Лекция 3. Обзор языка С, продолжение

5. Структуризация записи программы (блоки, объявления, процедуры , контексты и т.п.)

Основные понятия:

```
блок    {...}
{
   .....
}
```

Функция:

```
тип возвр.значения  имя (список параметров)      // это заголовок функции
{
   тело функции
}

сигнатура:  имя функции, типы параметров и их порядок в заголовке

double fun(int x, double y)    // определение функции
{
   double z;                   // локальные переменные
   z = x + y;                  // содержательные операции
   return 2*z;                 // возврат значения
}
....
int a,b,c;
double k,l,m;
....
k = fun(a, m);
l = fun(a, m) + 3*fun(b,c);
```

Объявления переменных:

```
int x;
double y = 0;
```

Объявления функций (проотип):

```
double fun(int x, double y);     // заголовок функции
int g(int, int, char);           // можно без имен аргументов

void    - "отсутствие"
void  AnotherFunction (int x);
int   OneMoreFunction (void);
```

Программа — последовательность объявлений функций и пременных и определений функций.

Объявления задают области видимости переменных и функций.

Начальная функция (точка входа) — функция main — с нее начинается выполнение программы

```
int main(void)          // есть и другие варианты заголовка ...
```

Программа может быть записана в нескольких файлах.

Блок {} задает контекст блока.

Файл задает контекст файла.

Объявления и определения действуют в пределах контекста.

Мы сразу будем рассматривать программу в нескольких файлах (в учебных целях, хотя иногда без этого можно обойтись).

6. Видимость объектов (локальные, глобальные, статические, автоматические и т.д.).

Объявления задают область видимости переменной или функции в пределах контекста всюду после этого объявления.

```
{          // блок                    ........ файл
  int a;                              .  int b;
  ....                                .  ....
}                                     .  ....
                                      .
                                      ........ конец файла
```

```
   file1              file2            file3            file4
<объявление>       <объявление>     <объявление>     <объявление>
<функция>          <объявление>     <объявление>     <функция>
<объявление>       <объявление>     <объявление>     <функция>
<функция>          <функция>        <объявление>     <функция>
<функция>          <функция>
<функция>
```

Переменные и функции определяются в контесте файла только а одном месте (в одном файле).

```
   file1              file2
   int a;             int a;
так нельзя - множественное определение
```

Чтобы расширить область видимости переменной или функции в другой файл, там следует записать их объявление со словом extern. Для объявления функций слово extern можно не писать.

```
  file1              file2            file3
  int a;           extern int a;    int f(int x) {...}
int f(int);        здесь f не видно здесь f видно
здесь a видно      здесь a видно    здесь a не видно
здесь f видно
```

Ключевое слово static при определении переменной или функции в контексте файла ограничивает область видимости только этим файлом.

```
  file1              file2
static int a;      static int a;
здесь a свое       здесь a свое
и не смешивается с a из другого файла
```

Ключевое слово static при определении переменной в контексте блока сохраняет эту переменную при выходе из блока. Вне блока эта переменная не видна, но при повторном входе в блок ей можно продолжать пользоваться, и она сохраняет свои значения с предыдущего посещения блока.

Простейший ввод и вывод в C и C++

`stdio.h` — заголовочный файл (объявления) функций стандартной библиотеки ввода/выводв (язык C)
`iostream` — заголовочный файл (объявления) функций потоковой библиотеки ввода/выводв (язык C++)

По сути очень близки, но у каждой есть свои удобства и неудобства.

Концепция:

stdio — вызов функций ввода/вывода, параметры задают:
- где выполняется ввод/вывод
- формат (преобразования значений в печатные символы при выводе и символов в значеия при вводе)
- значения для вывода и указатели на переменные для ввода

iostream
- объект “поток”, который отвечает за ввод/вывод
- предоставляет разумную форму ввода/вывода по умолчанию
- можно настраивать форму ввода/вывода
- запись операции как последовательность компонент, передаваемых в поток (извлекаемых из потока).

Консольный ввод/вывод (экран и клавиатура). Ввод/вывод с файлами.

Стандартные каналы (потоки)

C: stdin stdout stderr

C++: cin cout cerr

`using namespace std;` — чтобы не писать `std::cin` и т.п.

```
Функции stdio              операторы iostream
```

```
    scanf                       cin >>
    printf                      cout <<
и другие аналогичные

 Спецификации преобразования stdio      методы и манипуляторы  iostream

      %d        (decimal)
      %u        (unsigned)
      %x        (hexadecimal)               hex
      %c        (char)
      %s        (string)
      %f  %lf   (float, double)
      %e  %le   (exponential)               scientific
      %g  %lg
      ...
                                                                  1.23
      %N...       ширина                    width(N)            1.234566
      %.M         точность                  precision(M)
                                            fixed

     \n                                     endl
     \t  \r
```

Пример iocompare.cpp с пояснениями в ходе лекции. Разбирается несколько простейших образцов ввода/вывода строк, целых и вещественных чисел, а также ввод и вывод с файлами в простейшем варианте.