Уважаемые студенты групп!

Вашему вниманию представлена лабораторная работа на тему «ОРГАНИЗАЦИЯ ПРОГРАММ ДЛЯ РАБОТЫ С ТЕКСТОМ». Работа рассчитана на 2 часа

Задание

1. Реализовать в системе Паскаль приведенный пример выполнения задания, протестировать программу и исправить ошибки.
2. Лабораторные работы оформляются в тетради в клеточку!
3. Дата предоставления фотоотчет до 31.03.2023
4. С уважением Ганзенко Ирина Владимировна

!!! Если возникнут вопросы обращаться по телефону 0721134803 (вацап), +79591134803 (телеграмм)
disobuch.ganzenko2020@mail.ru

ОРГАНИЗАЦИЯ ПРОГРАММ ДЛЯ РАБОТЫ С ТЕКСТОМ

Цель работы: закрепление теоретического материала по изучению символьных переменных.

1 Теоретические положения

Для представления отдельных символов (например, букв, цифровых символов) в языке TURBO PASCAL есть стандартный тип **Char(**от англ. - character - «символ», «знак»).

Переменным типа **Char** можно присваивать любые символы, которые вводят из клавиатуры и отображаемые на экране монитора. Каждый такой символ в ЭВМ имеет свой внутренний код(номер). При этом буквы расположены по алфавиту, цифры - в порядке роста. Поэтому любой символ имеет предыдущий ему символ(имеющий номер, на единицу меньше) и следующий(имеющий номер, на единицу больше). Например, у символа 'B' следующим символом будет 'C', а для символа '7' предыдущим есть символ '6'.

1.1 Строки

TURBO PASCAL дает возможность сообщать переменные, которым можно присваивать последовательность символов. Такие переменные имеют тип **String(**из англ. - строка).

Тип **String** в языке Турбо-Паскаль широко используется для обработки текстов и во многом похожий на одномерный массив символов(**Array** [0.N] **OfChar**). Однако в отличие от массива количество символов в строке может меняться от 0 к N, где N — максимальное

количество символов в строке. Значение N определяется объявлением типа: **String** [N] и может быть любой константой порядкового типа, но не больше 255. TURBO PASCAL позволяет не указывать явно длину строки. В этом случае длина строки принимает максимально возможное значение — 255. Строка в Турбо-Паскале трактуется как цепочка символов. К любому символу в строке можно обратиться точно так же, как к элементу одномерного массива **Array** [0.N] **OfChar**. Самый первый байт в строке имеет индекс 0 и содержит текущую длину строки. Первый значимый символ строки занимает второй байт и имеет индекс 1.

Пример:

Var k :String[3];
Begin
k**:= 'Потек' + 'ДЭН И7 ' + 'факультету ЭН';** (* РЕЗУЛЬТАТ: k='Пот' *)
End.

1.2 Процедуры над строчными переменными

В Турбо-Паскале есть следующие стандартные процедуры для обработки строчных переменных.

Процедура Delete

Формат обращения :

Delete(Str, Pos, Num)

Удаляет из строки Str, начиная с позиции **Pos, Num** символов.

Pos, Num — аргументы целочисленного типа. Если **Pos** больше длины Str, то никакие символы из **Str** не изымаются.

Если употребляет попытка изъять символы, которые находятся за границей строки (то есть Pos + Num> длины Str), изымаются только те символы, которые находятся в пределах строки.

Если **Pos** выходит за рамки диапазона 1.255, возникает ошибка выполнения.

Пример:

Var a, b : String[8];
Begin
a**:=** 'ABCDSFG';
Delete(a, 2, 4); (* РЕЗУЛЬТАТ: a='AFG' *)
b**:=** '542317';
Delete(b, 2, 10); (* РЕЗУЛЬТАТ: b='5' *)
End.

Процедура Insert

Формат обращения:

Insert(Str, Target, Pos)

Вставляет строку **Str** в строку Target, начиная с позиции **PosStr, Target** — строчныепеременные; **Pos** — аргумент целочисленного типа.

Если **Pos** больше длины **Target**, то строка **Str** присоединяется к строке **Target**.

Пример:

```
Var a : String[15];
Begin
a:= 'ABCDEFG';
Insert('XX', a, 3); (* РЕЗУЛЬТАТ: a = 'ABXXCDEFG' *)
End.
```

Процедура Str

Формат обращения:

Str(Value, St)

Выполняет превращение числового значения **Value** в строку **St**

Value - переменная целого или вещественного типа, которой могут отображаться параметры форматирования;

St - строчная переменная.

