To:
Tenouk

C LAB WORKSHEET 13a
C & C++ Functions Part 5: With Return Values

1. Functions and array data types.
2. More on C/C++ functions that receive and/or return values.
3. Tutorial references that should be used together with this worksheet are: C and C++ function part 1, function part 2, function part 3 and function part 4.
4. For main() function complete story please refer to C and C++ main() story.

3. The following example shows two ways to call Average() function. Show the output and answer the questions.

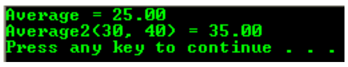
```
#include <stdio.h>

float Average2(int, int);

void main(void)
{
   float Avg;

   Avg = Average2(30, 20);

   printf("Average = %.2f\n", Avg);
   printf("Average2(30, 40) = %.2f\n", Average2(30, 40));
}

float Average2(int x, int y)
{
   return (float)((x+y) / 2.0);   // Line 1
}
```

a. How many times does main() call Average2()?
b. Is the function assigned to a variable the first or the second time that Average2() is called?
c. What happens to the value returned from the function when it is called the first time?
d. What about the second time, when Average2() is called? What happens to the value returned from that function if it is not assigned to Avg?
e. Add this line between the printf()'s in the main().

   Average2(10, 50);

   Does the program run? If so, does it print the average of 10 and 50? Why or why not?

f. When calling a function that returns a value, it should be assigned to a variable or printed. True or false?
g. In Line 5 exercise 2, the value of the function is not assigned or printed in main(). Why not?
h. Replace Line 1 in Average2() with printf("(x+y) / 2.0 = %.2f\n", (x+y)/2.0);. Are there any problems? Why?



```
Average = 25.00
Average2(30, 40) = 35.00
Press any key to continue . . .
```

a. Two times.
b. The first one in main().
c. The value has been assigned to Avg variable.
d. The value got printed or it becomes an argument to printf().
e. The program runs smoothly. However it does not print the average of 10 and 50 because the returned average value does not assigned to any variable to be used later or as an argument to any function.
f. False. Not really.
g. It got printed in the function itself.
h. Yes there is a warning: "warning C4716: 'Average2' : must return a value". The Average2() has been declared to return a value of type float.

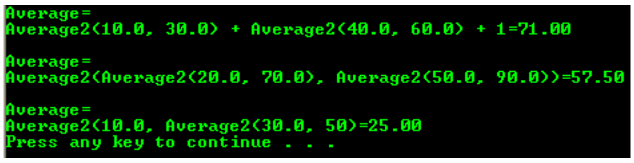
4. In the following exercise, there are other methods for calling Average2(). Watch the parentheses. In this exercise there are three printf()'s. Show the output and answer the questions.

```
#include <stdio.h>

float Average2(float, float);

void main(void)
{
   float Avg;

   Avg = Average2(10.0, 30.0) + Average2(40.0, 60.0) + 1;
   printf("Average = Average2(10.0, 30.0) + Average2(40.0, 60.0) + 1 = %.2f\n", Avg);

   Avg = Average2(Average2(20.0, 70.0), Average2(50.0, 90.0));
   printf("Average = Average2(Average2(20.0, 70.0), Average2(50.0, 90.0)) = %.2f
\n", Avg);

   printf("Average = Average2(10.0, Average2(30.0, 50) = %.2f\n", Average2(10.0,
Average2(30.0, 50)));
}

float Average2(float x, float y)
{
   return (float)((x+y) / 2.0);
}
```

a. When evaluating the value of Avg for the first time, the average of 10.0 and 30.0 is returned. What is its value?
b. The average of 40.0 and 60.0 is also returned. What is its value?

How is the value of Avg obtained?

c. The second time that Avg is assigned a value, the average of 20.0 and 70.0 is received first, then the average of 50.0 and 90.0 is received. How is the value of Avg obtained?
d. Can you have a function inside another function? If so, which function's result is obtained first, the nested one or the outer one?
e. Are the values received from the nested call (such as the value received from the average of 20.0 and 70.0) used to call the outer one?
f. In the last printf(), printf() calls which function? This function calls which function? Notice that there are multiple nestings of function calls.
g. How is the value received for the printf() to print in the last statement in main()?

