

Обработка строк

Стандартные подпрограммы работы со строками

Для обработки данных типа `string` в языке Паскаль существует несколько стандартных подпрограмм. Ниже приводятся некоторые из них.

Функция `Copy(s, Index, Count)` создаёт копию части строки `s`, начиная с символа номер `Index`, длиной `Count` символов. `Index` и `Count` должны быть переменными или константами целого типа. Ниже дан пример использования этой функции.

```
Program Prog1;
Var s1 : string;
Begin
    s1:=Copy('Столовая',1,4);
    WriteLn(s1)
End.
```

Эта короткая программа выведет слово «Стол», т.к. это часть слова «Столовая», начинающаяся с первого символа и длиной 4 символа.

Процедура `Delete(s, Index, Count)` удаляет `Count` символов строки `s`, начиная с позиции номер `Index`. `s` должна быть переменной типа `string`. Строковая константа в данном случае недопустима.

```
Program Prog2;
Var s : string;
Begin
    s:='Столовая';
    Delete(s,5,4);
    WriteLn(s)
End.
```

В этом примере будет также выведено слово «стол», т.к. в переменной `s` перед выводом удаляется 4 символа, начиная с пятого. Т.е. удалена будет часть строки 'овая'.

Процедура `Insert(source, s, Index)` вставляет строку `source` в строку `s`, начиная с позиции `Index` строки `s`. Остальные символы строки `s` сдвигаются вправо. При этом `s` должна быть строковой переменной, а `source` может быть и константой.

```
Program Prog3;
Var s1,s2 : string;
Begin
    s1:='Школа 5';      //В строке s1 один пробел
    s2:='номер 2';
    Insert(s2,s1,7);
    WriteLn(s1)
End.
```

Программа Prog3 выводит сообщение «Школа номер 25». Седьмой символ исходной строки `s1` — цифра «5». Именно с этой позиции вставляется содержимое строки `s2`, а цифра «5» смещается вправо.

Функция `Pos(substr, s)` находит первое вхождение подстроки `substr` в строку `s`. Результат — целое число, равное номеру позиции в строке `s`, где находится первый символ подстроки `substr`. Если подстроки `substr` в `s` не обнаружено, результат равен нулю.

```
Program Prog3;
Var s1,s2 : string;
    i      : Integer;
Begin
    s1:='К 12 прибавить 12'; //Слова разделены одним пробелом
```

```

s2:='12';
i:=Pos(s2,s1);
WriteLn(i)
End.

```

После запуска этой программы на экране увидим число 3. В строке *s1* подстрока '12' впервые встречается, начиная с символа номер 3.

Процедура *SetLength(s, len)* устанавливает длину строки *s* равной целому неотрицательному значению *len*.

Процедура *Val(s, V, code)* преобразует строку символов *s* в число *V*. Переменная *V* может быть целого или вещественного типа. В случае успешного преобразования переменная *code* типа Integer получает значение ноль. В противном случае в *code* будет содержаться номер символа, при обработке которого возникла ошибка преобразования, а значение переменной *V* останется неопределённым. Строка *s* может содержать лидирующие пробелы. Пробелы в конце строки не допускаются. Например, ' 12' будет преобразовано правильно, а '12 ' — нет. Число в строке *s* может быть записано в десятичной или шестнадцатеричной системе счисления. Например, '32' или '\$20'. В отличие от Delphi при программировании на FreePascal допускается также использование двоичной ('%100000') и восьмеричной ('&40') систем счисления.

Вот пример использования процедуры Val.

```

{$mode delphi}
Program pv;
Var s : string;
    x : Integer;
    code : Integer;
Begin
  Write('Введите число ');
  Read(s);
  Val(s,x,code);
  If code <> 0
  Then WriteLn('Ошибка в символе номер ', code)
  Else WriteLn(x)
End.

```

Эта программа получает целое число в виде строки и преобразует эту строку к типу Integer.

Процедура *Str(x,s)* преобразует число *x* в его запись в виде строки *s*. Переменная *x* может быть любого числового типа. После *x* можно указать ширину строки, а для вещественных типов и число знаков после запятой, как и при выводе посредством Write. Например, Str(x:5:2,s).

Функции преобразования из SysUtils

Для использования описанных ниже функций нужно подключить модуль SysUtils. Для этого после имени программы пишем

Uses SysUtils;

В Delphi это делается автоматически, а в FreePascal указанную строку придётся вписать вручную.

Функция *IntToStr(x)* преобразует целое число *x* в строку из десятичных цифр. Например, после вызова s:=IntToStr(158) значение строковой переменной s будет равно '158'.

