DSA assignment - 6

D. Gireeswar Varma
AP19110010557
CSE - F

1 (a)

```
# include <stdio.h>
Void binary - search ();
int a [50], n, item, loc, beg, mid, end; i,;
Void main ()
{
 Printf ("\n Enter the side of an array");
 scanF ("%d", &n);
 PrintF ("\n Enter elements of an array in sorted form"]
 For (i=0; i<n; i++ )
 scanF ("%d", &a[i]);
 PrintF ("\n Enter ITEM to be searched:");
 scanF ("%d", &item);
 binary - search ();
 getch ();
}
Void binary - search ();
{
   beg = 0
   end = n-1
   Mid = (beg + end ) /n ;
 While ( (beg < = end ) && (a [Mid] ! = item)]
 {
   if (item < a [Mid])
     end = mid - 1;
     else
       beg = mid+ 1
       mid = (beg + end /2
 }
if  (a [Mid] = - item )
 Printf ( "\n \n ITEM Found at location %d", mid+1);
else
   Printf ("\n\n ITEM doesn't exist ) ;
}
```

b)
```
# include < stdio.h >
 int main ()
 {
   int arr [10] :
   int  sum , Pro emct,0;
   Printf  ("\n enter elements : \n");
   for  ( i=0, i<10; i++ )
 {
   Printf  ( "enter arr [%d] :", i );
   Scanf   ("%d ", & arr [i] );
  {
    Sum  = 0 ;
    Product = 1;
     For  ( i=0; i<0 ; i++ )
  {
    Sum  =  Sum + arr [i];
    Product  =   Product arr [i] ;
   }
      PrintF  ("\n Sum of  array is  : %d", sum );
      PrintF  ("\n Product of array is : %d/Product");
    return 0 ;

   }
```

2)
```
# include < stdio.h >
# include < stdio.h >
// Merges two sub arrays of arr []
// First sub array is arr [1...m]
// second sub array is arr [net 1...r]
Void merge (int arr [], int l, int m, int r )
{
    int i, j K ;
    int n1 = m - l + 1 ;
    int n2 = r - m ;
    int L [n1] , R [n2] ;
    For ( i = 0 ; i < n1 ; i++ )
      L(i ) = arr [ l + i ]
    For ( j > 0 ; i < n2 ; j++ )
        R [i] = arr [m + 1 + ] ;
     i = 0 ;   (initial index of 1st sub array )
     j = 0 ;   ( initial index of end sub array )
     K = 1 ;   ( initial index of merge sub array )
     While ( i < n1 44 j < n2 )
     {
         i if ( 2 (i) < = R [j] )
       {
           arr [e] = 2 [i] ;
           i++ ;
         }
         else
         }
           arr [K] = R [j] ;
            j++ ;
         }
```

```
While   ( j < n2 )
 {
    arr [k] = R [j] ;
     j ++ ;
     k++ ;
    }
   Void merge sort (int arr [r], int l, int r )
   {
      i  ( i < r )
      {
        int  m = l + (r - l ) /2 ) ;
        merge  sort  (arr , l ,m ) ;
        merge  Sort   (arr , m +1, r ) ;
         merge    (arr, l , m, r ) ;
       }
     }
      void  Print array  ( int  + 1 ) , int  si3e )
     {
      int  i ;
       dor  ( i = 0 ;  i < Size ;  i++  )
       PrintF    ( "%d ; R [i] ) ;
       PrintF   ( " \n" ) ;
      }
        int  main ( )
       {
        int  arr [ ] =  { 12, 11, 13, 5, 6, 7 } ;
        int     arr - Size =  Size oF  (arr) | size of arr[0];
       Print F  ( " \n Sorted array is \u" ) ;
       Print F   array  (arr, arr - Size ) ;
       return 0 ;
       }
```

