Week 10: Assignment 10 Solutions

1. The 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) The root of the function
   d) None of the above

Solution: (c) The root of the function

2. 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.

3. What is the advantage of a recursive approach over an iterative approach?
   a) Consumes less memory
   b) Less code and easy to implement
   c) Consumes more memory
   d) More code has to be written

Solution: (b) Less code and easy to implement

4. Which of the following is not an application of binary search?
   a) To find the lower/upper bound in an ordered sequence
   b) Union of intervals
   c) Debugging
   d) To search an unordered list

Solution : (d) In Binary search, the elements in the list should be sorted. It is applicable only for the ordered list. Hence Binary search in an unordered list is not an application.

5. If for a real continuous function $f(x)$, $f(a) \times f(b) > 0$, then in the range of [a,b] for $f(x) = 0$, there is (are)

a) Exactly one root
b) no root exists
c) at least one root
d) roots are undermined

Solution: (b). If, $f(a) \times f(b) < 0$ then they have opposite signs; only then root(s) exists. Since $f(x)$ is continuous between a and b, the function needs to cross the x-axis to have $f(x)$ solvable. Here, as $f(a) \times f(b) > 0$, so the function does not cross x-axis. Hence, no root exists.

6. 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 (Numeric 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.

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 size of an integer and a pointer is 4 bytes. What is the output?

```
#include <stdio.h>
#define A 5
#define B 8
#define C 2
int main()
{
  int (*x)[A][B][C];
  printf("%d",  sizeof(*x));
  return 0;
}
```

Solution: (Numeric answer) 320. Output is 5*8*2*sizeof(int) which is “320” assuming integer size as 4 bytes.

9. Find the output of the following program

```
#include <stdio.h>
int main()
{
        int *ptr, a = 12;
        ptr = &a;
        *ptr  = *ptr - 2**ptr;
        printf("%d,%d", *ptr, a);
        return 0;
}
```

Solution: -12,-12 (short answer type)

The pointer variable ptr contains the address of the variable a. The operation essentially computes *ptr = -*ptr. Further the value at address also gets modified. Therefore, both will be containing -12.

10. What is the output?

```
#include <stdio.h>
   int main()
   {
      char *s = "programming";
      char *p = s;
      printf("%c,%c", *(p + 3), s[3]);
      return 0;
   }
```

a) o,o
b) p,g
c) g,g
d) g,r

Solution: (c) g,g

p points to the base address of ‘programming’ i.e. p. so *(p+3)= the fourth character i.e. g. Similarly s[3] is also g. This is a simple example of pointer arithmetic on strings.