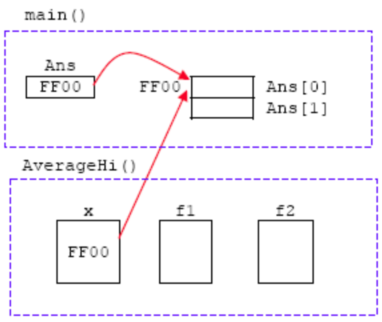


```
Average=
Average2(10.0, 30.0) + Average2(40.0, 60.0) + 1=71.00

Average=
Average2(Average2(20.0, 70.0), Average2(50.0, 90.0))=57.50

Average=
Average2(10.0, Average2(30.0, 50)=25.00
Press any key to continue . . .
```

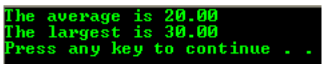
a. 20.00.
b. 50.00. The value of Avg=20.00+50.00+1=71.00.
c. Avg=Average2(45,70)=57.50.
d. Yes we can. The nested result is obtained first. If there are more than one nested functions the evaluations are from the innermost to the outermost.
e. Yes.
f. printf() calls Average2() and Average2() calls itself.
g. The printf() calls Average2(). Inside the Average2() there is another call to itself, Average2(). The result from the final Average2() has been used for the outer Average2(). Then the returned result from this Average2() has been used as a printf() argument.

5. Function can't return more than one value but it can choose from several values to be returned. In the following program AverageHi() calculates the average and the larger of two numbers and makes these two values available to main() via an array. In the provided diagram, the address FF00 is fictitious. Fill in the values in as many places in the diagram as you can. Don't forget to show the output before you answering the questions.

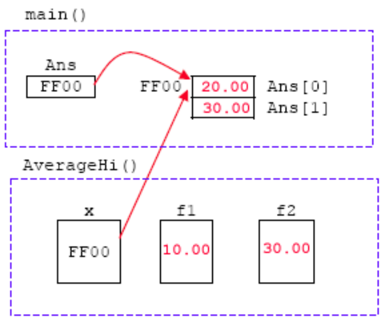
```
#include <stdio.h>

void AverageHi(float, float,  float[ ]);

void main(void)
{
   float Ans[2];

   AverageHi(10.0, 30.0, Ans);
   printf("The average is %.2f\n", Ans[0]);
   printf("The largest is %.2f\n", Ans[1]);
}

void AverageHi(float f1, float f2, float x[2])
{
   x[0] = (f1 + f2) / 2;
   if(f1 > f2)
         x[1] = f1;
   else
         x[1] = f2;
}
```



a. How many scalars and how many arrays does main() pass to AverageHi()?
b. What are the values of the scalars in main()? In AverageHi()?
c. What is the name of the array in main()? In AverageHi()?
d. AverageHi() calculates the average of f1 and f2. In which index of the array is it placed?
e. AverageHi() calculates the larger of f1 and f2. In which index of the array is it placed?
f. Does AverageHi() calculate two answers, the average and the largest and make them available to main()?
g. Since a return is not used in Average(), how are these two answers provided for main()?



```
The average is 20.00
The largest is 30.00
Press any key to continue . .
```



a. Two scalars and one array.
b. Both in main() and AverageHi() it is 10.00 and 30.00.
c. In main() it is Ans[2] and in AverageHi() it is x[2].
d. Index 0 as depicted by Ans[0].
e. Index 1 as depicted by Ans[1].
f. Yes.
g. When we pass an array to a function, a reference or pointer to the array was passed. When we change or manipulate the array in the function, it effects the original array. Then, when main() refer to the original array, those values are the current values.

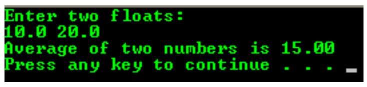
6. As a conclusion, if more than one answer has to be returned from a function, then those answers can be placed in an array that was passed from the calling function. One function can call a second function, which can call a third function. Enter 10.0 and 20.0 for the following example.

```
#include <stdio.h>

// prototypes
float AverageTwo();
void ReadTwo(float [ ]);

void main(void)
{
   printf("Average of two numbers is %.2f\n", AverageTwo());
}

float AverageTwo()
```

```
{
   float x[2];
   ReadTwo(x);
   return (float)((x[0] + x[1]) / 2.0);
}

void ReadTwo(float A[ ])
{
   printf("Enter two floats: \n");
   // for older compiler you can use scanf("%f %f", &A[0], &A[1])
   scanf_s("%f %f", &A[0], &A[1], sizeof(A), sizeof(A));
}
```

a. main() calls which function inside its printf()?
b. Does AverageTwo() call ReadTwo() or does ReadTwo() call AverageTwo()?
c. There are three functions defined here. Which functions receive arguments?
d. Which function (or functions) returns values?
e. Which function (or functions) places values in an array?
f. Number the sequence of the following events.

