Tutorial 1 – Basic C Programming and Control Flow

1. State the data type of each of the following:

a.	**'1'**	g.	**1870943465324L**
b.	**23**	h.	**1.234F**
c.	**0.0**	i.	**-564**
d.	**'\040'**	j.	**0177**
e.	**0x92**	k.	**0XfF4**
f.	**'\a'**	l.	**0xaaBB76L**

2. (a) What will the following program output? (refer to an ASCII table)
   (b) What will happen if the format specifier of the second printf is changed to **%d**?
   (c) What will be the result if **0x** in the third printf is removed?
   (d) What if the first **0** in the fourth printf is deleted?

```
#include <stdio.h>

int main()
{
    printf("%c", 'A');
    printf("%c", 65);
    printf("%c", 0x41);
    printf("%c", 0101);
    return 0;
}
```

3. Assume x and y are integer variables. What will happen if one of the following statements is executed?

   (a) **scanf("%d %d", &x, &y);**
   (b) **scanf("%d %d", x, y);**
   (c) **scanf("%d/%d", &x, &y);**

4. The output of the following code is not zero. Why?

```
{   …….
    double A = 373737.0;
    double B;

    B = A * A * A + 0.37/A - A * A * A - 0.37/A;
    printf(“ The value of B is %f.\n”, B);
}
```

5. Given the following declarations and initial assignments:

```
int      i,  j,  m,  n;
```

```
float       f, g;

i = j = 2;
m = n = 5;
f = 1.2;
g = 3.4;
```

evaluate the following expressions independently, i.e. all variables start with the same set of initial values. Show any conversions which take place and the type of result.

(a)	**m * j / j**	(b)	**m / j * j**
(c)	**(f + 10) * 20**	(d)	**(i++) * n**
(e)	**i++ * n**	(f)	**-12L * (g - f)**
(g)	**m = n = --j;**	(h)	**(int) g * 10**
(i)	**(int) (g * 10)**	(j)	**j = i + f**

6. Which of the following are acceptable case constant expressions? Assume the convention that upper case is used for defining a constant, e.g.

```
#define        SVALUE        10
```

and other identifiers are variables.

(a)	case 76:	(b)	case number*2:
(c)	case SVALUE*2:	(d)	case 80.1:

7. In some computer games it is necessary to introduce a delay to slow the computer down. Assume that you are running the following program on a computer which uses 16 bits to represent an integer. How can the delay be (a) shortened, (b) made a thousand times longer, (c) made variable after compilation?

```
#include  <stdio.h>
#define  DLENGTH 32000

int main()
{
    int count;
    ......
    for (count = -DLENGTH; count <= DLENGTH; count++)
            ;   /* this is a NULL statement which does nothing */
    ......
}
```

8. Are the following code segments the same?
(a) if (x != 0 && 2/x != 1) { …..}
(b) if (2/x != 1 && x != 0) { …..}

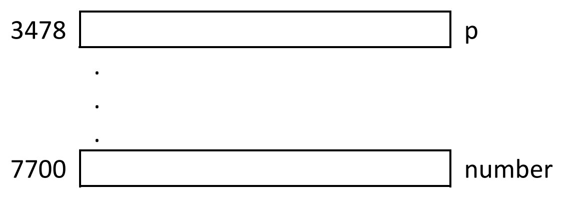
9. Write a section of C program to interchange the values of two integer variables. Is there a way of solving this problem without using a third variable?

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.

	(a) number	(b) &number	(c) p	(d) &p	(e) *p
(i) p = 100; number = 8	8	7700	100	3478	
(ii) number = p	100	7700	100	3478	
(iii) p = &number	100	7700	7700	3478	
(iv) *p = 10	10	7700	7700	3478	10
(v) number = &p	3478	7700	7700	3478	3478
(vi) p = &p	3478	7700	3478	3478	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;
}       return result;
```

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

        if (n == 0)
              return 0;        j  is  not  defined
        else
           for (j = 1; j <= n; j++)  sum += j * j ;
}               return ?
```

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

Double declaration, it erases argument

```
      float a;

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

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

Argument is already a pointer
h is alr address

can do scanf("%f", h);

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

No Return

```
 (f) int divideBy4(int n)
      {
         int divideBy2(int m)
         {
               return m/2;
          }

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

No nested definition allowed

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 = 5, K= 15 i

h = -100, K=100 V

h = 5 , K = 15 ii

h = 5 , K = 15 Vi

h = 100 , K = 100 Vii

h = 5 , K = 15 iii

h = 5 , K = 15 Viii

h = 200, K = 200 ix

h = 200, K = 200 iV

```
    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.