Sanchita Rawat
535
CSE-H

Date ___/___/___
Page _____

Assignment-6

1(a)
```
#include<stdio.h>

int bsearch(int a[], int l, int r, int x)
{ if (r>=l)
  { int mid =(l+r)/2;
    if (a[mid]==x)
        return mid;
    if(a[mid]>x)
        return bsearch(a,l,mid-1,x);
    return bsearch(a,mid+1,r,x);
    }
    return -1;
  }
void main()
{ int num[30], a, n;
  printf("Enter size");
  scanf("%d",&n);
  printf("Enter elements");
  for(int i=0; i<n; i++)
        scanf("%d",&num[i]);
  for(int i=0; i<n; i++)
  { for(j=i+1; j<n; j++)
    { if(num[i]<num[j])
      { a=num[i];
        num[i]=num[j];
        num[j]=a;
      }
    }
  }
```

```
printf("Elements in descending order are -");
 for(int k=0; k<n; k++)
      printf("%d \n", num[k]);
 int val=0;
 printf("Enter value to be searched");
 scanf("%d", &val);
 int result=bsearch(a,0,S-1,val;
(result==-1)?printf("Element is not present");
printf("Element is present at position %d", result);
 }
```

(b)

```
void main()
{int n1, n2, sum=0, product=0;
 printf("Enter location 1");
 scanf("%d", &n1);
 printf("Enter location 2");
 scanf("%d", &n2);
 sum=num[n1]+num[n2];
 product=num[n1]*num[n2];
 printf("Sum:%d", sum);
 printf("Product:%d", product);
}
```

Date __/__/__
Page _____

2.
```
//merge sort
#include<stdio.h>
void merge(int a[], int l, int m, int r)
{   int n1 = m-l+1;
    int n2 = r-m;
  int left[n1], right[n2];
  for(int i=0; i<n1; i++)
      left[i] = a[l+i];
  for(int j=0; j<n2; j++)
      right[j] = a[m+1+j];
  int i=0;
  int j=0;
  int k=l;
  while((i<n1)&&(j<n2))
  {   if(left[i]<=right[j])
      {  a[k] = left[i]
         i++;
      }
      else
      { a[k] = right[j];
        j++;
      }
      k++;
  }
  while(i<n1)
  { a[k] = left[i];
    i++;
    k++;
  }
  while(j<n2)
```

```
{ a[k]=right[j];
    j++;
    k++;
}

}
void merge_sort(int ar[], int l_i, int r_i)
{ if (l_i < r_i)
  { int m=l_i+(r_i-l_i)/2;
  merge_sort(ar, l_i, m);
  merge_sort (ar, m+1, r_i);
  merge(ar, l_i, m, r_i);
  }

}

void print (int A[], int s)
{ for(int i=0; i<s; i++)
      printf("%d", A[i]);
    printf("\n");
}

Void main()
{ int a[20], a_size;
  printf("Enter array size");
  scanf("%d", &a_size);
  printf("Enter array elements");
  for(int n=0; n<a_size; n++)
  scanf("%d", &array[n]);
  merge_sort(array, 0, a_size-1);
  printf("\n Sorted array is");
   print(array, a_size);
}
```

Date ___/___/___
Page _____

3. Insertion Sort -

It sorts the array one at a time. It is efficient for small data sets. At each iteration insertion sort removes one element from the array and places it at the location it belongs in the sorted array. Time complexity is $O(n^2)$ (average)

Eg — 4 3 2 10 12 1

3 4 2 10 12 1

2 3 4 10 12 1

2 3 4 10 12 1

2 3 4 10 12 1

1 2 3 4 10 12

Selection Sort —

In this technique the array is divided into 2 parts, a sorted sublist and an unsorted one. It finds the smallest element in the unsorted list and exchange it with the leftmost unsorted element. Time complexity is $O(n^2)$

20 12 10 15 2 min at index 1

20 12 10 15 2 min at index 2

20 12 10 15 2 min at index 2

20 12 10 15 2 min at index 4

2 12 10 15 20

swapping

4.
```
//bubble sort
#include <stdio.h>
void main ()
{ int a[20], n, temp=0;
  printf ("Enter size");
  scanf("%d", &n);
  printf ("Enter %d elements\n", n);
  for(int c=0; c<n; c++)
    scanf("%d", &a[c]);
  for(int c=0; c<n-1; c++)
  { for(int d=0; d<n-c-1; d++)
    { if(a[d] > a[d+1])
      { temp = a[d];
        a[d] = a[d+1];
        a[d+1] = temp;
      }
    }
  }
printf("Sorted list is:");
```

Date __/__/__
Page _____

```
print(int c=0; c<n; c++)
  printf("%d", a[c]);
(i) printf("Alternate elements are:"),
  for(int i=0; i<n; i=i+2)
      printf("%d\n", a[i]);
(ii) int sum=0, int product=1;
  printf("Sum of elements at odd position:");
  for(int d=0; d<n; d++)
   {if (d % 2 != 0)
       product = product * a[d];
   }
  printf("%d", product);
  printf("Sum of elements at odd position:"),
  for(int c=0; c<n; c++)
   {if (c % 2 != 0)
       sum = sum + a[c];
   }
  printf("%d", sum);

(iii) int m
  printf("Enter the number for checking"),
   scanf("%d", &m);
  for(int i=0; i<n; i++)
   {if (a[i] % m == 0)
      printf("%d", a[i]);
   }
  Printf("These elements are divisible by %d", m);
  }
```

Scanned with CamScanner

Date __/__/__
Page

5.
```
#include< stdio.h>
void b_search(int a[], int l, int r, int k,
{ int mid;
  if (l>r)
  { printf( "Value not found \n");
  }
  mid =(l + r) / 2 ;
  if( a[mid] = = k)
      printf ("Value found ");
  else if (a[mid] > k)
     b_search( a, l, r-1, k);
  else if
     b_search( a, mid+1, r, key);
}
void main()
{ int val, s ;a[20];
  printf ("Enter size ");
   scanf ("%d",&s);
  printf ("Enter elements");
  for(int i=0; i< s; i++)
     scanf ("%d", &a[i]);
  printf ("Enter value to be search
   scanf ("%d", &val);
   b_search( a, 0, s, val);
}
```