C assignment (DSA)

AP19110010111
Chandu.ch
CSE-G

1.

(a)
```
# include <stdio.h>
    void binary_search();

    int a[50], n, item, loc, beg, mid, end, i;
    void main()
    {
      printf ("\n Enter the size of an array");
      scanf ("%d", &n);
      printf ("\n Enter elements of an array in sorted form:\n");
      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) /2;
         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, product, i;

    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",\nproduct");

    return 0;
}
```

Q. ~~# include <stdio.h>~~

```
# include <stdio.h>

// Merges two subarrays of arr[]
// First subarray is arr[l... m]
// second subarray is arr[m+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; j<n2; j++)
       R[j]= arr [m+1+j];
    i = 0;      (initial index of 1st subarray)
    j = 0 ;     ( initial index of 2nd subarray)
    k = l;      ( initial index of merge subarray)

  while (i<n1 && j<n2)
  {
     if (L[i] <= R[j])
     {
        arr[k] = L[i];
         i++;
      }
      else
      {
       arr [K] = R[j];
        j++;
      }
       k++;
   {
```

```
while (j<n2)
 {
   arr[k] = R[j];
   j++;
   k++;
 }
void mergesort (int arr[ ] , int l, int r)
 {
   if (l<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 A[], int size)
  {
    int i;
    for (i=0; i<size; i++)
      printf ("%d", A[i]);
     printf ("\n");
  }
  int main()
  {
    int arr[] = {12, 11, 13, 5, 6, 7};
    int arr_size = size of(arr) / sizeof(arr[0]);
   printf (" Given array is \n");
   print Array (arr, 0, arr_size -1);
  Print ("\n sorted array is \n");
  print 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  printarray (int array[ ], int size)
      {
       for (int  i = 0; i < size ; ++i ) {
         printf(" %d ", array [i ]);

         }
          printf (" \n") ;

         }
         }.
```

```
int  main()
 {
   int  data[] = {20, 12, 10, 15, 2};
   int size = size of (data) / size (data[0]);
   Selectionsort (data, size)
   printf (" sorted  array  in ascending order: \n");

   printarray (data, size);
   }
```

3.
```
# include <math.h>.
# include <stdio.h>

void insertion sort (int arr[], int n)
{
  int i, key, j;
   for (i=1; i<n; i++)
   {
    key = arr[i];
    j = i-1
    while (j>=0 && arr[i] > key)
    {
     arr [j+1] = arr [i];
     j = j-1
    }
     arr [j+1] = key;
   }
  }
  void printArray (int arr[], int n)
  {
   int i;
    for (i=0; i<n; i++)
        printf ("%d ", arr [i]);
    printf ("\n");
   }
   int main()
   {
     int arr[] = {12, 11, 13, 5, 6};
     int n = sizeoff (arr) / sizeof arr[0]);
     insertionsort (arr, n);
     printArray (arr, n);
     return 0;
   }
```

Scanned with CamScanner

4.(ii) # include <stdio.h>

```
# include <math.h>

int main()
{
  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++;
    }
     if (!swapped)
      break;
    }
     for (i=0; i<7 ;i++)
       printf ("%d\n", a[i]);

       return0 ;

    }
```

```
nid  bubble- sort (int

4(ii) # include <stdio.h>
      # include <conio.h>
         {
          int  num, evensum = 0, odd prod = 1, rem ,temp;
            printf (" Enter any number: ");
             scanf ("%d", & num);
             while (num>0)
              {
                rem = num % 10;
                if (rem% 2 == 0)
                  evensum = evensum+rem;
               else
                  odd prod = oddprod * rem;
               num = num/10;
             }
              printf ("\n sum of Even digit = %d ",evensum);
              printf("\n product of odd digit = %d", odd prod);
              getch();
              return 0;
            }
```

4 (iii) #include <stdio.h>

```
void swap ( int * xp, int * yp )
{
    int temp = * xp;
    * xp = * yp;
    * yp = temp;
}

        int i,j;

        for (i=0; i<n-1; i++)

          for (j=0; j<n-i-1; j++)

            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]);
        printf ("\n");
      }
        int main()
        {
           int arr [] = {64, 34, 25, 12, 22, 11, 90};

        int n = sizeof (arr) / sizeof (arr[0]);
        bubble sort (arr, n);
        printf (" sorted array :\n");
        Print 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 \n");
          scanf ("%d ", key);
          binary- search (list, 0, size key );

    }
     void bubble_ search (list, 0, size, key );
     {
        int temp, i, j;
      for (i=0; i<size; i++)
       {
         for (j=i; j<size; j++)
          {
            if (list[i] > list [j])
             {
              temp = list [i] ; aq
        list [i] = list[j]
          list [j] = temp;
        }
      }
    }
  }
   Void  binary-search (int list[], int lo, int hi, int key)
   { int maid ;
      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) ;
       }
        elseif ( list [mid ] < key )
        {
          binary_search (list, mid +1, hi , key);
        }
     }
```