Параметры форматирования записываются через двоеточие, указывают формат числа (общее количество символов в строке St и количество символов, дробной частью).

Пример:

```
Var  i : Integer; k : real; a : String[10];
Begin
i := 1234;
Str( i : 5, a);     (* РЕЗУЛЬТАТ: a = '  1234' *)
k := 2.6E4;
Str( k :8: 0, a);  (* РЕЗУЛЬТАТ: a = '            2500' *)
End.
```

Процедура Val

Формат обращения :

Val(Str, Var, Err)

Осуществляет превращение строчного выражения **St** в переменную **Var** целого или вещества типа.

Числовое значение строки **Str** определяется в соответствии с правилами, применяемыми к числовым константам. Строка **Str** не должна

содержать незначительных пробелов в начале или в конце. **Err** - целая переменная.

Если при выполнении превращения ошибок не выявлено, переменная **Err** принимает значение 0. В противном случае(например, Str содержит символы, недопустимые в записи численной константы) значения **Errr** равняется номеру позиции первого ошибочного символа. Значение **Var** в этом случае не определенно.

Пример:

```
Var  i, Result :Integer; k :real;  a : String[10];
Begin
a:= '234';
Val(a, i, Result); (* РЕЗУЛЬТАТ: i=234, Result=0 *)
a:= '2.5E4';
Val(a, k, Result); (* РЕЗУЛЬТАТ: k=2500, Result=0 *)
a:= '23X';
Val(a, i, Result);  (* РЕЗУЛЬТАТ: i - Не определен, Result=3 *)
End.
```

1.3 Функции над строчными переменными

В Турбо-Паскале есть следующие стандартные функции над строчными переменными.

Функция Copy

Формат обращения :

Copy(Str, **Pos, Num).**

Значением функции **Copy** является подстрокой строки **Str,** которая содержит **Num** символов, начиная с позиции **Pos.**

Str — строчное выражение; **Pos, Num** — аргументы целочисленного типа. Если **Pos** превышает длину **Str**, то значением функции **Copy** будет пустая строка(то есть строка в котором нет ни одного символа).

Пример:

```
Var  a, b : String[10];
Begin
a:= 'ABCDEFGH ';
b:= Copy(a, 3, 2);    (* РЕЗУЛЬТАТ: b = 'CD' *)
End.
```

Функция Concat

Формат обращения :

Concat(Strl, Str2...,Str);

Результатом функции **Concat** является строка, составленная из цепочек строк - аргументов функции **Concat.** Строки сцепляются в том порядке, в котором они отмечены в обращении к функции. Количество строк - аргументов может быть произвольным. Как аргументы могут быть строчные выражения. Аргументы отделяются один от одного запятыми. Если длина полученного в результате строки больше 255, возникает ошибка выполнения.

То же можно сделать с помощью оператора «+». Функция **Concat** включена в систему Турбо-паскаль только для совместимости с другими версиями языка Турбо-Паскаль.

Пример:

```
Var  a, b, k : String[10];
Begin
a:= 'ТУРБО-';
   b:= '    самый БЫСТРЫЙ';
k:= Concat( a, 'ПАСКАЛЬ', b);
   (* РЕЗУЛЬТАТ: b = 'ТУРБО-ПАСКАЛЬ БЫСТРЕЕ' всего *)
End.
```

Функция Length

Формат обращения :

Length(Str);

Результатом функции является число целочисленного типа, которое равняется длине строчного выражения **Str**.

Пример:

```
Var  k : Integer; a : String[10];
Begin
a:= 'ТУРБО-';
k:= Length(a);   (* РЕЗУЛЬТАТ: k = 6 *)
End.
```

Функция Pos

Формат обращения :

Pos(Str, Target)

Результатом функции **Pos** является номер позиции в строке с которой начинается первое вхождение в ней строки **Str.**

Str, Target - строчные выражения; результат целочисленного типа. Если строка **Str** не входит в строку **Target**, результатом функции Будет 0

Пример:

```
Var  k : Integer; a, b : String[10];
Begin
a:= 'ABCdeFG';
k:= Pos( 'de', a); (* РЕЗУЛЬТАТ: k = 4 *)
k:= Pos('h', a);   (* РЕЗУЛЬТАТ: k = 0 *)
End.
```

Строки и символы

Данные строчного типа **String** и стандартного скалярного типа **Char(**символьного) - взаимосовместимые.

Таким образом, если в каком-либо выражении предусматривается использование строчной величины, то вместо ее можно задать величину символьного типа и наоборот. Более того, строки и символы в выражениях можно смешивать. Когда символьной величине присваивается значение срочного типа, то длина строки должна равняться единице, в противном случае возникает ошибка выполнения .

