***EXPERIMENT-1***

**Aim**: - *To implement Bubble Sort, Selection Sort and Insertion Sort and find the time complexity.*

**Source Code:-**

*void swap(int arr[100],int a,int b)*

*{*

*int t=arr[a];*

*arr[a]=arr[b];*

*arr[b]=t;*

*}*

*void bubble\_sort(int arr[100],int size)*

*{*

*float start=clock();*

*int t=0;*

*for(int i=1;i<=size-1;i++)*

*{*

*for(int j=1;j<=size-1;j++)*

*{*

*if(arr[j-1]>=arr[j])*

*{*

*swap(arr,j-1,j);*

*}*

*}*

*}*

*cout<<endl<<"Time Complexity is:"<<(((float)clock()-start)/CLOCKS\_PER\_SEC);*

*}*

*void selection\_sort(int arr[100],int size)*

*{*

*float start=clock();*

*int min=0;*

*for (int i=0;i<size;i++)*

*{*

*min=i;*

*for(int j=i;j<size;j++)*

*{*

*if (arr[j]<=arr[min])*

*{*

*min=j;*

*}*

*swap(arr,i,min);*

*}*

*}*

*cout<<endl<<"Time Complexity is:"<<(((float)clock()-start)/CLOCKS\_PER\_SEC);*

*}*

*void insertion\_sort(int arr[100],int size)*

*{*

*float start=clock();*

*int temp=0;*

*int j=0;*

*for(int i=1;i<size;i++)*

*{*

*temp=i;*

*j=i-1;*

*while(arr[temp]<=arr[j] && j>=0)*

*{*

*// cout<<"Temp and j "<<temp<<" "<<j<<endl;*

*// cout<<"Array Elements: "<<arr[temp]<<" "<<arr[j]<<endl;*

*swap(arr,j,temp);*

*j=j-1;*

*temp=temp-1;*

*}*

*// cout<<"Main Loop"<<endl;*

*}*

*cout<<endl<<"Time Complexity is:"<<(((float)clock()-start)/CLOCKS\_PER\_SEC);*

*}*

*int main()*

*{*

*cout<<"Implement Array Sorting.............";*

*cout<<endl<<"Enter size of array(<100):";*

*int size;*

*cin>>size;*

*cout<<endl<<"Enter Array Elements............";*

*int arr[100];*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Enter Element "<<i+1<<" is:";*

*cin>>arr[i];*

*}*

*cout<<endl<<"Press 1. To implement Bubble Sort.";*

*cout<<endl<<"Press 2. To implement Insertion Sort.";*

*cout<<endl<<"Press 3. To implement Selection Sort.";*

*cout<<endl<<"Enter Choice:";*

*int ch;*

*cin>>ch;*

*switch(ch)*

*{*

*case 1:bubble\_sort(arr,size);*

*break;*

*case 2:insertion\_sort(arr,size);*

*break;*

*case 3:selection\_sort(arr,size);*

*break;*

*default:cout<<endl<<"Entered choice is wrong.......... ";*

*exit(0);*

*}*

*cout<<endl<<"Final Array is:";*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Element "<<i+1<<" is:"<<arr[i];*

*}*

*return 0;*

*}*

***Experiment-2(a)***

**Aim**: - *To implement Merge Sorting and find the time complexity*

**Source Code:-**

*#include <iostream>*

*using namespace std;*

*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;*

*j = 0;*

*k = l;*

*while (i < n1 && j < n2)*

*{*

*if (L[i] <= R[j])*

*{*

*arr[k] = L[i];*

*i++;*

*}*

*else*

*{*

*arr[k] = R[j];*

*j++;*

*}*

*k++;*

*}*

*while (i < n1)*

*{*

*arr[k] = L[i];*

*i++;*

*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;*

*mergeSort(arr, l, m);*

*mergeSort(arr, m+1, r);*

*merge(arr, l, m, r);*

*}*

*}*

*void printArray(int A[], int size)*

*{*

*int i;*

*for (i=0; i < size; i++)*

*printf("%d ", A[i]);*

*}*

*int main()*

*{*

*cout<<"Implement Merge Sorting.............";*

*cout<<endl<<"Enter size of array(<100):";*

*int size;*

*cin>>size;*

*cout<<endl<<"Enter Array Elements............";*

*int arr[100];*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Enter Element "<<i+1<<" is:";*