Функция *IntToHex(x, digits)* преобразует целое число *x* в его запись в шестнадцатеричной системе счисления. Строка-результат содержит минимум *digits* символов. Если *digits* меньше, чем требуется цифр для записи числа, строка не усекается. Если же *digits* больше, чем нужно знаков для записи числа, в строку записываются лидирующие пробелы. Например, если x=\$12345, то вызов s:=IntToHex(x,2) приведёт к результату s, равному '12345', а вызов s:=IntToHex(x,7) — к результату '0012345'.

Функции *StrToInt(s)* и *StrToInt64(s)* преобразует строку *s*, содержащую запись числа десятичными или шестнадцатеричными цифрами, в целочисленную переменную типа *Integer* или *Int64*. Так же, как и у процедуры *Val*, допускаются пробелы в начале строки и не допускаются в конце.

Функция *StrToFloat(s)* преобразует строку цифр в числовую переменную вещественного типа. Обратное преобразование осуществляет функция ***FloatToStr(x)***. В этом случае строка-результат может содержать запись числа как в обычной (12.34), так и в научной (1.234e1) нотации. Выбирается более короткая.

Часто бывает полезной **функция *Trim(s)***. Она удаляет пробелы на обоих концах строки *s*. Если строка состоит только из пробелов, результатом будет пустая строка. Например, после вызова *s:=Trim(' ab cd ')* значение переменной *s* будет равно 'ab cd'.

Примеры решения задач

Задача 1

Удалить часть символьной строки, заключённой в скобки (вместе со скобками). В строке имеется не более одной пары правильно расставленных скобок.

Решение.

Идея алгоритма решения этой задачи очень проста.

1. Находим открывающую скобку. Пусть это символ номер *first*.
2. Находим закрывающую скобку. Пусть это символ номер *last*.
3. Начиная с позиции номер *first*, удаляем *last – first + 1* символов.

Вводить данные будем с клавиатуры.

```
{ $mode delphi }
Program DelBrackets;
Var s : string;
    first, last : Integer;
Begin
  WriteLn('Введите строку');
  ReadLn(s);
  first := Pos('(', s);
  last := Pos(')', s);
  Delete(s, first, last-first+1);
  WriteLn('Результат');
  WriteLn(s);
End.
```

В этой программе мы использовали только что рассмотренные подпрограммы обработки строк. Но можно было обойтись и без них.

1. Идём по строке *s*, пока не встретим символ '(', но не далее предпоследнего символа строки. (Последним символом открывающая скобка быть не может. Где тогда закрывающая?) Если открывающая скобка не найдена, закончить работу алгоритма.
2. Пусть '(' расположена в позиции номер *first*. *Продолжаем* просматривать строку, пока не встретим символ ')'. Пусть это случится в позиции *last*.
3. Начиная с позиции *last+1* и до конца строки смещаем все символы на *lb = last – first + 1* позиций влево. (*lb* — это длина части строки в скобках вместе со скобками.)
4. Уменьшаем длину строки до *length(s) – lb* символов.

```
{ $mode delphi }
Program DelBrackets2;
Var s : string;
    first, last : Integer;
```

```

        i, lb, len  : Integer;
Begin
    WriteLn('Введите строку');
    ReadLn(s);

    (* Длина строки *)
    len:=length(s);

    (* Ищем '(' *)
    first:=1;
    While (first<len) and (s[first]<>'(') Do
        first:=first+1;

    (* Открывающая скобка не может быть последним символом строки *)
    If first<len Then //Открывающая скобка найдена
    Begin
        (* Ищем ')' *)
        last:=first+1;
        While s[last]<>')' Do
            last:=last+1;

        (* Длина выражения в скобках *)
        lb:=last-first+1;

        (* Подтягиваем хвост строки *)
        i:=last+1;
        While i <= len Do
            Begin
                s[i-lb]:=s[i];
                i:=i+1
            End;

        (* Теперь строка короче. Обрезаем лишние символы *)
        SetLength(s, len-lb);
    End;

    WriteLn('Результат');
    WriteLn(s)
End.

```

При поиске закрывающей скобки мы продолжили просматривать строку с позиции, следующей за найденной открывающей скобкой, а не начинали новый поиск с начала строки. Это позволяет уменьшить время работы программы.

Задача 2

В первом туре регионального этапа Всероссийской олимпиады школьников по информатике 2010 года была предложена следующая задача.

Задача А. Соревнование картингистов

Имя входного файла: gase.in

Имя выходного файла: gase.out

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

Максимальная оценка: 100 баллов

После очередного этапа чемпионата мира по кольцевым автогонкам на автомобилях с открытыми колесами Формула-А гонщики собрались вместе в кафе, чтобы обсудить полученные результаты. Они вспомнили, что в молодости соревновались не на больших болидах, а на картах – спортивных автомобилях меньших размеров.

