C assignment (DSA)

Nitin Varma Rudraraju
AP19110010336
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 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 );
    else
        printf("\n\n ITEM doesn't exit");
    }
```

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", \n
        product);
return 0;
}
```

2.
```
#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; i<n2; j++)
```

```
R[i] = arr[m+1+j];
i=0;  (initial index of 1st subarray)
j=0;  (initial index of end 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 merge sort (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 array is \n");
   print f(" 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, 1915, 2};
    int size = sizeof(data)/sizeof(data[0]);
    selectionsort(data, size)
    printf("sorted array in ascending order
                                    :\n");
    print array(data, size);
}
```

③.

```
#include<stdio.h>
#include<math.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 = size of (arr) / size of arr[0]);
insertionsort (arr, n);
  printarray (arr, n);
  return 0;
}
```

4 (i)

```
# Include <stdio.h>
# Include <math.h>
int main ()
{
int a[ ] = {16,9,11,15,10,12,14};
int i,j;
for (j=0; j<7; j++)
{
int supposal = 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])
return 0;
}
```

4(ii)

```
#include <stdio.h>
#include <conio.h>
{
int num,even sum =0, odd prod=1,
rem,temp;
printf("Enter any number: ");
scanf("%d",&num);
while (num>0
{
sum = num%10;
if (rem%2==0)
even sum = even sum + rem;
else
odd prod = odd prod * 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; j++)
if (arr[j] > arr[j+1])
swap (&arr[j], &arr[j+1]);
}
void printArray (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 = size of (arr)/size of (arr[0]);
bubble sort (arr,n);
print f("sorted array : \n");
print Array(arr, n);
return 0;
}
```

5.
```
# include <stdio.h>
void binary_search (int [];int,int
void bubble sort(int[],int); int];
int main ()
{
int key, site, ];
int list [25]};
print f("Enter size of a list");
scan f (":%d", & size);
print f ("Enter element \n");
for (i=0; i<size, i++)
{
scan f ("%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]; ag
list[i] = list[j]
list[j] = temp;
}
}
}
}
void binary_search (int list[], int
```

```
- lo, int - hi, int key)
{
 int mid;
 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);
  {
}
```

*** The End ***