Tutorial - Week 2

Objectives: To practice with
- user defined functions

1. Considering the general structure of a C program. Indicate an appropriate place for:
-function prototypes
-function definitions
-function calls

```
#include <stdio.h>

#define PI 3.141

    Function prototypes

int main(void)
{

    Function calls

    return 0;
}

    Function definitions
```

2. Find and correct errors in the following function definitions

```
a. void add2Numbers( int num1, int  num2)
   {
      int num1; int sum;

      sum = num1 + num2;
      return sum;
   }


b. int sum2Num( int num1, int num2  );
   {
     int sum;

     sum = num1 * num2;
     return      ;
   }
```

a. 1) Wrong re-declaration of the formal parameter `num1; a variable can be declared only ONCE within its own scope and C supports five variable scopes, see slide 16 of lecture 2`
   2) The return type of the function should be `int.`

b. 1) The semicolon at the end of the function header should be omitted.
   2) The function does not return an integer result as its return type indicates.
   3) The function name does not match its purpose (not an error but not a good practice).

`See further remarks at the end of this document.`

3. Find errors in the following function prototypes:

```
a. int sum (int x, y, z);
b. int sum (int, int, int);
c. int sum (void, int y, int z);
```

a. Error, data types for variables `y` and `z` are missing
b. No error
c. Error, void cannot be a formal parameter among more; it is used, it should be the only function parameter.

4. Considering the following function prototype:

```
int findMax(int num1, int num2,int num3);
```

Write a function call to find the maximum value among three numbers `inputA, inputB` and `1000.`

```
int inputA, inputB, maxValue;
.............

maxValue = findMax (inputA, inputB, 1000);
```

5. Sound level is commonly measured in decibels (dB)

$$Sound\ level = 20*\log_{10}(sp/rp)$$

where:
sp **is a measured sound pressure**
rp **is a reference pressure (the sensitivity of human ear $20*10^{-6}$ Pa)**

Provide a function prototype and its definition that calculates the sound level based on the sound pressure measured. Provide an example of the function call.

```
#define          REF_PRES 20.0e-6

float getDbLevel ( float sountPressure  );
```

```
..  . .

soundLevel = getDbLevel( measuredSdPress  );


.   . .

float getDbLevel ( float sountPressure  )
{

     float soundLev;

     soundLev = 20.0*log10( soundPressure / REF_PRES );

     return soundLev;
}
```

Math.h library should be used

6. Write statements that compute and display the absolute difference of two type double variables, *x* and *y* (| *x* – *y* |).

```
#include <stdio.h>
#include <math.h>

double tmp, x, y;

tmp = fabs(x - y);

printf("%.2f", tmp) ;

            or

#include <math.h>

double tmp, x, y;

printf("%.2f", fabs(x - y));
```

7. Write a complete C program that prompts the user for the coordinates of two 3-D points (x_1, y_1, z_1) and (x_2, y_2, z_2) and displays the distance between them computed using the following formula:

$$distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

```
/*
 * Compute the distance between two 3-dimensional points.
 */

#include <stdio.h>
```

```
#include <math.h>

int
main(void)
{
   double x1, y1, z1, x2, y2, z2, distance;

   /* Get the coordinates. */
   printf("Enter the first set of x y z coordinates> ");
   scanf("%lf%lf%lf", &x1, &y1, &z1);
   printf("Enter the second set of x y z coordinates> ");
   scanf("%lf%lf%lf", &x2, &y2, &z2);

   /* Compute the distance. */
   distance = sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2) + pow(z1 -
   z2, 2));

   /* Display the result. */
   printf("The distance between (%.2f,%.2f,%.2f) and (%.2f,%.2f,
   %.2f) is %.2f.\n", x1, y1, z1, x2, y2, z2, distance);

   return (0);
}
```

8. If an actual argument of −35.7 is passed to a type `int` formal parameter, what will happen? If an actual argument of 17 is passed to a type `double` formal parameter, what will happen?

The formal parameter's value will be −35. The formal parameter's value will be 17.0.

9. Write the prototype for a function called `script` that has three input parameters. The first parameter will be the number of spaces to display at the beginning of a line. The second parameter will be the character to display after the spaces, and the third parameter will be the number of times to display the second parameter on the same line.

```
int script(int num_spaces, char char_to_display, int num_of_chars);
```

10. Write a function that computes the time one must leave in order to reach a certain destination by a designated time. You need to deal only with arrivals occurring later in the same day as the departure. Function inputs include the arrival time as an integer on a 24-hour clock (8:30 P.M. = 2030), the distance to the destination in kilometers, and the speed you plan to average in km/h. The function result should be the required departure time (rounded to the nearest minute) as an integer on a 24-hour clock. Also, write a driver program to test your function.

```
/*
```

```
 * Computes the departure time required to reach a destination that
 * is a given (positive) distance away, based on supplied arrival
 * time and estimated average speed.  Arrival must be on same day
  * as departure.
 */
#include <stdio.h>

/* function prototype */
int find_departure_time (int arr_time, float distance, float speed);

int  main(void)
{
   int arr_time;      /* input--arrival time */
   float distance,   /* input--distance traveled (km) */
         avg_speed;   /* input--anticipated average speed (km/hr) */
    int dep_time;     /* output--required departure time */

   /* Get arrival time */
   printf("Enter arrival time as integer on a 24 hour clock.  For
           example,");
   printf("\n8:30 PM would be entered as 2030\n");
   printf("Arrival time> ");
   scanf("%d", &arr_time);

   /* Now get the distance to travel and anticipated average speed*/
   printf("Enter the distance in km> ");
   scanf("%f", &distance);

  printf("Enter anticipated average speed (including stops) in
           km/hr> ");
   scanf("%f", &avg_speed);

   /* Compute and display the required departure time. */
   dep_time = find_departure_time(arr_time, distance, avg_speed);
   printf("You need to leave at %d.\n", dep_time );

   return (0);
```

```
}

/*
 * Returns the departure time required to travel distance
 *    given arrival time (arr_time) and speed.
 * Pre:  dist/speed <= arr_time converted to hours since midnight
  *    (i.e., arrival is same day as departure)
 */

int
find_departure_time (int arr_time, float distance, float speed)
{
   float journey;       /* travel time in hours */
   int journey_min,       /* travel time in minutes (rounded) */
   arr_min,           /* total minutes to arrival time */
   dep_time_min,       /* departure time in minutes */
   dep_hr,            /* hour of departure (24-hr clock) */
   dep_min,           /* minutes of departure time */
   dep_time;          /* departure time (24-hr clock) */

   journey = 60*distance / speed;
   journey_min = (int)(journey + 0.5); /* rounding */

   arr_min = arr_time / 100 * 60 + arr_time % 100;

   dep_time_min = arr_min - journey_min;

   dep_hr = dep_time_min / 60;
   dep_min = dep_time_min % 60;
   dep_time = dep_hr * 100 + dep_min;

     return (dep_time);
  }
```