_______ - ReadTwo() scans two numbers in an array.
_______ - AverageTwo() returns the average of floats stored in its array back to main()
_______ - AverageTwo() calls ReadTwo() passing an array (actually the address of the array)
_______ - main() prints the average.
_______ - main() calls AverageTwo(), passing no arguments.
_______ - ReadTwo() receives the address of an array into A from the calling function.



```
Enter two floats:
10.0 20.0
Average of two numbers is 15.00
Press any key to continue . . .
```

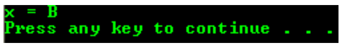
a. AverageTwo().
b. AverageTwo() calls ReadTwo().
c. ReadTwo() receives a float as an argument.
d. AverageTwo() returns value of type float.
e. ReadTwo() places values in array A[ ].
f. The sequences:

__4___- ReadTwo() scans two numbers in an array.
__5___- AverageTwo() returns the average of floats stored in its array back to main()
__2___- AverageTwo() calls ReadTwo() passing an array (actually the address of the array)
__6___- main() prints the average.
__1___- main() calls AverageTwo(), passing no arguments.
__3___- ReadTwo() receives the address of an array into A from the calling function.

More Questions on Functions

1. Show the output for the following program. What do you know about the return statement in the Socks() function body?

```
#include <stdio.h>

char Socks(char [ ]);

void main(void)
{
     char x;
     x = Socks("Black");
     printf("x = %c\n", x);
}

char Socks(char knee[ ])
{
     return (knee[0]);
}
```

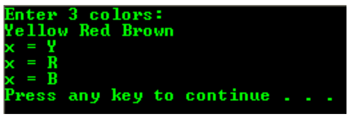


```
x = B
Press any key to continue . . .
```

The return statement return the first element of the array, denoted by the knee[0], that is a B character from Black string.

2. You are reading in “Yellow”, “Red” and “Brown”. Why the output is like that? Why not the whole words displayed?

```
#include <stdio.h>

char Socks(char [ ]);

void main(void)
{
     int i;
     char x, str[20];
     printf("Enter 3 colors: \n");
     for(i = 1; i <= 3; ++i)
     {
          // for older compiler you may use scanf("%s", &str)
          scanf_s("%s", &str, 20);
          x = Socks(str);          // Line 1
          printf("x = %c\n", x);  // Line 2
     }
}

char Socks(char knee[ ])
{
     return (knee[0]);
}
```



```
Enter 3 colors:
Yellow Red Brown
x = Y
x = R
x = B
Press any key to continue . . .
```

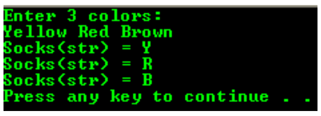
Similar to the previous example, the return statement just return denoted by the knee[0], the first element of the string though the whole string has been passed to the Socks() function through a pointer.

3. For question 2, show the output if Line 1 and 2 are combined as shown below and you can delete the variable x.

```
#include <stdio.h>

char Socks(char [ ]);
void main(void)
{
   int i;
   char str[20];
   printf("Enter 3 colors: \n");
   for(i = 1; i <= 3; ++i)
   {
      // for older compiler you may use scanf("%s", &str)
      scanf_s("%s", &str, 20);
      printf("Socks(str) = %c\n", Socks(str));
   }
```

```
    printf("Socks(str) = %c\n", Socks(str));
```

```
}
char Socks(char knee[ ])
{
   return (knee[0]);
}
```

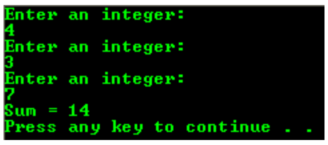


```
Enter 3 colors:
Yellow Red Brown
Socks(str) = Y
Socks(str) = R
Socks(str) = B
Press any key to continue . .
```

In this example the Socks() has been called from printf() function.

