Tampere University of Technology
Institute of Software Systems
Programming I

Exercise # 4 - Solution

(17-10-2001)

Exercise 1

function prototype
A function declaration without the body of the function

e.g. void Test(int, int, int);

function definition
A function declaration that includes the body of the function

```
e.g.    void Test (int d, int e, int f)
        {
                cout << (d + e)*f << endl;
        }
```

function heading
A function return type followed by a function name followed by a list of parameters

e.g. void Test (int d, int e, int f)

formal parameters
Variables declared in a function headings

e.g. d, e, and f in void Test (int d, int e, int f)

actual parameters
Variables or expressions listed in a call to a function.

e.g. a, c, and b in Test(a, c, b);

function call
A statement that transfers control to a function

e.g. Test(a, c, b);

function declaration
Same as function prototype

function body
A sequence of statements within a compound statement that shows the function implementation

```
e.g.    {
                cout << (d + e)*f << endl;
        }
```

Exercise 2

The output is:
5 3 13
3 3 9
9 12 30

Exercise 3

```
#include <iostream.h>

/* Function to return the number of hours */
int calc_hours(int time)
{
      return time/3600;
}

/* Function to return the number of minutes */
int calc_minutes(int time)
{
      return (time%3600)/60;
}

/* Function to return the number of seconds */
int calc_seconds(int time)
{
      return time%60;
}

/* Main fuction */
int main()
{
      int time;
      /* Prompting the user for the value of the time amount */
      cout << "Enter the amount of time in seconds" << endl;

      /* Reading in the time amount */
      cin >> time;

      /* Displaying number of hours, minutes and seconds */
      cout << "Number of hour(s) is ";
      cout << calc_hours(time) << endl;
      cout << "Number of minute(s) is ";
      cout << calc_minutes(time) << endl;
      cout << "Number of second(s) is ";
      cout << calc_seconds(time) << endl;

      return 0;
}
```

Exercise 4

1. Functions can have no return value but should always be passed a number of parameters. **False**
2. Functions cannot call previously undeclared functions. **True**
3. A function is allowed to have only one return statement. **False**
4. The actual parameters are evaluated before being passed to the function. **True**