Discussion 3

Aside: using increments properly

- we know that i++ ⇔ i = i+1
- int i = 5;

  int a = i++; //x = 6, a = 5
- int i = 5;

  int a = ++i; //x = 6, a = 6
- for loops: doesn’t matter which you use:
  - ***for (i = 0; i<10; i++)*** same as ***for (i = 0; i<10; ++i)***

Switch - syntax

```
syntax: switch (expression)
            {
               case constant:
                    statements;
               default:
                    statements;
             }
```

Switch Example: Character counting

```
char c; //c = ‘a’ or c = ‘A’
int cap =0; int lower = 0; int total = 0;
switch (c)
{
  case ‘A’:
        cap++; break; //break vs no break
   case ‘a’:
        lower++; break; //break vs no break
   default:
        total++;
}
```

Switch Example: grades

```
char grade;
switch (grade)
{
   case ‘A’:
   case ‘B’:
   case ‘C’: // A. B. C are all passing grades
       printf(“Pass”);
       break;
   case ‘D’:
   case ‘F’:
        printf(“Fail”); //D, F are failing grades
}// no default
```

Function Definition

- return type, name, input parameters
- parameters vs arguments
- prototype functions vs defining the function
- how to call functions using arguments and assign return values

Function Definition

```
#include <stdio.h>
int add (int x); //prototype
int main()
{ int num = 2;
   int var = add(num);
   return 0; }
int add (int x)
{ return x+2; }
```

Function Definition

```
#include <stdio.h>
#include “genlib.h”
#include “simpio.h”
void printGrade (char c); //prototype
int main ()
{ char a; int i;
  for (i = 0; i<10; i++)
     a = getChar();
     printGrade(a);
     return 0;
}
```

```
void printGrade (char c)
{
    switch (c)
    {
         case ‘A’:
            printf(“Excellent”); break;
          case ‘B’:
             printf(“Great”); break;
          case ‘C’:
              printf(“Good”); break;
          case ‘D’:
          case ‘F’:
               printf(“Fail”); break;
          default:
                printf(“Invalid”);
    }
```

//printGrade written to the right due to lack of space