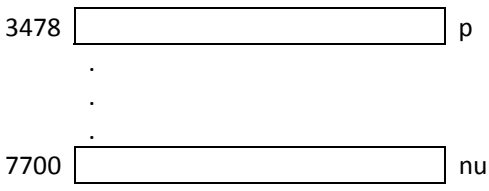
Tutorial 2 – Functions and Pointers

1. Assume the following declaration:

```
int number;
int *p;
```

Assume also that the address of number is 7700 and the address of p is 3478. That is,



For each case below, determine the value of

(a) number (b) &number (c) p (d) &p (e) *p

All of the results are cumulative.

(i) p = 100; number = 8
(ii) number = p
(iii) p = &number
(iv) *p = 10
(v) number = &p
(vi) p = &p

i) a) 8 b) 7700 c) 100 d) 3478 e) 100
ii) a) 100 b) 7700 c) 100 d) 3478 e) 100
iii) a) ~~7700~~ 100 b) 7700 c) 7700 d) 3478 e) 100
iv) a) ~~7700~~ 10 b) 7700 c) 7700 d) 3478 e) 10
v) a) 3478 b) 7700 c) 7700 d) 3478 e) ~~10~~ 3478
vi) a) 3478 b) 7700 c) 3478 d) 3478 e) ~~10~~ 3478

2. Find the error in each of the following program segments and explain how the error may be corrected.

(a)
```
int product(int m, int n)
{
int result;

        result =m * n;
}
```
no return
return result

(b)
```
int sumofSquare(int n)  /* assume n is non-negative */
{
  int j;
  int sum = 0;

        if (n == 0)
                return 0;
        else
           for (j = 1; j <= n; j++) sum += j * j ;
}
```
next line

(c)
```
void  ft(float a)
{
```

```
    float a;

            printf("%f\n", a);
    }

(d) void  height(float * h)
    {
            scanf("%f", &h);
    }

(e) void  height(float * h)
    {
            scanf("%f", h);
            return *h;
    }

(f) int divideBy4(int n)
    {  a = divideby2(n)
       int divideBy2(int m)
       {
              return m/2;
        }

       return (divideBy2(divideBy2(n));
  }
```

definition of divideby2() taken out of function divideby4()

3. What will be the output of the following program?

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main()
{
  int h, k;

     h = 5;
     k = 15;
     printf("h = %d, k = %d\n", h, k);  /* line (i) */
     function0();
     printf("h = %d, k = %d\n", h, k);  /* line (ii) */
     function1(h, k);
     printf("h = %d, k = %d\n", h, k);  /* line (iii) */
     function2(&h, &k);
     printf("h = %d, k = %d\n", h, k);  /* line (iv) */
     return 0;
}
void function0()
{
  int h, k;
```

```
    h = k = -100;
    printf("h = %d, k = %d\n", h, k);  /* line (v) */
}
void function1(int h, int k)
{
    printf("h = %d, k = %d\n", h, k);  /* line (vi) */
    h = k = 100;
    printf("h = %d, k = %d\n", h, k);  /* line (vii) */
}
void function2(int *h, int *k)
{
    printf("h = %d, k = %d\n", *h, *k);  /* line (viii) */
    *h = *k = 200;
    printf("h = %d, k = %d\n", *h, *k);  /* line (ix) */
}
```

4. **(calDistance)** Write a C program that accepts four decimal values representing the coordinates of two points, i.e. (x1, y1) and (x2, y2), on a plane, and calculates and displays the distance between the points:

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

Your program should be implemented using functions. Provide two versions of the function for calculating the distance: (a) one uses call by value only for passing parameters; and (b) the other uses call by reference to pass the result to the calling function.

```
//call by value

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

int square(int);
int main()
{
   float x1,x2,y1,y2,diff1,diff2,num1,num2,sum;
   printf("Enter the 1st x coord:\n");
   scanf("%f", &x1);
   printf("Enter the 1st y coord:\n");
   scanf("%f", &y1);
   printf("Enter the 2nd x coord:\n");
   scanf("%f", &x2);
   printf("Enter the 2nd y coord:\n");
   scanf("%f", &y2);

   diff1=x2-x1;
   num1 = square(diff1);
   diff2=y2-y1;
   num2 = square(diff2);
   sum = num1+num2;
   printf("The distance is:%f",sqrt(sum));
   return 0;
}
int square(int value)
{
   return value*value;
}
```

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

void distance(float *a,float*b,float *c);
int main()
{
   float x1,x2,y1,y2,diff1,diff2,c = 0;
   printf("Enter the 1st x coord:\n");
   scanf("%f", &x1);
   printf("Enter the 1st y coord:\n");
   scanf("%f", &y1);
   printf("Enter the 2nd x coord:\n");
   scanf("%f", &x2);
   printf("Enter the 2nd y coord:\n");
   scanf("%f", &y2);

   diff1 = x2-x1;
   diff2 = y2-y1;
   distance(&diff1, &diff2, &c);
   return 0;
}
void distance(float *a, float*b, float *c)
{
   *a *= *a;
   *b *= *b;
   *c = *a + *b;
   *c = sqrt(*c);
   printf("The distance is:%f",*c);
}
```