Друзья решили выяснить победителя в одной из гонок на картах. Победителем гонки являлся тот гонщик, у которого суммарное время прохождения всех кругов трассы было минимальным.

Поскольку окончательные результаты не сохранились, то каждый из n участников той гонки вспомнил и выписал результаты прохождения каждого из m кругов трассы. К сожалению, гонщикам было сложно вычислить победителя той гонки. В связи с этим они попросили сделать это вас.

Требуется написать программу, которая вычислит победителя гонки на картах, о которой говорили гонщики.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m ($1 \leq n, m \leq 100$). Последующие $2n$ строк описывают прохождение трассы каждым из участников. Описание прохождения трассы участником состоит из двух строк. Первая строка содержит имя участника с использованием только латинских букв (строчных и заглавных). Имена всех участников различны, строчные и

заглавные буквы в именах различаются.

Вторая строка содержит m положительных целых чисел, где каждое число — это время прохождения данным участником каждого из m кругов трассы (каждое из этих чисел не превосходит 1000). Длина имени каждого участника не превышает 255 символов.

Формат выходного файла

В выходной файл необходимо вывести имя победителя гонки на картах. Если победителей несколько, требуется вывести имя любого из них.

Пример

race.in	race.out
5 3 Sumaher 2 1 1 Barikelo 2 1 2 Olonso 1 2 1 Vasya 1 1 1 Fedya 1 1 1	Fedya

Решение

Это лёгкая задача. Здесь она рассматривается просто для того, чтобы показать возможность хранения данных в строковых массивах. Следует также аккуратно вводить данные, чтобы не попасться в ловушку, связанную с символами с кодами #13 и #10 в конце строки.

Просто храним имена гонщиков в массиве строк, время прохождения кругов считываем и сразу складываем, вычисляя время прохождения трассы. Запоминаем минимальное время прохождения трассы и номер любого из гонщиков, который его показал. В ниже приведённой программе используется номер последнего гонщика, показавшего лучшее время. В начале работы программы в качестве времени победителя используется явно завышенное время. Максимально возможное время победителя mintime равно $1000 \cdot 100 = 10^5$, что гораздо меньше, чем MAXINT — максимальное значение величины типа `Integer`. Это сделано для того, чтобы начальное значение `mintime` было гарантировано заменено на время одного из гонщиков. Программа написана на Delphi.

```
program race;  
  
{$APPTYPE CONSOLE}  
  
uses  
    SysUtils;
```

```

Const MAX = 100;

Var f      : Text;
    m,n    : Byte;
    i      : Byte; //Номер гонщика
    j      : Byte; //Номер круга
    name    : array[1..MAX] of string; //Имена гонщиков
    mintime : Integer; //Время победителя
    time    : Integer; //Время прохождения трассы очередным гонщиком
    winner  : Byte;    //Номер победителя
    x      : Integer;  //Время прохождения очередного круга
begin
    Assign(f, 'race.in');
    Reset(f);
    ReadLn(f,n,m);

    mintime:=MAXINT; //Это время завышено. Оно не будет минимальным
    For i:=1 To n Do
        Begin
            //Вводим имя гонщика
            ReadLn(f,name[i]);

            //Считываем времена x прохождения кругов
            //и вычисляем время time прохождения трассы гонщиком номер i
            time:=0;
            For j:=1 To m Do
                Begin
                    Read(f,x);
                    time:=time+x
                End;

            //Если время прохождения трассы гонщиком номер i не хуже,
            //чем у других,
            //считаем пока его победителем (можно было < вместо <=)
            If time<=mintime Then
                Begin
                    mintime:=time;
                    winner:=i
                End;

            //Пропускаем символы #13 и #10 в конце строки,
            //иначе имя следующего гонщика окажется пустой строкой
            ReadLn(f)
        End;
    Close(f);

    Assign(f, 'race.out');
    Rewrite(f);
    Write(f,name[winner]);
    Close(f)
end.

```

Задачи для самостоятельного решения

1. Составьте программу удаления из строки всех пробелов. Попробуйте это сделать двумя способами: с использованием вспомогательной строки и без неё.
2. Дана строка. Определить, сколько раз в неё входит группа букв **abc**.
3. Результаты вступительных экзаменов представлены в виде списка из N строк, в каждой строке которого записаны фамилия студента и отметки по каждому из M экзаменов. Определить количество студентов, сдавших вступительные экзамены только на «отлично».

Формат входных данных.

В первой строке файла input.txt записаны через пробел натуральные числа N и M , не превышающие 1000. В каждой из следующих N строк записана фамилия студента и через один пробел M отметок по пятибалльной системе без пробелов между ними.

Формат выходных данных

Файл output.txt содержит единственное число — ответ на вопрос задачи.

Пример

input.txt	output.txt
3 4 Иванов 5432 Петров 5555 Сидоров 5455	1