1.4 Введение символьных данных

Особенности:

Пропуск также является символом, при введении символьных данных его нельзя использовать в качестве разделитель данных при введении числовых значений. Точнее говоря, пробел при введении символьных данных будет также восприниматься как символ.

Пример:

```
var  a, b, c: Char;
Begin
Read(a, b, c);
End.
```

Если символьным переменным a, b, c необходимо присвоить при введенные значения:

a = 'S', b = 'N', c = 'R', то на клавиатуре стоит набрать SNR.

Если же ввести: S N R то получим: a ='S', b ='_', c ='N'.

Другая особенность введения символьных данных заключается в том что нажатие клавиши (<ENTER>) воспринимается как символ (пробел). Рассмотрим следующий фрагмент программы :

```
var  a, b :Integer;  c, d: Char;
Begin
Read(a, b);
Read(c, d);
End.
```

Если ввести:

3_,4

То переменные получат следующие значения:

a = 3, b = 4(',_,' - воспринимается как пробел)

Для правильного введения символьных данных необходимо использовать оператор **Readln.**

С введением символьных данных связана стандартная функция EOLN(EndOfLine - конец строки).

Функция **EOLN** чаще всего используется в операторах цикла :

```
Var a, i : Integer ;
Begin
Readln;
i:= 0;
While Not EOLN Do
begin
Read(a);
i:= i + 1;
end;
End.
```

Циклическая часть выполняется до тех пор, пока не встретится символ конца строки <ENTER>.

Например, вводится:

4,_,5 ,_,13 ^ 6,_, 2 , ,17, ,14

Цикл **While** будет выполняться, пока нет конца строки < ENTER>.

Перед использованием оператора **WhileNot EOLN Do** необходимо ставить оператор **Readln** без параметров.

1.5 Стандартные символьные функции

PRED(X)

X является переменной или константой типа **Char**. Значением функции является символ, код которого на 1 меньше кода аргумента(из англ. PREDecessor - предыдущий элемент).

Например, функция **PRED**('L') определяет предыдущий символ('K').

SUCC(X)

X является переменной или константой типа CHAR. Значением функции является символ, код которого на 1 больше кода аргумента(из англ. SUCCessor - следующий элемент).

Например, функция SUCC('L') определяет следующий символ('M').

ORD(X)

X является переменной или константой типа CHAR. Значением

функции является код(порядковый номер) аргумента(из англ. ORDer порядок).

Например, функция ORD('A') определяет порядковый номер символа 'A'.

CHR(N)

N является переменной или константой типа INTEGER, Значением функции является символ, код которого равняется N(из англ. СнаRaetersимвол).

Пример:

```
Program LETTER;
var X1, X2, X3: Char;
Begin
X1 := 'L';
writeln(X1);
X2 := PRED(X1);
writeln('PRED=', X2);
X3 := SUCC(X1);
writeln('SUCC=', X3);
End.
```

РеакцияЭВМ :

```
L
PRED=K
SUCC=M
```

Пример:

```
Program LETTER;
var N : Integer; X: Char;
Begin
X := 'L';
writeln(X);
N := ORD(X);
writeln(N);
X := 'A';
writeln(X);
X := CHR(N);
writeln(X);End.
```

РеакцияЭВМ :

```
L
76
A
I
```

1.6 Упакованные массивы

Как правило, Турбо-Паскаль отводит в памяти ЭВМ для переменной типа **Char** одно машинное слово(2 байта). Это связано с тем, что при таком распределении памяти доступ к данным происходит наиболее быстро. В то же время для представления символьной переменной достаточно одного байта памяти. Поэтому с целью экономии памяти при использовании символьных данных в языке Турбо-Паскаль введенное понятие *упакованного массива.* Элементы упакованного массива хранятся по двум в одном слове. Упакованный массив символьных данных имеет следующее описание:

Type
T = PACKED **Array [1.N] Of Char**;
Var
FAM: T;

Здесь N - целочисленная константа, которая задает количество символов в массиве; FAM - упакованный массив символов.

Тип упакованного массива, в частности, имеют срочные константы:

1.7 Обработка символьной(строчной, текстовой) информации

Как мы уже отмечали, срочный тип данных(**String**) является структурным типом. Он во многом похожий на тип символьного массива(**Array**). Расхождение между ними в том, что количество символов в строке(длина строки) может изменяться динамически от 0 до 255, тогда как количество элементов в массиве строго фиксировано.

