WEEK 10 ASSIGNMENT SOLUTION

1. How can you improve the best-case efficiency in bubble sort? (The input is already sorted)

   a)
```
boolean swapped = false;
for(int j=arr.length-1; j>=0 && swapped; j--)
{
        swapped = true;
        for(int k=0; k<j; k++)
        {
                if(arr[k] > arr[k+1])
                {
                        int temp = arr[k];
                        arr[k] = arr[k+1];
                        arr[k+1] = temp;
                        swapped = false;
                }
        }
}
```

   b)
```
boolean swapped = true;
for(int j=arr.length-1; j>=0 && swapped; j--)
{
        swapped = false;
        for(int k=0; k<j; k++)
        {
                if(arr[k] > arr[k+1])
                {
                        int temp = arr[k];
                        arr[k] = arr[k+1];
                        arr[k+1] = temp;
                }
        }
}
```

c)
```
boolean swapped = true;
for(int j=arr.length-1; j>=0 && swapped; j--)
{
        swapped = false;
        for(int k=0; k<j; k++)
        {
                if(arr[k] > arr[k+1])
                {
                        int temp = arr[k];
                        arr[k] = arr[k+1];
                        arr[k+1] = temp;
                        swapped = true;
                }
        }
}
```

d)
```
boolean swapped = true;
for(int j=arr.length-1; j>=0 && swapped; j--)
{
        for(int k=0; k<j; k++)
        {
                if(arr[k] > arr[k+1])
                {
                        int temp = arr[k];
                        arr[k] = arr[k+1];
                        arr[k+1] = temp;
                        swapped = true;
                }
        }
}
```

Solution: (c)
A boolean variable ‘swapped’ determines whether any swapping has happened in a particular iteration, if no swapping has occurred, then the given array is sorted and no more iterations are required.

2. Bisection method is used to find

a) Derivative of a function at a given point
b) Numerical integration of a function within a range
c) Root of the function
d) None of the above

Solution: (c) Root of the function

3. In ……………, search starts at the beginning of the list and checks every element in the list.

a) Linear search
b) Binary search
c) Hash search
d) Binary tree search

Solution: (a) Linear search

4. What is the output of the following program?

```
# include <stdio.h>
void func(int x)
{
   x = 40;
}

int main()
{
  int y = 30;
  func(y);
  printf("%d", y);
  return 0;
}
```

a) 40
b) 30
c) Compilation error
d) Runtime error

Solution: (b) 30
Parameters are always passed by value in C. Therefore, in the above code, value of y is not modified using the function func().
Note that everything is passed by value in C. We only get the effect of pass by reference using pointers.

5. Assuming an initial range [1,5], the second (at the end of 2 iterations) iterative value of the root of $te^{-t} - 0.3 = 0$ using the bisection method is (Note: you need to find the root, not the function value)

Solution: 2 (short answer type)

$$t_u = 5 \; and \; f(t_u) = -0.2663$$

$$t_l = 1 \; and \; f(t_l) = 0.0679$$

$$f(t_u) \times f(t_l) = 0.0181 < 0$$

Therefore, at least one root exists between [1,5]
Iteration1: $t_m = \frac{t_u+t_l}{2} = \frac{5+1}{2} = 3$ Thus, $f(t_m) = -0.1506 < 0$
Therefore, at least one root exists between [1,3] and we make $t_u = 3$

Iteration2: $t_m = \frac{t_u+t_l}{2} = \frac{3+1}{2} = 2$ Hence, the root after the second iteration is 2.

6. What would be the equivalent pointer expression for referring to the array element a[i][j][k][l]?

a) (((*(a+i)+j)+k)+l)
b) *(*(*(*(a+i)+j)+k)+l)
c) (*(*(a+i)+j)+k+l)
d) *((a+i)+j+k+l)

Solution: (b)

7. What will be output when you will execute the following c code?

```
#include<stdio.h>
int main()
{
   short num[3][2]={2,5,11,17,23,28};
   printf("%d,%d",*(num+2)[0],**(num+1));
   return 0;
}
```

a) 23,11
b) 23,23
c) 11,17
d) 17,17

Solution: (a) 23,11

(num+2)[0]=(*((num+2)+0))=*(*(num+2))=*(num[2])=num[2][0]=23
And **(num+1)=*(num[1]+0)=num[1][0]=11
This is an example of pointer arithmetic on an array.

8. Assume sizeof an integer and a pointer is 2 byte.
What is the output?

```
#include <stdio.h>
#define A 5
#define B 8

int main()
{
  int (*x)[A][B];
  printf("%d",  sizeof(*x));
  return 0;
}
```

a) 40
b) 80
c) 120
d) 160

Solution: Output is 5*8*sizeof(int) which is “80” assuming integer size as 2 bytes.

9. Find the output of the following program

```
#include <stdio.h>
int main()
 {
  int *ptr, a = 7;
  ptr = &a;
  *ptr  =*ptr - 2;
  printf("%d,%d ", *ptr, a);
  return 0;
}
```

Solution: 5,5 (short answer type)
The pointer variable ptr contains the address of the variable a. Incrementing the value at address also modifies the content of a. Thus, both will be containing 5.

10. What is the solution of the equation given below using the Bisection Method up to four decimal places? (Consider the root lying on positive quadrant only and compute the root till five iterations only)

$$f(x) = xe^{2x} - 3x^2 - 5$$

Solution: 1.0312 (short answer type)

The root lies between 1 and 2. Using bisection method, we can find the root as 1.0312 (upto three decimal accuracy)