Уважаемые студенты групп!

Вашему вниманию представлена лекция на тему «ОРГАНИЗАЦИЯ ПРОГРАММ ДЛЯ РАБОТЫ С ПЕРЕМЕННЫМИ ТИПА STRING»

Задание

1. Прочитать внимательно лекцию.
2. Законспектировать лекцию в рабочую тетрадь не менее 3-5 страницы рукописного текста. В конспекте лекции обязательно должно быть приведены примеры.
3. Решить приведенные в лекции в контрольных вопросах задачи.

С уважением Ганзенко Ирина Владимировна

!!! Если возникнут вопросы обращаться по телефону +79591134803(телеграмм)

disobuch.ganzenko2020@mail.ru

ОРГАНИЗАЦИЯ ПРОГРАММ ДЛЯ РАБОТЫ С ПЕРЕМЕННЫМИ ТИПА STRING

1 Теоретическое положение

Строчные переменные - это одномерные массивы символов, для описания которых в TURBO PASCAL введен тип String. Например, если строка содержит до 30 символов, ее тип будет определен как

type s= String[30].

Длина строки не может содержать более чем 255 символов.

В TURBO PASCAL определенно понятие строки переменной длины, в этом случае ее описание задается как

type s= String;

Тип String без указания длины соединен со всеми типами строк. Особенностью строчных переменных является то, что к ним можно обращаться как к скалярным переменным, так и к *массивам*. Во втором случае применяется конструкция "переменная с индексом", которая обеспечивает доступ к отдельным символам строки. При этом нижняя граница индекса равняется 1. Отдельный символ строки s получим с типом Char.

В памяти ЭВМ строка занимает количество байтов, на единицу больше ее длины. Нулевой байт строки содержит ее длину. Для строк определенные операции присвоения, слияния (конкатенации) и сравнения.

Для сравнения строк применяются все операции отношения. Сравнение строк происходит для каждого символа, начиная с первого символа. Строки равны, если имеют одинаковую длину и имеют эквивалентные символы.

Строки могут быть элементами списка ввода - вывода, при этом записывается имя строки без индекса.

При введении строчных переменных количество символов, которые вводятся, может быть меньше, чем длина строки. В этом случае вводят символы, что, размещаются с начала строки, а байты, которые остались, заполняются пробелами. Если количество символов, которые вводят, превышает длину строки, лишние символы отбрасываются.

Инициализация строк может производиться так же с помощью типизирующих констант:

consts Name: String[9]= 'IBM PC/AT';

Для работы со строками в TURBO PASCAL включенны процедуры и функции, которые обеспечивают редактирование и превращение строк.

Тип **String(**строка) в Турбо Паскале широко используется для обработки текстов. Этот тип является стандартным и во многом похожий на одномерный массив символов **Array** [0..N] **of** Char. Значение N отвечает количеству символов в строке и может изменяться от 0 до 255. Символы, которые входят в строку, занимают позиции с 1 до N. Начальный байт строки с индексом 0 содержит информацию о ее длине, то есть это символ с кодом, равным длине строки.

Можно, также описывать переменные типа **String**[K], где K - целое число не больше 255. Так определяются строки с длиной не больше K. Этот тип уже не является стандартным. С символами строки можно работать как с элементами массива из символов, но в отличие от массивов, строки можно вводить полностью, сравнивать друг с другом и соединять операцией "+".

Сравнение строк выполняется из сравнения каждого символа в соответствии с их кодами к первому расхождению. Если одна из строк закончилась к первому расхождению, то она считается меньшей. Пустая строка меньше любой строки.

Пример*: Сравнение строк.*

'abcd'> 'abcD'{ 'd'>'D'}
'abcd'> 'abc' { 'd'>'' }
'abc'< 'axxc' { 'b'<'x'}
'abcd' = 'abcd'

Существует ряд стандартных функций и процедур для работы со строками.

- Функция **Length(s)** выдает длину строки s.
- Функция **Concat**(s1, s2.,sn) возвращает строку s1+s2+.+sn.
- Функция **Copy**(s, p, k) возвращает фрагмент строки s, что начинается в позиции p и имеет длину k.
- Функция **Pos**(s1, s) ищет первое вхождение подстроки s1 в строку s и возвращает номер первого символа s1 в строке s или 0 если сходства не нашли.

- Процедура **Delete**(St, poz, n) удаляет из строки s фрагмент, который начинается в позиции p и имеет длину k.

