
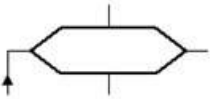
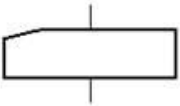
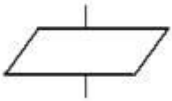
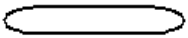
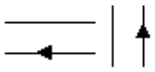
ЗАДАНИЯ К ЗАЧЕТУ/ЭКЗАМЕНУ

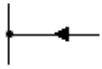
Перечень заданий /вопросов
1. Вопрос (Вопросы) для проверки уровня обученности ЗНАТЬ*
1. Основные свойства и способы представления алгоритма. 2. Структура программы Паскаль и типы данных. 3. Операторы ввода – вывода.
2. Вопрос (Вопросы) для проверки уровня обученности УМЕТЬ*
1. Условный оператор IF. 2. Оператор выбора CASE. 3. Цикл с известным числом повторений. 4. Цикл с предусловием. 5. Цикл с постусловием. 6. Одномерные массивы. 7. Двумерные массивы (матрицы). 8. Множества. 9. Модуль CRT. 10. Символы и строки, процедуры и функции для работы со строками. 11. Записи, массивы от записей. 12. Текстовые файлы. Типизированные файлы. Нетипизированные файлы. 13. Процедуры и работа с ними. 14. Функции и работа с ними.

Лабораторная работа 1.

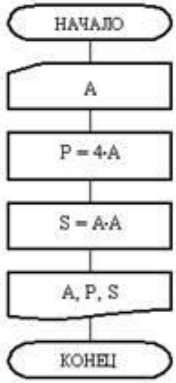
1-ый этап:

В таблице приведены наиболее часто используемые блоки, изображены элементы связей между ними и дано краткое пояснение к ним.

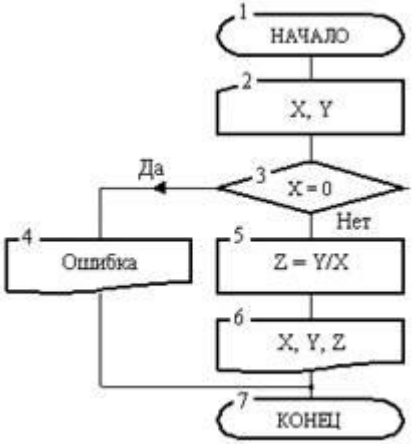
Название	Элемент	Комментарий
Процесс		Вычислительное действие или последовательность вычислительных действий
Решение		Проверка условия
Модификация		Заголовок цикла
Предопределенный процесс		Обращение к процедуре
Документ		Вывод данных, печать данных
Перфокарта		Ввод данных
Ввод/Вывод		Ввод/Вывод данных
Соединитель		Разрыв линии потока
Начало, Конец		Начало, конец, пуск, останов, вход и выход во вспомогательных алгоритмах
Комментарий		Используется для размещения надписей
Горизонтальные и вертикальные потоки		Линии связей между блоками, направление потоков

