Нестандартные типы данных.

В языке Паскаль существуют типы данных, характеристики которых определяются пользователем, работающим с программой. Такое определение дается в разделе описания типов, начинающегося служебным словом ***type*.**
Необходимость в описании нестандартных типов данных возникает тогда, когда пользователь имеет дело с некоторым набором конкретных значений. Например, наша эра, двадцатый век, цвета радуги, дни недели и т.п.

1. Скалярный или перечислимый тип данных.

При определении перечислимого типа данных в круглых скобках через запятую просто перечисляются все возможные значения переменных этого типа.
*Пример***.**

```
type st_sv=(North, South, East, West);
            month = (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dek);
            den_ned = (pn, wt, sr, ch, pt, sb, ws);
```

Имена значений, перечисленные в списке, есть константы описываемого типа. Поэтому, если в разделе описания переменных описаны переменные
var komp: st_sv; m: month; d: den_ned;
то их значениями могут быть только значения, перечисленные в соответствующем типе. Например:

```
komp:=West;
m:=Nov;
```

Недопустимы операторы присваивания типа: komp:=sb; d:=Feb;

Над переменными перечислимого типа можно выполнять только ***операции отношения***. Значения переменных перечислимого типа считаются упорядоченными в соответствии с естественным порядком их следования в списке описания данного типа. Поэтому отношения
North<South, Apr<Sep, wt<sb
имеют значение *true*, а отношения

```
May>Oct, West< North, ch<pn
```

имеют значение *false*.

Для данных перечислимого типа определены функции ***pred()*** и ***succ()***. Например:
pred(West)=East; succ(sr)=wt;
Первый элемент списка не имеет предшествующего, а последний элемент – следующего, поэтому значение функций pred от первого элемента и succ от последнего не опеределены.

Если к данным перечислимого типа применить функцию ***ord()***, то в результате будет получен порядковый номер значения в списке определения перечислимого типа. Например:
ord(Mar) = 3; ord(ws)=7;

Данные перечислимого типа не могут быть параметрами стандартных процедур ввода и вывода данных (read, readln, write, writeln). Недопустимо использование, например, таких операторов:

```
a:=West;
writeln(a);
```

Но данные перечислимого типа могут быть использованы в операторе цикла с параметром. Например:

```
for i:= Jan to Dec do …
```

Нестандартный тип данных

2. Ограниченный тип данных.

Синонимы: отрезочный, интервальный.

При определении данного типа данных накладываются ограничения на стандартный или уже заданный тип данных, множество значений которого является перенумерованным (integer, char, boolean, перечислимый тип).

Таким образом, из всего множества значений выбранного типа берется некоторый диапазон (или интервал), который задается двумя константами, разделенными двумя точками. Например:

```
type ne = 1..2004;
          l_alph = 'a'..'z';
          polugodie = Jun .. Jun;
```

Константы в определении ограниченного типа задают нижнюю и верхнюю границы отрезка, из которого может принимать значения переменная этого типа.

Определение отрезочного типа верно только тогда, когда выполнено условие:

нижняя граница <= верхняя граница.

Переменные ограниченного типа могут быть описаны так:

```
var year: ne: bukva: l_alph;
```

Эти переменные будут принимать значения того же типа, что и константы, ограничивающие отрезок.

К переменным ограниченного типа применимы операции и функции, которые применимы к данным тех типов, к которым относятся константы, ограничивающие диапазон значений.

3. Пример использования нестандартных типов данных.

Составить программу определения победителя шахматного турнира, в котором участвовало 6 человек, каждый из которых сыграл по 1 партии с каждым.

```
Program turnir;
    uses crt;
    type fam = (Ivanov, Pavlov, Kotov,
Sidorov, Petrov, Vasin);
    var win, uch: fam; k, max: integer;
Begin
    clrscr;
    max:=-1;
    i:=1;
    writeln('Итоги турнира');
    for uch:=Ivanov to Vasin do
       begin
          write ('Сколько очков набрал ', i,
'-й участник?');
          readln(k);
          if k>max then
             begin
                win:=uch;
                max:=k
             end;
          i:=i+1;
       end;
    write('');
    case win of
       Ivanov: writeln('Иванов');
       Pavlov: writeln('Павлов');
       Kotov: writeln('Котов');
       Sidorov: writeln('Сидоров');
       Petrov: writeln('Петров');
       Vasin: writeln('Васин');
    end;
    readln;
End.
```