1.8 Строчные выражения

Операции над строками выполняются с помощью выражений, элементами которых являются строки.

Строчные выражения могут содержать:
1) строчные константы;
2) строчные переменные;
3) обращение к функциям;
4) операции над строками.

В Турбо-Паскале есть следующие операции над строками

А. Операция сцепления(присоединение)

Данная операция соединяет два строки в один. Обозначается знаком '+'.

Пример:
’TURBO'+'PASCAL'='TURBOPASCAL'
’125’ + ’.’ + ’328’ = ’125.328’
‘A' + 'B' + 'C'+'D' = 'ABCD'

Ту же функцию выполняет и функция **Concat(**из англ. сцепления,

конкатенация).

Если длина получаемой строки больше 255, возникает ошибка выполнения.

В. Операции отношения =,<>, >, <, >=, <=

Операции отношения используются для сравнения двух строк.

Результат выполнения операций отношения имеет тип **Boolean** то есть принимает значение **True(**истина) или **False(**неправда).

Сравнение выполняется над каждым символом с левой стороны справа. Считается один символ меньше(больше) другого, если его код меньше кода другого символа.

Строки считаются равными тогда и только затем, когда они содержат одинаковую длину и одинаковые символы на соответствующих местах.

Например:

‘TURBO' = 'TURBO' (значение операции - TRUE)

‘TUBBO _' = 'TURBO' (значение операции - FALSE)

1.9 Оператор присвоения значений строчным переменным

Используется для присвоения значения строчного выражения, переменной срочного типа.

Например:

A := '157'; B := 'ПОТЕК' + 'ДЭН И7' + 'ФАКУЛЬТЕТУ ЭН';

Если длина результата выражения превышает максимальную длину строки, лишние символы отрезаются справа.

2 Примеры решения задач из обрабатывания символьной информации

Пример 1.

Сложить таблицу идентификаторов и ТР - программу. Определить, сколько раз в этом тексте встречается буква "и". Текст: Возраст прожил а ума не нажил.

Составим таблицу идентификаторов :

I	Количество букв «и»	Длина текста
I	N	len

Сложим ТР - программу:

```
Program UIPA;
var a, b, c : String;
i, n, len : Integer;
Begin
```

```
writeln('Введитетекст');
read(a);
writeln;
writeln(a);
len:= Length(a);
n:= 0;
b:= 'и';    (* Букву 'и' набирать на русском регистре *)
for i := 1 to len do
if a[i] = b then n := n + 1;
writeln('Количество  букв и = ', n: 2);
End.
```

Реакция ЭВМ :

Сложим блок-схему алгоритма :

Пример 2.

Составить программу которая удалит все пропуски из текста

```
Program UIPA;
vara : String;
Begin
writeln('Введитетекст :');
read(a);
writeln;
writeln(a);
whilepos(' ', a) <> 0 do Delete(a, pos(' ', a), 1);
writeln('Результат:');
writeln(a);
End.
```

Пример 3.

Составить программу которая удалит слово 'РОЗЫ' из текста.

```
Program UIPA;
vara, b, c : String[255];
k :Integer ;
Begin
writeln('Введитетекст :');
read(a);
repeat
k:= pos(' ', a);
b:= copy(a, 1, k - 1);
```

```
if  b<> 'РОЗЫ' then c := c + b + ' ';
Delete(a, 1, k);
until k = 0;
writeln('Результат:');
writeln(c);
End .
```

Пример 4.

Составить программу которая на место пробелов поставить символы ';:'.

```
Program UIPA;
var a : String;
i :Integer;
Begin
writeln('Введитетекст');
read(a);
for i := 1 to Length(a) do
if a[i] = ' ' then a[i] := ';:' ;
writeln('Результат:');
writeln(c);
End.
```

3 Вопрос для самоконтроля подготовки к лабораторной работе

1. Дайте характеристику переменных типа **Char**.
2. В каком разделе Pascal - программы должен быть объявленным переменная символьного типа?
3. Как сравниваются между собой переменные типа char?
4. Почему символом украинской азбуки отвечают числа, больше чисел, которые отвечают символам латинской азбуки?
5. Что значит символ#13?
6. Чем в Pascal отличаются функции **Chr** и **Ord**?
7. Чем между собой отличаются типы **String** и **Char**?
8. Возможная совместимость ли строчного типа string и стандартного скалярного типа Char**(**символьного)?
9. Какие особенности следует иметь в виду при введении символьных(строчных) данных в память компьютера?
10. Как строчные данные выводятся к печати?
11. Перечислите стандартные символьные функции языка Pascal для обработки текста.
12. Чтоозначаютфункции**Succ(c)**, **Pred(c)**, **Ord(a)**, **Chr(n)?**
13. Что это - упакованные массивы?
14. Как объяснить эту запись: k = Packed**Array** [l...n] **ofchar**?
15. Из каких частей состоят строчные выражения?