Слияние		Слияние линий потоков
Межстраничный соединитель		Нет

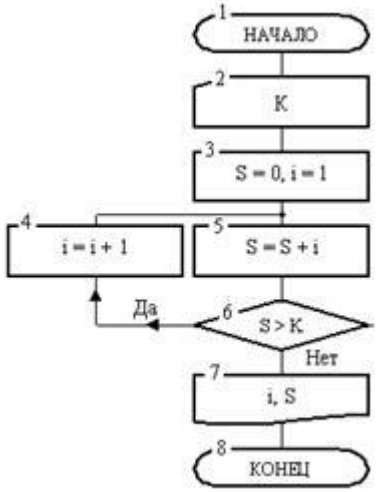
Линейный алгоритм

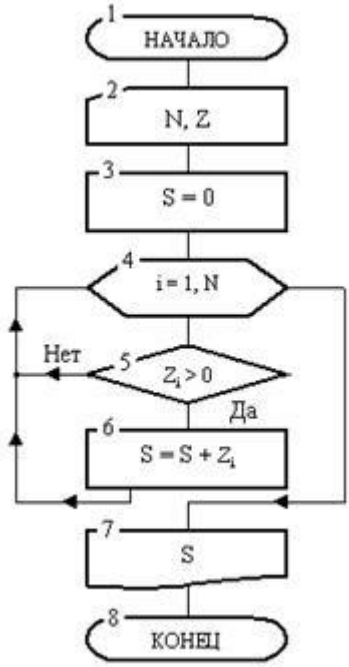


Разветвляющийся алгоритм



Циклический алгоритм





2-ой этап:

1. Запустить Паскаль.
2. Набрать текст программы для вычисления объема шара $V = 4/3ПR^3$, где

R – радиус шара, П = 3,14:

PROGRAM SHAR;
{Вычисление объема шара} комментарий,
VAR
R, V : REAL;
BEGIN
WriteLn (‘Введите радиус шара R’); оператор выводит строку на экран,
ReadLn (R); вводим нужное число,
V := 4/3*PI*R*R*R; оператор присваивания,

WriteLn (‘Объем шара = ‘, V:8:3); оператор выводит результат на экран

End.

3. Откомпилировать программу,
4. Запустить программу на выполнение,
5. Просмотреть полученный результат,
6. Сохранить файл под именем **«Объем»**

1. Самостоятельно написать программу для вычисления **площади (S)** и **периметра (P)** прямоугольника, полученный результат вывести на экран.
2. Откомпилировать программу,
3. Запустить программу на выполнение,
4. Просмотреть полученный результат,
5. Сохранить файл под именем **«Прямоугольник»**

Лабораторная работа 2

1. *Ввести радиус окружности R. Вычислить длину окружности и площадь круга с радиусом R.*

```
Var
R: Integer;
S,L : Real;
Begin
    Write('Введите радиус окружности R');
    Readln(R);
    L:= 2 * pi * R;
    S:= pi * R * R;
    Writeln('Длина окружности L = ',L);
Writeln('Площадь круга S = ',S);
End.
```

Структура составного условного оператора

```
IF <условие>        THEN
Begin
<группа операторов 1>
End
```

```
ELSE
Begin
< группа операторов 2>
End;
<следующий оператор>;
```

В операторные скобки **Begin End** можно заключить любое ко-личество операторов. Знак "точка с запятой" не ставится перед служебным словом **ELSE**, но операторы в группах, естественно, отделяются друг от друга этим знаком.

Примеры выполнения работы:

Условие: Ввести X. Вычислить значение Y

$$Y = \begin{cases} \text{SIN X , если X>1} \\ \text{COS X , если X<=1} \end{cases}$$

Программа:

```
Var
      x, y: Real;
Begin
   Readln(x);
   If x>1 Then
        Y:=Sin(x)
     Else
        Y:=Cos(x);
   Writeln ('Значение Y=', Y);
   End.
```

Условие: Ввести два числа. Если первое число больше, то оба увеличить в 3 раза, в противном случае оба числа обнулить.

```
Var  x, y: Integer;
Begin
      Readln(x,y);
      If x>y Then
            Begin
                  X:=X*3;   Y:=Y*3;
            End
      Else
            Begin
                  X:=0;   Y:=0;
            End;

          Writeln(x,y);
   End.
```

Очень часто приходиться выбирать путь решения задачи не из двух, а из нескольких возможных вариантов («выбор» или «выбор-иначе»). В Паскале такой вариант можно реализовать с помощью нескольких условных операторов.

```
IF <Условие1> THEN P1
   ELSE
      IF <Условие2> THEN P2
                        ELSE
                           IF <Условие 3> THEN P3      ELSE P4;
```

Алгоритм работы такой конструкции следующий:

- если **Условие1** истинно, то выполняется оператор **P1** (или блок операторов), следующих за конструкцией **THEN**, в противном случае этот блок пропускается;
- если **Условие1** ложно, то анализируется **Условие2**, следующее за **ELSE IF**.
- если оно истинно, то выполняется оператор **P2** (или блок операторов), следующих за **THEN**, а остальные операторы пропускаются.
- операторы, следующие за последним **ELSE**, выполняются лишь в том случае, если ложны все условия в конструкциях **IF**.

