Операторы языка C++
Структура программы

Пустой оператор – `;`

Составной оператор – `{…}`

```
try { <операторы> }
catch (<объявление исключения>)
 { <операторы> }
...
catch (<объявление исключения>)
 { <операторы> }
```

```
if (<выражение>)
 <оператор1>
[else
 <оператор2>]
```

```
switch (<выражение>)
  {  case <константное выражение1>: <операторы1>
     case <константное выражение2>: <операторы2>
     …
     case <константное выражение N>: <операторы N>
     [default: <операторы>]
   }
```

```
int n = 1;
int a = 2;

switch (n)
 { case 1: a += 3;
   case 2: a *= 5;
  }
```

```
switch (<выражение>)
  {  case <константное выражение1>: <операторы1>
     case <константное выражение2>: <операторы2>
     …
     case <константное выражение N>: <операторы N>
     [default: <операторы>]
   }
```

```
int n = 1;
int a = 2;

switch (n)
 { case 1: a += 3; break;
   case 2: a *= 5; break;
  }
```

```
while (<выражение>) <оператор>

do <оператор> while (<выражение>);
```

```
for ([<начальное выражение>];
       [<условное  выражение>];
     [<выражение приращения>])
 <оператор>
```

```
for (int i = 0; i < n; i++) …

for (int i = 0, j = 0;
          i < m && j < n; i++, j += 3) …

for ( ; i < n; i++) …

for ( ; f(x) > g(y); ) …

for (int i = 0; ; ) …

for ( ; ; ) …
```

break;

```
for (int i = 0; i < m; i++)
  for (int j = 0; j < n; j++)
    { if (x[i][j] == 0)
        break;
      …
     }
```

continue;

```
for (int i = 0; i < n; i++)
  { if (i == 5) continue;
    …
    …
    …
   }
```

```
return [<выражение>];
```

Программа на языке C++ состоит из:

- директив препроцессора,
- указаний компилятору,
- объявлений переменных и/или констант,
- объявлений и определений функций.

```
<тип> <имя> [= <инициализатор>]
    [,<имя> [= <инициализатор>] ...];


    int    x;

    int    a, b = 0;

    int    a = 0, b = 0;

    double y = exp(1);
```

```
// a является константой
const int a   = 100;

// Все b[i] являются константами
const int b[] = {1, 2, 3, 4, 5};

// Ошибка – нет инициализатора
const int c;
```

```
typedef <имя типа> <новое имя типа>;

typedef unsigned char uchar;
```

```
<тип> <имя> (<список формальных параметров>)
 {
    [<объявления>]
    [<операторы>]
  }


<тип> <имя> (<список формальных параметров>);
```

```
double Cube(double x); // Прототип

void main()
 {
   printf("%lf\n", Cube(5));
  }

double Cube(double x) // Определение
 {
   return x * x * x;
  }
```

```
// Ошибка - не возвращается значение
int  f1() { }
// Правильно
void f2() { }

// Правильно
int  f3() { return 1; }
// Ошибка - значение возвращается в функции void
void f4() { return 1; }

// Ошибка - не указано возвращаемое значение
int  f5() { return; }
// Правильно
void f6() { return; }
```

```
void g() { ... }


// Правильно - возвращается «никакое значение»
void h() { ...  return g(); }
```

```
inline int max(int x, int y)
 { return x > y ? x : y; }
```

```
// Ошибка
int g(int m = 1, int n);

int f(int m = 1, int n = 2);

// Ошибка – повтор параметров по умолчанию
int f(int m = 1, int n = 2) { ... }

// Ошибка – изменение параметров по умолчанию
int f(int m = 0, int n = 0) { ... }

// Правильно
int f(int m, int n) { ... }

// Вызов функции с двумя параметрами
f(5, 6);

f(5);       // Эквивалентно f(5, 2);
f();        // Эквивалентно f(1, 2);
```

```
void main(int argc, char *argv[])
 { ... }
```

```
#include <имя_файла>

#include "имя_файла"
```

```
#define <идентификатор> <текст>

#define <идентификатор>(<список
параметров>) <текст>


#undef <идентификатор>
```

```
#define N 100
#define MULT(a, b) ((a) * (b))
#define MAX(x, y)  ((x) > (y)) ? (x) : (y)


MULT(x + y, z)
((x + y) * (z))


MAX(i, a[i++])
((i) > (a[i++])) ? (i) : (a[i++])
```

```
#include <cstdio>
#include <math.h>

double f(double x) { return sin(2 * x) + cos(x / 5); }

void main()
 { double a, b, e, x, c, fa, fc;
   int n;

   printf("Введите границы отрезка и точность: ");
   scanf("%lf%lf%lf", &a, &b, &e);
   for (n = 0; fabs(a - b) > e; n++)
    { c  = (a + b) / 2;
      fa = f(a);
      fc = f(c);
      if (fa * fc < 0)
       b = c;
      else
       a = c;
     }
   x = (a + b) / 2;
   printf("Корень уравнения = %lf\nЧисло итераций   = %d\n", x, n);
  }
```

```
#include <cstdio>
#include <math.h>

double f(double x) { return sin(2 * x) + cos(x / 5); }

void main()
 { double a, b, e, x, c, fa, fc;
   int n;

   printf("Введите границы отрезка и точность: ");
   scanf("%lf%lf%lf", &a, &b, &e);
   n = 0;
   while (fabs(a – b) > e)
    { c  = (a + b) / 2;
      fa = f(a);
      fc = f(c);
      if (fa * fc < 0)
       b = c;
      else
       a = c;
      n++;
     }
   x = (a + b) / 2;
   printf("Корень уравнения = %lf\nЧисло итераций   = %d\n", x, n);
  }
```

```
#include <cstdio>
#include <math.h>

double f(double x) { return sin(2 * x) + cos(x / 5); }

void main()
 { double a, b, e, x, c, fa, fc;
   int n;

   printf("Введите границы отрезка и точность: ");
   scanf("%lf%lf%lf", &a, &b, &e);
   for (n = 0;
        fabs(a – b) > e;
        c  = (a + b) / 2, fa = f(a), fc = f(c),
        fa * fc < 0 ? b = c : a = c, n++) ;
   x = (a + b) / 2;
   printf("Корень уравнения = %lf\nЧисло итераций   = %d\n", x, n);
  }
```

```
#include <cstdio>
#include <stdarg.h>

void print(char *format, ...);

void main()
 { int    a = 45, b = 87;
   double f = 2.75;

   print("dfd", a, f, b);
  }
```

```
void print(char * format, ...)
 { va_list list;
   int    n, i;
   double f;

   va_start(list, format);
   for (i = 0; format[i]; i++)
    switch(format[i])
     { case 'd':
        n = va_arg(list, int);
        printf("%d\n", n);
        break;
       case 'f':
        f = va_arg(list, double);
        printf("%lf\n", f);
        break;
     }
   va_end(list);
  }
```