Sequence Points in C | Set 1

In this post, we will try to cover many ambiguous questions like following.

Guess the output of following programs.

```
// PROGRAM 1
#include <stdio.h>
int f1() { printf ("Geeks"); return 1;}
int f2() { printf ("forGeeks"); return 1;}
int main()
{
  int p = f1() + f2();
  return 0;
}

// PROGRAM 2
#include <stdio.h>
int x = 20;
int f1() { x = x+10; return x;}
int f2() { x = x-5;  return x;}
int main()
{
  int p = f1() + f2();
  printf ("p = %d", p);
  return 0;
}


// PROGRAM 3
#include <stdio.h>
int main()
{
   int i = 8;
   int p = i++*i++;
   printf("%d\n", p);
}
```

Run on IDE

The output of all of the above programs is undefined or unspecified. The output may be different with different compilers and different machines. It is like asking the value of undefined automatic variable.

The reason for undefined behavior in PROGRAM 1 is, the operator '+' doesn't have standard defined order of evaluation for its operands. Either f1() or f2() may be executed first. So output may be either "GeeksforGeeks" or "forGeeksGeeks". Similar to operator '+', most of the other similar operators like '-', '/', '*', Bitwise AND &, Bitwise OR |, .. etc don't have a standard defined order for evaluation for its operands.

Evaluation of an expression may also produce side effects. For example, in the above program 2, the final values of p is ambiguous. Depending on the order of expression evaluation, if f1() executes first, the value of p will be 55, otherwise 40.

The output of program 3 is also undefined. It may be 64, 72, or may be something else. The subexpression i++ causes a side effect, it modifies i's value, which leads to undefined behavior since i is also referenced elsewhere in the same expression.

Unlike above cases, *at certain specified points in the execution sequence called sequence points, all side effects of previous evaluations are guaranteed to be complete*. A sequence point defines any point in a computer program's execution at which it is guaranteed that all side effects of previous evaluations will have been performed, and no side effects from subsequent evaluations have yet been performed. Following are the sequence points listed in the C standard:

— The end of the first operand of the following operators:

a) logical AND &&

b) logical OR ||

c) conditional ?

d) comma ,

For example, the output of following programs is guaranteed to be “GeeksforGeeks” on all compilers/machines.

```
// Following 3 lines are common in all of the below progra
#include <stdio.h>
int f1() { printf ("Geeks"); return 1;}
int f2() { printf ("forGeeks"); return 1;}

// PROGRAM 4
int main()
{
   // Since && defines a sequence point after first operai
   // guaranteed that f1() is completed first.
   int p = f1() && f2();
   return 0;
}

// PROGRAM 5
int main()
{
   // Since comma operator defines a sequence point after
   // guaranteed that f1() is completed first.
  int p = (f1(), f2());
  return 0;
}


// PROGRAM 6
int main()
{
   // Since ? operator defines a sequence point after firs
   // guaranteed that f1() is completed first.
  int p = f1()? f2(): 3;
  return 0;
}
```

Run on IDE

— The end of a full expression. This category includes following expression statements

a) Any full statement ended with semicolon like “a = b;”

b) return statements

c) The controlling expressions of if, switch, while, or do-while statements.

d) All three expressions in a for statement.

The above list of sequence points is partial. We will be covering all remaining sequence points in the next post on Sequence Point. We will also be covering many C questions asked in forum (See this, this, this , this and many others).

References:

http://en.wikipedia.org/wiki/Sequence_point

http://c-faq.com/expr/seqpoints.html

http://msdn.microsoft.com/en-us/library/d45c7a5d(v=vs.110).aspx

http://www.open-std.org/jtc1/sc22/wg14/www/docs/n925.htm

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.