Лекция 2. Обзор языка C, продолжение

Напоминаем как знакомиться с новым алгоритмическим языком:

0. идеология языка, что делает программа и как она строится;
1. базовые типы данных и переменные;
2. операции с базовыми типами, выражения;
3. формирование производных типов данных;
4. операторы и управляющие конструкции;
5. структуризация записи программы (блоки, объявления, процедуры , контексты и т.п.);
6. видимость объектов (локальные, глобальные, статические, автоматические и т.д.);
7. организация ввода-вывода, внешние и стандартные библиотеки;
8. прочие специфические особенности и возможности;
9. запуск и отладка программы в конкретной системе программирования.

3. Простые переменные и составные типы

```
простые переменные
int x, y, z;                - значения могут быть не определены
                              но иногда могут быть = 0   (обсудим позже)
int i, j, k, _a, d_35;
double f, u123, result;
char r2d2, c3po;
float val;
unsigned long abc, size;
signed short m;                                  j = m;

массивы
                                                !!!! int k;    k = 1000;
int a[100];             a[0] a[1] ... a[99]     !!!! error    z = a[k] + 1;
double b, c[1024];      c[0] ... c[1023]        !!!! a[100] - в массиве нет такого элемента

многомерные массивы
int a[100][20], b[3][5][10];      a[i][j]  b[i][j][k]  все индексы от 0 до n-1
тут есть специфика, о которой пока тоже не говорим ...

!!! проверка индексов на попадание в границы не делается !!!

иициализация в объявлении
int x = 10, y = -3;          - значения определены
double e = 2.71828;
int r[4] = {3, 2, -7, 6};
double a[] = {3.3, 4.1, -2.22222};   создается массив на указанное количество
                                     значений, т.е. здесь на 3
```

Есть еще структуры, объединения, битовые поля — о них будет отдельный разговор.

4. Операторы и управляющие конструкции.

Оператор присваивания:

```
lvalue = rvalue;     (выражение, имеюшее значение)
a[2*k+1] = a[2*k-1] + 3.14*r;
b = c = d = 1;                  a = (b=3) + 2;      b = 3;
                                                    a = 5;
```

Разветвления:

```
if (выражение)          if (выражение)        можно вкладывать
{                       {                     друг в друга
   операторы-true          операторы-true
} else {                }
   операторы-false
}

if (x < 3)              if ( x >= 0)        ловушка для новичка:
{                       {                   if (x==1) { .... }
   y = 1;                  y = x;           if (x=1) { .... }
} else {                } else {
   z = 2*x + 5;            y = -x;
}                       }                    if(x) { a = 1; }
```

Условное выражение:

```
(выражение_0) ? выражение_1 : выражение_2

y = (x>=0) ? x : -x;

  if (x>=0) {
     y = x;
   } else {
     y = -x;
   }


double abs(double x)
{
   if (x>=0) {
     return x;
   } else {
     return -x;
   }
}
y = abs(x);
```

Переключатель:

```
switch (целое выражение)            switch (k + m)
{                                   {
   case константа1 :                   case 0:
```

```
      операторы
      break;
   case константа2 :
      операторы
      break;
   ...

   ...
   ...
   default:
      операторы
}
```

```
      y = 0;
      break;
   case 25:
      y = 1;
      break;
   case 100:
      k = 333;
   case 200:
   case 300:
      y = 2;
      break;
   default:
      y = -1;
}
```

Циклы:

```
while ( выражение )
{
    тело цикла
}
```

```
int a[10], i = 0;
while (i < 10)
{
    a[i++] = 0;
}
```

```
while(true)
{

}
```

```
do
{
    тело цикла
} while (выражение);
```

```
int a[10], i = 10;
do
{
   a[--i] = 0;
} while(i);
```

Цикл for:

```
for (выражение1; выражение2; выражение3)
{
    тело цикла
}
```

эквивалентно

```
выражение1;
while(выражение2)
{
    тело цикла
    выражение3;
}
```

```
for (i=0; i<n; i++)
for (i=n; i>0; i--)

for (i=1; i<n; i=fun(i))

for (i=1, j=n; i<j; i++, j--)
```

```
break           - выход из цикла
continue        - переход к следующему шагу цикла
```

```
for (i=0; i<n; i++)
{
    a = ....
    if (a<0) continue;
    b = ....
}
```

```
k = 0;
while(k<100)
{
    a = .....
    if (a < 0) break;
    k++;
}
```