*cin>>arr[i];*

*}*

*float start=clock();*

*mergeSort(arr, 0, size - 1);*

*float end=clock();*

*cout<<endl<<("\nSorted array is: \n");*

*printArray(arr, size);*

*cout<<endl<<"Time Complexity is:"<<((end-start)/CLOCKS\_PER\_SEC);*

*return 0;*

*}*

***Experiment-2(b)***

**Aim**: - *To implement Quick Sort and find the time complexity*

**Source Code:-**

*#include <iostream>*

*using namespace std;*

*void swap(int\* a, int\* b)*

*{*

*int t = \*a;*

*\*a = \*b;*

*\*b = t;*

*}*

*int partition (int arr[], int low, int high)*

*{*

*int pivot = arr[high];*

*int i = (low - 1);*

*for (int j = low; j <= high- 1; j++)*

*{*

*if (arr[j] <= pivot)*

*{*

*i++;*

*swap(&arr[i], &arr[j]);*

*}*

*}*

*swap(&arr[i + 1], &arr[high]);*

*return (i + 1);*

*}*

*void quickSort(int arr[], int low, int high)*

*{*

*if (low < high)*

*{*

*int pi = partition(arr, low, high);*

*quickSort(arr, low, pi - 1);*

*quickSort(arr, pi + 1, high);*

*}*

*}*

*void printArray(int arr[], int size)*

*{*

*int i;*

*for (i=0; i < size; i++)*

*printf("%d ", arr[i]);*

*}*

*int main()*

*{*

*cout<<"Implement Quick Sorting.............";*

*cout<<endl<<"Enter size of array(<100):";*

*int size;*

*cin>>size;*

*cout<<endl<<"Enter Array Elements............";*

*int arr[100];*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Enter Element "<<i+1<<" is:";*

*cin>>arr[i];*

*}*

*float start=clock();*

*quickSort(arr, 0, size-1);*

*float end=clock();*

*cout<<endl<<("\nSorted array is: \n");*

*printArray(arr, size);*

*cout<<endl<<"Time Complexity is:"<<((end-start)/CLOCKS\_PER\_SEC);*

*return 0;*

*}*

***EXPERIMENT-3(a)***

**Aim: -** *Write a program to implement and find time complexity of Linear Search.*

**Source Code:-**

*#include <iostream>*

*#include <ctime>*

*using namespace std;*

*int linearSearch(int input[],int n,int num){*

*int i=0;*

*while(i<n){*

*if(input[i]==num) {*

*return i;*

*}*

*i++;*

*}*

*return -1;*

*}*

*int main()*

*{*

*cout<<"Implementing Linear Search.\n";*

*cout<<endl<<"Enter size of array(<100):";*

*int size;*

*cin>>size;*

*cout<<endl<<"Enter Array Elements............";*

*int arr[100];*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Enter Element "<<i+1<<" is:";*

*cin>>arr[i];*

*}*

*int num\_to\_search;*

*cout<<endl<<"Enter any number to search:";*

*cin>>num\_to\_search;*

*float start=clock();*

*int result=linearSearch(arr,size,num\_to\_search);*

*float end=clock();*

*(result == -1)? printf("Element is not present in array")*

*: printf("Element is present at index %d",*

*result);*

*cout<<endl<<"Time Complexity is:"<<((end-start)/CLOCKS\_PER\_SEC);*

*return 0;*

*}*

***EXPERIMENT-3(b)***

**Aim: -** *Write a program to implement* *and find time complexity of Binary Search.*

**Source Code:-**

*#include <iostream>*

*#include <ctime>*

*using namespace std;*

*int binarySearch(int arr[], int l, int r, int x)*

*{*

*if (r >= l)*

*{*

*int mid = l + (r - l)/2;*

*if (arr[mid] == x)*

*return mid;*

*if (arr[mid] > x)*

*return binarySearch(arr, l, mid-1, x);*

*return binarySearch(arr, mid+1, r, x);*

*}*

*return -1;*

*}*

*int main()*

*{*

*cout<<"Implement Binary Search.............";*

*cout<<endl<<"Enter size of array(<100):";*

*int size;*

*cin>>size;*

*cout<<endl<<"Enter Array Elements............";*

*int arr[100];*

*for(int i=0;i<size;i++)*

*{*

*cout<<endl<<"Enter Element "<<i+1<<" is:";*

*cin>>arr[i];*

*}*

*int num\_to\_search;*

*cout<<endl<<"Enter any number to search:";*

*cin>>num\_to\_search;*

*float start=clock();*

*int result = binarySearch(arr, 0, size-1, num\_to\_search);*

*float end=clock();*

*(result == -1)? printf("Element is not present in array")*

*: printf("Element is present at index %d",*

*result);*

*cout<<endl<<"Time Complexity is:"<<((end-start)/CLOCKS\_PER\_SEC);*

*return 0;*

*}*