Пример

Значение St	**Выражение**	**Результат**
‘абвгде'	Delete(St, 4, 2)	‘абве'
‘река Волга'	Delete(St, 1, 5)	‘Волга'

- Процедура **Insert**(s, s1, p) вставляет в строку s подстрока s1, начиная из заданной позиции p.
- **Chr(x)** - превратит выражение x в символ и возвращает значение символа
- **Ord**(ch) - превратит символ ch в его код и возвращает значение кода
- **Pred**(ch) - возвращает предыдущий символ
- **Succ**(ch) - возвращает следующий символ

Пример.
Ord(':') =58
Ord('A') =65
Chr(128) =Б
Pred('Б') =А
Succ('Г') =Д

Сначала любой описан в разделе Var строка содержит "мусор" и рекомендуется заполнять строки пустыми значениями или чем-либо другим.

Для заполнения достаточно длинных строк одинаковыми символами используются встроенная процедура FillChar, например:

```
VarS : string[80];
Begin. . .
      FillChar(S[1],80' '); {Заполнениестрокипробелами}
      S[0]:=Chr(80);        {Занос кода длины строки}
            . . .
End.
```

Длину строки можно определить также, используя встроенную функцию **Length(S)**, где S - строка типа String.

В ряде случаев возникает необходимость превращения числовых значений в строку и наоборот. Для этого можно использовать две процедуры:

1) **Str**(X, S) - превратит числовое значение X в строчное S. Возможно задание формата для X в виде: X: F: n(для чисел веществ, где F - общее число позиций выделяемых под число, включая десятичную точку, а n - число знаков в дробной части) или X: F(для целых чисел). Эта функция чаще всего используется при работе с процедурами модуля GRAPH. Например:

Str(55, s); - строчная переменная s принимает значение, равное '55'.

2) **Val**(S, X, ErrCode) - превратит строку S в числовое значение, если это возможно. Параметр ErrCode содержит ноль, если превращение прошло успешно, и тогда в X содержится результат превращения, в противном

случае он содержит номер позиции в строке S, где выявлен ошибочный символ. Например:

Val('125',K, kod) - в результате выполнения этой процедуры переменная К получает целое значение, ровное 125, параметр kod=0;

Val(' 1.05',M, code) - M=1.05, code=0;

Val('100, ',N, code) - это ошибочный вызов, потому что в исходной строке на 4-й позиции располагается недопустимый для числа символ ',' и потому параметр code=4, а переменная N остается без изменения.

Турбо Паскаль позволяет делать превращение числовых значений в сроку и наоборот. Для этого используются процедуры **Str**(X: n: d, S) и **Val**(S, X, e). Первая получает их числа X строчка S из изображения этого числа, в которой не менее n символов и из них d знаков после запятой. Параметры n и d необязательны. Вторая процедура получает из строки S число X. При успешном результате e = 0.

Пример оформления программы для работы с текстом:

```
var   s, x, y, z : string;
c : char;
begin
x:='turbo';
y:='pascal';
z:=x+' '+y; { z='turbo pascal'}
writeln(z);
s:=''; { пустая строка }
for c:='a' to 'z' do s:=s+c; { s='abcd.xyz'}
writeln(s);
end.
```

РеакцияЭВМ :

```
turbopascal
abcdefghijklmnopqrstuvwxyz
```

Блок-схема алгоритма:

Вывод: эта лабораторная работа раскрывает понятие символьной переменной, дает возможность получить знание и умение использовать текстовые данные для программирования в языке Pascal.

2 Контрольные вопросы

1. Дайте характеристику переменным типа String?
1. Что значит символ #13?

2. Чем отличаются функции **Chr** и **Ord**?
3. Объясните запись **var** s :**string[23];**?
4. Какой оператор следует использовать для ввода данных с клавиатуры?
5. Как работает функция **pos**, приведите примеры.
6. Что означает функция **Ord**?
7. Что означает функция **Chr**?
8. Для чего нужная процедура **FillChar**?
9. Что это - упакованные массивы?
10. Как объяснить запись: constsName: String[9]= 'IBM PC/AT';?
11. Какие операции можно выполнять над строками?
12. С помощью каких операций можно сравнивать строки?
13. Какие стандартные процедуры существуют для обрабатывания текстовой информации?
14. Приведите пример использования процедуры **delete**?
15. Как работают процедуры **copy, str** и **val**?
16. Что будет выведено на экран после выполнения программы?

```
var
s : string;
c : char;
begin
s:='';
for c:='a' to 'z' do s:=s+c;
writeln(s);
end.
```

17. Объясните разницу между типами **string** и **char**.
18. Для чего используют функцию **Length**?