Любая встретившаяся часть **ELSE** принадлежит ближайшей к ней части **THEN** условного оператора. Условные операторы **IF** в такой конструкции называются вложенными.

Пример выполнения работы:

Условие: Вычислить значение функции Y

$$Y=\begin{cases}\sin X, & \text{если } X<0\\ \cos X, & \text{если } 0\le X\le 1\\ \operatorname{tg} X, & \text{если } X>1\end{cases}$$

Программа:

```
Var   X,Y: Real;
Begin
Readln(X);
If X<0 Then
     Y:=SIN(X)
     Else If X>1 Then
       Y:=SIN(X)/COS(X)
            Else
            Y:=COS(X);
Writeln('Значение функции Y=',Y);
End.
```

Оператор выбора CASE может быть использован вместо условного оператора, если требуется сделать выбор более чем из двух возможностей.

Оператор выбора CASE позволяет выбрать одно из нескольких возможных продолжений программы. Параметром, по которому осуществляется выбор, служит селекторное выражение – выражение порядкового типа (целочисленный, логический, символьный).

```
CASE <селекторное выражение> OF
    диапазон_1     : оператор_1;
    диапазон_2     : оператор_2;
    ...
    диапазон_N     : оператор_N;
    ELSE оператор;
END;
Следующий оператор;
```

Оператор выбора выполняется следующим образом.

- сначала вычисляется селекторное выражение;
- затем ищется тот диапазон, в который входит это значение и выполняется тот оператор, который соответствует этому диапазону. Остальные операторы игнорируются.
- Если значение селектора не вошло ни в один из приведенных диапазонов, то выполняется оператор, который записан после слова ELSE;
- после этого происходит выход из оператора CASE на следующий оператор;
- если ветвь ELSE отсутствует, то управление передает-ся следующему за CASE оператору.

Примеры выполнения работы:

Условие:	Программа:

Составить программу, которая анализирует человека по возрасту и относит к одной из четырех групп: дошкольник, ученик, работник, пенсионер.	**Var**
	vozr: Integer;
	Begin
	Writeln('Какой возраст человека ?');
	Readln(vozr);
	Case vozr **Of**
	1..6 : Writeln('Это - дошкольник');
	7..17 : Writeln('Это - ученик');
	18..59 : Writeln('Это - работник');
	60..100: Writeln('Это - пенсионер');
	End;
	End.

Условие:
 	
Пользователь вводит целое число от 1 до 10, программа должна приписать к нему слово «ученик» с необходимым окончанием (нулевое, «а» или «ов»).
	Программа:
	
	Var n: integer;
	begin
	write('Введите число учеников ');
	readln(n);
	if (n>=1) and (n<=10) then
	begin
	write(n,' ученик');
	case n of
	2..4: write('а');
	5..10: write('ов');
	end;
	end
	else
	writeln('Ошибка ввода!');
	end.

Лабораторная работа 3

1. Создайте массив из 15 целочисленных элементов и определите среди них минимальное значение. Массив и результат выведите на экран.

2. Введите массив и целое число, определите, есть ли в массиве данное число. Массив и результат выведите на экран.

3. Напишите программу анализа значений температуры больного за сутки, определите минимальное и максимальное значения. Замеры температуры проводятся 6 раз, и результаты вводятся с клавиатуры в массив.

4. Напишите программу анализа значений температуры больного за сутки, определите среднее арифметическое значение. Замеры температуры проводятся 6 раз, и результаты вводятся с клавиатуры в массив.