Functions (cont.)

Example Program

```
/* This program finds the factorial value of a given
 * positive integer
 */

#include <stdio.h>

int Factorial(int n);

int main(void)
{
    int num;

    printf(“Enter a positive integer: ");
    scanf(“%d”,&num);
    printf("Given positive integer: %d, “, num);
    printf(“its factorial value is : %d \n“, Factorial(num));
    return 0;
}
```


Function `Factorial()`

```
/* Function: Factorial
 * This function computes and returns the factorial
 * value of its parameter. The function assumes that
 * the parameter is a positive integer.
 */
int Factorial (int num)
{
    int counter = 2;
    int factVal = 1;

    while (counter <= num) {
      factVal = factVal * counter;
      counter = counter + 1;
    }
    return factVal;
}
```


More Examples

```
/* Function: Comp_Grade
 * It returns a numerical value in between 0 and 4
   corresponding to
 * a given letter grade. If the grade is invalid  -1 will be
   returned.
 * Input: a single character. The character MUST BE one of
   the following: ‘A’,‘B’,‘C’,‘D’, or ‘F’.
 * Output: a numerical value.
 */
 int Comp_Grade(char grade) {
   if (grade == ‘A’) return 4;
   else if (grade == ‘B’) return 3;
   else if (grade == ‘C’) return 2;
   else if (grade == ‘D’) return 1;
   else if (grade == 'F') return 0;
   else return -1;
}
```

Void functions

A void function that prints out a menu:

```
void menu() {

   printf("Please choose one of the following.\n");
   printf("\t 1. Square\n");
   printf("\t 2. Rectangle\n");
   printf("\t 3. Circle\n");
   printf("\t 4. Quit\n");
   printf("\n\n\t Enter your choice: ");

}
```

You can call this function as follows:

```
menu();
```


- Another situation where void functions might be useful is a menu driven program where the menu choices are completely unrelated. Here is a skeleton of the main program of such a function:

```
int main()
{  int choice;
   menu();
   scanf("%d", &choice);
   while (choice != 4) {
     if (choice == 1)
        function1();
     else if (choice == 2)
        function2();
     else if (choice == 3)
        function3();
     else if (choice != 4)
        printf("Sorry, please enter your choice again.\n");
     menu();
     scanf("%d", &choice);
   }
}
```


Programming Exercise

- Write a program that that reads in the side of a square and then prints a hollow square.Your program should work for squares of all side sizes between 1 and 20. For example, if your program reads a size of 4, it should print:

```
****
*  *
*  *
****
```



```
/* This program reads in the size of a square and draws
 * the square on the screen. The program validates that
 * the user enters a side size between 1 and 20.
 */

#include <stdio.h>

/* Function prototypes */
int GetSide();
void draw_square(int);

int main(void)
{
    int size;

    size = GetSide();      /* Read the side length */
    draw_square(size);     /* Draw hallow square   */

}
```

```
/* Function: GetSide
 * This function reads an integer representing the side
 * of a square. The function verifies that the input
 * value is between 1 and 20. Once input is correct
 * it is returned to the calling environment.
 */
int GetSide()
{  int side;

   printf("Enter the side of the square (1..20): ");

   do {
      scanf("%d",&side);
      if (side < 1 || side > 20){
         printf(“Invalid input.\n”);
         printf("Enter a number between 1 and 20:");
      }
   } while (side < 1 || side > 20);

   return side;
}
```



```
/* Function: draw_square
 * This function draws a hallow square on the screen
 * given its side length.
 */
void draw_square(int number)
{
   int i, j;
   for(i=1; i<=number; i++){
      for(j=1; j<=number; j++){
         if (j==1 || j== number || i==1 || i== number)
              printf("*");
          else
              printf(" ");
       }
       printf("\n");
    }
    return;
}
```