Name: Sanket Wakankar

Reg. No.: 2020BIT055

Practical No. 3: Write C++ code to implement concept of

1. Searching Algorithm
2. Sorting Algorithm

\*\*\*LINEAR SEARCH\*\*\*

// Sanket Wakankar

#include<iostream>

using namespace std;

void search(int arr[],int m,int t){

    for(int i=0;i<m;i++){

        if(arr[i]==t){

            cout<<"target found at index "<<i<<endl;

            return;

        }

    }

    cout<<"target not found"<<endl;

}

int main(){

    int n;

    cout<<"please enter the no. you want in a array"<<endl;

    cin>>n;

    int arr[n];

    cout<<"please enter the numbers in array"<<endl;

    for(int i=0;i<n;i++){

        cin>>arr[i];

    }

    int target;

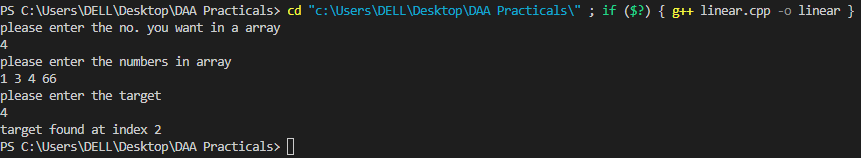
    cout<<"please enter the target"<<endl;

    cin>>target;

    search(arr,n,target);

}

OUTPUT:



\*\*\*BINARY SEARCH\*\*\*

// Sanket Wakankar

#include<iostream>

using namespace std;

int binary\_search(int narr[],int t,int c){

    int low = 0;

    int high = c-1;

    while(low<=high){

        int mid = (low + high)/2;

        if(t==narr[mid]){

            return mid;

        }

        else if(narr[mid]<t){

            low = mid+1;

        }

        else if(narr[mid]>t){

            high = mid-1;

        }

    }

    return -1;

}

int main(){

    int narr[] = {1,2,3,4,5};

    int target = 9;

    int ans = binary\_search(narr,target,5);

    if(ans==-1){

        cout<<"target not found"<<endl;

    }

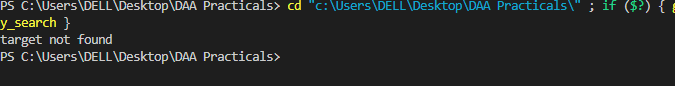
    else{

        cout<<"target found at index "<<ans<<endl;

    }

}

OUTPUT:



\*\*\*JUMP SEARCH\*\*\*

// Sanket Wakankar

#include<iostream>

#include<cmath>

using namespace std;

int jump\_Search(int a[], int n, int item) {

   int i = 0;

   int m = sqrt(n);

   while(a[m] <= item && m < n) {

      i = m;

      m += sqrt(n);

      if(m > n - 1)

         return -1;

   }

   for(int x = i; x<m; x++) {

      if(a[x] == item)

         return x;

   }

   return -1;

}

int main() {

   int n, item, loc;

   cout << "\n Enter number of items: ";

   cin >> n;

   int arr[n]; //creating an array of size n

   cout << "\n Enter items: ";

   for(int i = 0; i< n; i++) {

      cin >> arr[i];

   }

   cout << "\n Enter search key to be found in the array: ";

   cin >> item;

   loc = jump\_Search(arr, n, item);

   if(loc>=0)

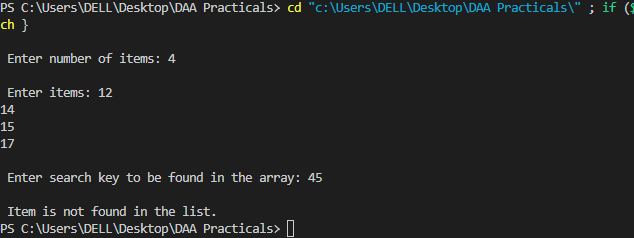
      cout << "\n Item found at location: " << loc;

   else

      cout << "\n Item is not found in the list.";

}

OUTPUT:



\*\*\*BUBBLE SORT\*\*\*

// Sanket Wakankar

#include <iostream>

using namespace std;

void bubbleSort(int arr[], int n) {

    for (int i = 0; i < n - 1; i++) {

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

            if (arr[j] > arr[j + 1]) {

                int temp = arr[j];

                arr[j] = arr[j + 1];

                arr[j + 1] = temp;

            }

        }

    }

}

int main() {

    int arr[] = {89,23,55,23,78,1,0,100};

    int n = sizeof(arr)/sizeof(arr[0]);

    bubbleSort(arr, n);

    cout << "Sorted array: \n";

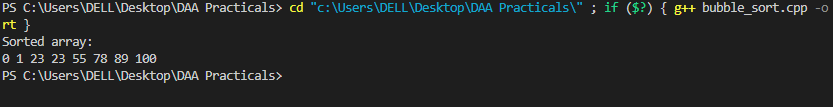
    for (int i=0; i < n; i++)

        cout << arr[i] << " ";

    return 0;

}

OUTPUT:



\*\*\*QUICK SORT\*\*\*

// Sanket Wakankar

#include <iostream>

using namespace std;

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

    int x = arr[high];

    int i = (low - 1);

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

        if (arr[j] < x) {

            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);

    }

}

int main() {

    int arr[] = {12,43,11,33,66,32,20,100};

    int n = sizeof(arr) / sizeof(arr[0]);

    quickSort(arr, 0, n - 1);

    cout << "Sorted array: \n";

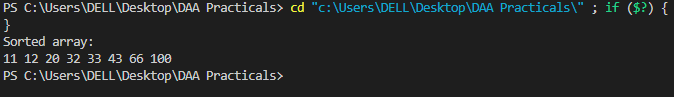
    for (int i = 0; i < n; i++)

        cout << arr[i] << " ";

    return 0;

}

OUTPUT:



\*\*\*SELECTION SORT\*\*\*

// Sanket Wakankar

#include <iostream>

using namespace std;

void selectionSort(int arr[], int n) {

    int i, j, minIndex, temp;

    for (i = 0; i < n-1; i++) {

        minIndex = i;

        for (j = i+1; j < n; j++) {

            if (arr[j] < arr[minIndex])

                minIndex = j;

        }

        temp = arr[minIndex];

        arr[minIndex] = arr[i];

        arr[i] = temp;

    }

}

int main() {

    int arr[] = {32,12,44,2,41,67,76,100};

    int n = sizeof(arr)/sizeof(arr[0]);

    selectionSort(arr, n);

    cout << "Sorted array: \n";

    for (int i=0; i < n; i++)

        cout << arr[i] << " ";

    return 0;

}

OUTPUT:

