16.216: ECE Application Programming

Summer 2012

Lecture 6: Key Questions
July 27, 2012

In today's exercise, you will write a program that does the following:

- Prompts the user to enter a single input character followed by an integer, `n`.
  - If not correctly formatted, print error, clear rest of line, and repeat prompt
- Depending on the character entered, do the following:
  - '`F`' or '`f`': Compute and print the factorial of `n`, `n!`
    - For example, if the user enters **`F 5`**, print **`5! = 120`**
  - '`P`' or '`p`': Compute 2^n, but only if `n >= 0`.
    - For example, if the user enters **`p 2`**, print **`2^2 = 4`**
    - Print an error if `n < 0`.
  - '`X`' or '`x`': Exit the program
  - In all other cases, print an error:
    - For example: `Invalid command Z entered`
- If the user enters any command other than '`X`' or '`x`', return to the initial prompt and repeat the program.

Steps in the programming exercise:

1. Draw a general flowchart for the overall program flow.
   - Treat each of the processes listed in part 2 as a single block—don't worry about the details just yet.
2. Draw smaller flowcharts for each of the following processes:
   - Reading the input character & integer until correct.
   - Computing n!
   - Computing 2^n if n >= 0 and printing an error otherwise.
3. Convert the flowcharts to actual code (time permitting). Note that:
   - If we don't complete this part today, I'll post the solutions.
   - There are a couple of different ways to build up your larger program, but it might be advisable to test each of the smaller processes individually, then put them together with the larger, overall program.

Space to draw flowchart of overall program flow:

Flowchart for reading input character until correct:

Flowchart for n!

Flowchart for 2^n

1. Describe the basic use and syntax of functions, including return types and arguments.

2. What is a function prototype? When and why is it necessary?

3. Explain the idea of scope and how it relates to functions. Also, explain what happens when function arguments are passed by value.

You may wish to refer to the following example:

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

double hyp(double a, double b);

void main()
{
  double x,y,h;
  printf("Enter two legs of triangle: ");
  scanf("%lf %lf",&x,&y);
  h=hyp(x,y);
  printf("Trgle w legs %lf and %lf has hyp of %lf\n",
          x,y,h);
}

double hyp(double a, double b)
{
  double sum, result;
  sum = a*a + b*b;
  result = sqrt(sum);
  return result;
}
```

4. **Example:** What does the following program print?

```
#include <stdio.h>

int f(int a, int b);              // Function prototype

int main() {
      int x = 1;
      int y = 2;
      int result1, result2, result3;

      result1 = f(x, y);
      result2 = f(y, result1);
      result3 = f(result1, result2);

      printf("x = %d, y = %d\n", x, y);
      printf("Result 1: %d\n", result1);
      printf("Result 2: %d\n", result2);
      printf("Result 3: %d\n", result3);

      return 0;
}

int f(int a, int b)         // Function definition
{
      int i;                // Loop index
      int r = 0;            // Result

      for (i = 0; i < a; i++)
            r += b;

      return r;
}
```

5. **Example:** Write a function to do each of the following:

a. Prints a series of LINE_LENGTH dashes on a single line, where LINE_LENGTH is a predefined constant (using #define)

b. Reads a value from the console input and returns 1 if the value is even, 0 if it's odd

c. Takes four numbers as arguments and returns their average