4. You are reading in 4, 3 and 7 line by line. What is this program tries to show?

```
#include <stdio.h>

int MoreSocks(void);

void main(void)
{
     int Sum = 0, i;
     for(i = 1; i <= 3; ++i)
          // notice here, summing the function
          Sum = Sum + MoreSocks();
     printf("Sum = %d\n", Sum);
}

int MoreSocks(void)
{
     int x;
     printf("Enter an integer: \n");
     scanf_s("%d", &x, 1);
     return (x);
}
```

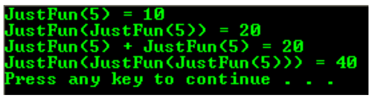


```
Enter an integer:
4
Enter an integer:
3
Enter an integer:
7
Sum = 14
Press any key to continue . .
```

The return value from the MoreSocks() has been summed up.

5. Just show the output. What this program tries to show to us?

```
#include <stdio.h>

int JustFun(int x);

void main(void)
{
     printf("JustFun(5) = %d\n", JustFun(5));
     printf("JustFun(JustFun(5)) = %d\n", JustFun(JustFun(5)));
     printf("JustFun(5) + JustFun(5) = %d\n", JustFun(5) + JustFun(5));
     printf("JustFun(JustFun(JustFun(5))) = %d\n", JustFun(JustFun(JustFun(5))));
}

int JustFun(int x)
{
     return (x + x);
}
```

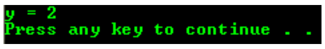


```
JustFun(5) = 10
JustFun(JustFun(5)) = 20
JustFun(5) + JustFun(5) = 20
JustFun(JustFun(JustFun(5))) = 40
Press any key to continue . . .
```

This program example shows a nested function, function in another function. The innermost function will be evaluated first and then toward the outermost. In this case the same function was used that show a recursive function where the same function calls itself.

6. Show the output for the following program. What this program tries to demonstrate to us?

```
#include <stdio.h>

int Remainder(int a, int b);

void main(void)
{
     int x = 3, y = 8;
     for( ; y >= x ; )
          y = Remainder(x, y);
     printf("y = %d\n", y);
}

int Remainder(int a, int b)
{
     return (b - a);
}
```



```
y = 2
Press any key to continue . .
```

The Remainder() function calculate the difference between the two given numbers when the second number is grater than the first number.

7. For the given main(), calls to function are given below. These calls will be placed where the comment is shown. By observing how the calls are made in each part, write the function prototype so that the data type match.

```
void main(void)
{
     int i = 7, a[4];
     float f = 7.5;
     char c = 'R', b[4];

     // place a function call here.
}
```

a. Testing();
b. Testing(i, f);
c. c = Testing(a, f);
d. f = Testing (b[2], c);
e. printf("Testing(b) = %.2f\n", Testing(b));

a. void Testing(void);
b. void Testing(int, float);
c. char Testing(int [ ], float);
d. float Testing(char [ ], char);

f. printf("Testing(c) = %d\n", Testing(c));

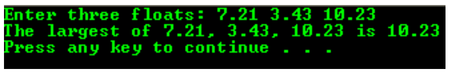
Write Programs Using Functions

Write a complete program that include the main() and the functions needed to test each of the following program.

1. The function Largest() is to receive three floats and return the largest of them.

e. void Testing(char [ ]);
f. void Testing(char);

```
#include <stdio.h>

float Largest(float, float, float);

void main(void)
{
    float p, q, r, s;

    printf("Enter three floats: ");
    // scanf("%d%d%d", &p, &q, &r);
    scanf_s("%f%f%f", &p, &q, &r, 1,1,1);
    s = Largest(p, q, r);
    printf("The largest of %.2f, %.2f, %.2f is %.2f\n", p, q, r, s);
}

float Largest(float x, float y, float z)
{
    float Largest = 0;

    if((x > y) & (x > z))
        Largest = x;
    if((y > z) & (y > x))
        Largest = y;
    if((z > y) & (z > x))
        Largest = z;
    return Largest;
}
```



```
Enter three floats: 7.21 3.43 10.23
The largest of 7.21, 3.43, 10.23 is 10.23
Press any key to continue . . .
```