16. Какие операции можно выполнять над строками?
17. Почему равняется это выражение: k: = 'a' + 'b' + 'c'?
18. С помощью каких операций можно сравнивать строки?
19. Приведите примеры сравнения строк.
20. Как работает инструкция присвоения? Приведите примеры.
21. Какие стандартные процедуры существуют в Pascal для обрабатывания строчных переменных?
22. Какой синтаксис обращения к процедуре **Delete**?
23. Как работают процедуры **Insert**, **Str** и **Val**?
24. Как работает функция **Pos**?
25. Как работает функция **Copy**?

4 Задание

Варианты задания

НаписатьТР - программу решения задания обработки заданной символьной информации.

Таблица 1. ***Задание***

№ варианта	**Задание**	**Текст**
1	Найти самое длинное слово в тексте	Звуки вальса особенно волнуют того, кто не умеет танцевать.
2	Зашифровать текст, записывая все слова наоборот, а потом расшифровать его	Видит глаз глубоко, а ум еще глубже.
3	Поменять местами первые и последние четыре слова.	Не каждый господин у кого штаны на выпуск.
4	Вставить между словами в тексте вместо одного пропуска, точку с запятой.	Человек без сомнений должен непременно вызывать сомнению окружающих.
5	На какую букву начинаются больше слов в тексте.	Идеалы голодного желудка после сытного обеда переменчивы.
6	Удалить часть текста, который находится между вторым и первым пропуском.	Не рядись бараном потому что еще волк съест.
7	Выписать из данного текста строку, которая состоит из последних трех слов.	Не бойся ругательного а бойся кусливого.
8	Определить, какие символы и сколько раз встречаются *и* тексте.	И на здоровой яблоне гнилое яблоко найдешь.

9	Удалить из текста шестое слово.	Нет более доброй водицы как из родного колодца.
10	Удалить все пропуски из текста.	Умение никто за плечами не носит.
11	Зашифровать и расшифровать слова, которые стоят на парных местах в тексте, записывая их наоборот.	Не жалеет добрый жнец, что широкий чугунок.
12	Найти слова одинаковой длины.	Кучер хороший и кони не везут.
13	Выписать из текста последние три слова.	В хорошегопильщика пилка острая.
14	Выбрать все слова, которые стоят на непарных местах.	Не всякое блестящее ружье стреляет.
15	Определить само длинное слово текста.	Старый дурак исторически терпеть не может младшего умника.
16	Выбрать четыре последних слова текста.	Пасечник и врач и в городе от голода не умрут.
17	Выписать слова, которые начинаются с «Р».	Кто попал под микроскоп не понимает своего размера.
18	Поменять местами первое и третье слова.	Еду на завтра отложи, а работу сегодня сделай.
19	Выбрать все слова, которые начинаются на букву «К».	Большое чудо, что корова черная, а молоко седо.
20	Выбрать все слова, которые стоят на парных местах.	В глазах волка светилась искренняя тоска и неукротимая любовь к баранине.
21	Выбрать пять первых слов.	Дождь идет не тогда когда просят, а тогда когда жнут и косят.
22	Поменять местами слова, которые начинаются на буквы «З» и «Л».	Музыкальное искусство всегда любило хвататься за дирижерскую палочку.
23	Определить, сколько слов в тексте начинаются на букву «З».	Человек, влюбленный у самого себя, не оставляет места влюбленности для другого.
24	Подсчитать количество слов, которые состоят из трех и пяти букв.	В азбуке лица и внешнего вида не все одинаково грамотные.

25	Заменить символом «*» все пробелов текста	На телевизионных экранах вспыхнула многосерийная эпидемия.
26	Выбрать все слова, которые состоят из пяти букв.	Жажде упасть из коня непременно должна сопровождать жажда сесть на нее верхом.
27	Выбрать слова, которые заканчиваются на «ГО» и «ОМ».	Глаза будущего смотрят на нас суровым взглядом всего прошлого.
28	Выписать из текста слова, которые содержат пять и шесть букв.	Глупого учит, как мертвого считает.
29	Сложить текст из первого, четвертого и двух последних слов.	Поголовье раскидистых оленьих рогов усилиями человечества не уменьшается.
30	Найти количество слов, которые начинаются с «П».	Протертые штаны еще не признак усидчивости.