Programming for Engineers I
Lab 10
Functions

Hina Ashraf, Attique Dawood

November 19, 2012

1 Functions

The functions are like subtasks. They receive some information, do some process and provide a result. Functions are invoked through a calling program. Calling program does not need to know what the function is doing and how it is performing its task. There is a specific function-calling methodology. The calling program calls a function by giving it some information and receives the result.

We have a main ( ) in every C program. main ( ) is also a function. When we write a function, it must start with a name, parentheses, and surrounding braces just like with main ( ). Functions are very important in code reusing.

There are two categories of functions:

1 Functions that return a value

2 Functions that do not return a value

Suppose, we have a function that calculates the square of an integer such that function will return the square of the integer. Similarly we may have a function which displays some information on the screen so this function is not supposed to return any value to the calling program.

1.1 Structure of a Function

The declaration syntax of a function is as follows:

return-value-type function-name(argument-list)
{
declarations and statements
}

```
//This function calculates the square of a number and returns it.
#include <stdio.h>
int square(int number)
{
    int result = 0;
    result = number * number;
    return result;
}

void main()
{
    int number, result;
    result = 0;
```

```
    number = 0;
    // Getting the  input from the user
    printf("Please enter the number to calculate the square");
    scanf("%d", &number);

    // Calling the function square(int number)
    result = square(number);
    printf("The square of %d is %d.\n", number, result);
}
```

1.2 Advantages of using Functions:

- Code Reusability
- Modularity
- Readability

2 Exercise

Question No. 1: Write a function that calculates the area of a rectangle. Function takes two integers (length and width) as input and returns an integer that is the area of rectangle.
Question No. 2: Write a program that takes an integer as input from the user. Pass that integer as an argument to a function named as evenodd (int) that prints number is even or number is odd after evaluating the number.
Question No. 3: Write a function that calculates the power of an integer number. For example you want to calculate $(2)^5$. Your function takes 2 and 5 as arguments and return the value of $(2)^5$.