3)

selection Sort :-

```
# include <stdio.h>
Void swaP (int *a, int* b)
{
 int temP = *a:
  *a = *b
  *b - temP
 }
  Void selection sort (int array [], int size)
  {
  For (int steP = 0 ; steP < size-1 ; steP ++ )
  {
   int (min - idx = steP ;
   For  (int i= steP+1 ; i<size ; i++ )
   if  (array [i] <array [min - idx ] )
    Min - idx = i ;
   }
   SwaP ( array [min - idx] array [steP]);
   }
   }
    Void Print array (int array [] , int size
    For  (in t  i=0 i < size ; t++ 1) {
     PrintF ( " 1.d ", array [i] );
    }
     PrintF ( " In " ) ;
    }
    }
```

```
int  main()
}
 int  data [] - { 20 , 12 , 10 , 15 , 2 } ;
  int  size  =  size of  (data ) )size (data [0]);
   Selections  (data , size )
   Print f  (" Sorted  array in asending order \n");
   PrintF array  (data , size );
  }

4 (i)     # include  <stdio.h>
         # include  <math.h>

           intmain ()
          {
           int  a[] =  { 16, 19, 11, 15, 10, 12, 14 } ;
           int  i, j ;
           For  ( j=0 , j<7 ; j++ )
           {
              int  swapped = 0 ;
             i=0 ;
             while  ( i<7-1 )
             {
              if  ( a[i] >a [i+1] )
              {
               int  temp = a [i] ;
                 a[i] = a [i+1] ;
                  a [i+1] = temP ;
                  SwaPPed = 1 ;
                 }
```

```
        i++;
        {
         iA   (2 swapped)
          break;
        }
          For  (i=0; i<7; i+1)
          Printf  ("%d In", a [i]);
           return 0;
         }
ii)     # include <stdio.h>
        # include <conio.h>
        {
         int  num, evenSum=0, odd Prod=1, ren, temP;
         Printf  ("enter any numbers;");
         scanf  ("%d", &num);
          While   (num>0 )
          {
           rem = num % 10;
            if  (rem % 2 = =0)
            even sum = evenSum+rem;
          else
              odd  Prol = odd Prod * rem;
           num = num /10;
          }
          Printf  ("In sum of even digit= %d",even num);
          Printf  ("In Product of odd digit = %d", oddProd);
          getch s;
          return0;
         }
```

iii)

```
# incude <stdio.h>
Void SWAP (int * xP, int * yP )
{
   int temP - *xP ;
   * xP = *yP ;
   * uP = *temP ;
   }
    int i, j ;
    For (i=0; i<u-1; i++)
    For  (j=0; j<u-i-1; i++)
     if  (arr [j] > arr [j+1])
      swaP  & arr [j] & arr [j+1] ;
     }
      Void  Print Array (int arr [] ,int Size)
     {
       int i ;
       For  (i=0 ; i< Size ; i++)
        PrintF  ( " %.d ", arr [i] );
        Print F  ("\n") ;
       }
        int main ( )
       {
         int arr [] = {64 , 84 , 25, 12, 22, 11, 90};
     int  n= Size OF F (arr) | Size OF (arr [0]);
     bubble  sort  (arr , n );
      PrintF  sort (arr, n ) ;
       PrintF  ( " sorted array : \n" ) ;
       PrintF Array  (arr . n ) ;
       return 0;
      }
```

5)

```
# include < stdio.h>
  void binary-search (int[],int ,int,int );
  void bubble-sort  (int [],int );
  int main ()
  {
    int key ,size,i;
    int list [25] ;
    Printf ("enter size of a list");
     scanf  ("%d", & size );
     Printf  ("enter elements \n");
      for  (i=0, i< size, i++ )
     {
       scanf  (" %d ", & list [i])
     }
       bubble - sort  (list , size );
       Printf  ("\n");
      Printf  ("enter key to search in");
       Scanf  ("%d",key );
       binary - search  (list ,0, size key );
     }
        void  bubble -search ( list ,size, key );
     {
           int  temp , i, j;
          for  (i=0,  i< size ;i++ )
       {
            for (list [i] > list [i] )
         {
            temp = list [i] ; ag
        list [i] - list [j]
        list [j] - temP ;
      }
    }
  }
}
 Void  binary .search list [] ,int ,lo,int hi,int Key)
   {
     int  main ;
     if  (lo > hi )
     {
       PrintF ( " Key not Found \n" );
       return ;
      {
        mid - (lo + hi) /2;
       if    (list [mid] = = Key )
       {
          PrintF  (" Key Found \n");
        }
          else if (list [mid] >Key ]
         {
           binary -Search (list , lo ,mid-1, Key );
         }
           else if  (list [mid ] <Key )
          {
            binary  -search  (list ,mid+1 hi, Key);
          }
        }
```