PEN203

C++ Functions

C++ How to Program
Deitel & Deitel

Outline

- Program Modules in C++
- Math Library Functions
- Functions
- Function Definitions
- Function Prototypes
- Calling Functions: Call by Value and Call by Reference
- Random Number Generation
- Recursion

Review-Calling Functions: Call by Value and Call by Reference

- Call by value
  - A copy of the argument is created and passed to function.
  - Modifications performed in function do not effect the original value.
- Call by reference
  - Original argument passed to function
  - Modifications in function effect the original value.

Random Number Generation

- rand function is defined in <stdlib.h>
- rand returns a random number between 0 and RAND_MAX
- To produce a random number between 1 and n

  1 + (rand() % n) expression can be used.

  rand() % n returns a number between 0 and n-1

Random Number Generation

- srand function is defined in <stdlib.h>
- It takes an integer seed and jumps to that location in its random sequence

  srand(seed)
- srand(time(NULL));

  time(NULL) returns the number of seconds since January 1, 1970 and therefore randomizes the seed.

Random Number Generation

```
1  // Fig. 3.7: fig03_07.cpp
2  // Shifted, scaled integers produced by 1 + rand() % 6.
3  #include <iostream>
4
5  using std::cout;
6  using std::endl;
7
8  #include <iomanip>
9
10 using std::setw;
11
12 #include <cstdlib>   // contains function prototype for rand
13
14 int main()
15 {
16   // loop 20 times
17   for ( int counter = 1; counter <= 20; counter++ ) {
18
19     // pick random number from 1 to 6 and output it
20     cout << setw( 10 ) << ( 1 + rand() % 6 );
21
22     // if counter divisible by 5, begin new line of output
23     if ( counter % 5 == 0 )
24       cout << endl;
25
26   } // end for structure
```

Random Number Generation

```
27
28      return 0;  // indicates successful termination
29
30   } // end main
```

```
6     6     5     5     6
5     1     1     5     3
6     6     2     4     2
6     2     3     4     1
```

Recursion

- Recursive functions call themselves.
- A base case need to be provided.
- Example:

  5! = 5 * 4 * 3 * 2 *1

  5! = 5 * 4!

  4! = 4 * 3! ...

  Base case (1! = 0! = 1)

Recursion

```
1  // Fig. 3.14: fig03_14.cpp
2  // Recursive factorial function.
3  #include <iostream>
4
5  using std::cout;
6  using std::endl;
7
8  #include <iomanip>
9
10 using std::setw;
11
12 unsigned long factorial( unsigned long ); // function prototype
13
14 int main()
15 {
16    // Loop 10 times. During each iteration, calculate
17    // factorial( i ) and display result.
18    for ( int i = 0; i <= 10; i++ )
19       cout << setw( 2 ) << i << "! = "
20            << factorial( i ) << endl;
21
22    return 0;  // indicates successful termination
23
24 } // end main
```

Recursion

```
25
26  // recursive definition of function factorial
27  unsigned long factorial( unsigned long number )
28  {
29     // base case
30     if ( number <= 1 )
31        return 1;
32
33     // recursive step
34     else
35        return number * factorial( number - 1 );
36
37  } // end function factorial
```

```
 0! = 1
 1! = 1
 2! = 2
 3! = 6
 4! = 24
 5! = 120
 6! = 720
 7! = 5040
 8! = 40320
 9! = 362880
10! = 3628800
```