Design and Analysis of Algorithms

NAME : SHUBHAM SHIVRAJ SURYAWANSHI

ASSIGNMENT = 3

1) Leanear search

STRING > basic_cpp > Array_practice > leanear_search.cpp > main()

```cpp
#include<iostream>
using namespace std;
int main(){
 int A[10],n=5,i,key;
 cout<<"Enter a number ";
 for (i=0;i<n;i++)
 cin>>A[i];
 cout<<"Enter a key";
 cin>>key;
 for(i=0;i<n;i++){
    if(key==A[i]){
    cout<<"FOUND";
    return 0;
    }
 }
 cout<<"NOT FOUND";
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
insert element at front end
front element: 8
after deletion of front element new front element: 5
PS D:\DSA PRACTICE\DAA_PRACTICAL> cd "d:\DSA PRACTICE\STRING\b
 } ; if ($?) { .\leanear_search }
Enter a number 1 2 3 4 5
Enter a key5
FOUND
```

2)Binary Search

DAA_PRACTICAL > binary_search.cpp > ...

```cpp
#include <iostream>
using namespace std;
int binarySearch(int arr[], int n, int x) {
 int low = 0;
 int high = n - 1;
 while (low <= high) {
 int mid = low + (high - low) / 2;
 if (arr[mid] == x)
 return mid;
 if (arr[mid] < x)
 low = mid + 1;
 else
 high = mid - 1;
 }
 return -1;
}
int main() {
 int arr[] = {1, 2, 3, 4, 5};
 int n = sizeof(arr) / sizeof(arr[0]);
 int x = 4;
 int result = binarySearch(arr, n, x);
 if (result == -1)
 cout << "Element not found." << endl;
 else
 cout << "Element found at index " << result << endl;
 return 0;
}

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Element found at index 3
PS D:\DSA PRACTICE\DAA_PRACTICAL>
```

3)Jump Search

```cpp
#include <bits/stdc++.h>
using namespace std;
int jumpSearch(int arr[], int x, int n)
{
    int step = sqrt(n);
    int prev = 0;
    while (arr[min(step, n)-1] < x)
    {
        prev = step;
        step += sqrt(n);
        if (prev >= n)
            return -1;
    }
    while (arr[prev] < x)
    {
        prev++;
        if (prev == min(step, n))
            return -1;
    }
    if (arr[prev] == x)
        return prev;
    return -1;
}
int main()
{
    int arr[] = { 0, 1, 1, 2, 3, 5, 8, 13, 21,
                  34, 55, 89, 144, 233, 377, 610 };
    int x = 55;
    int n = sizeof(arr) / sizeof(arr[0]);
    int index = jumpSearch(arr, x, n);
    cout << "\nNumber " << x << " is at index " << index;
    return 0;
}
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

Number 55 is at index 10
PS D:\DSA PRACTICE\DAA_PRACTICAL>
```

SORTING

1) Selection Sort

```
1   //SHUBHAM SHIVRAJ SURYAWANSHI
2   #include <bits/stdc++.h>
3   using namespace std;
4   void swap(int *xp, int *yp)
5   {
6       int temp = *xp;
7       *xp = *yp;
8       *yp = temp;
9   }
10  void selectionSort(int arr[], int n)
11  {
12      int i, j, min_idx;
13      for (i = 0; i < n-1; i++)
14      {
15          min_idx = i;
16          for (j = i+1; j < n; j++)
17          if (arr[j] < arr[min_idx])
18              min_idx = j;
19          if(min_idx!=i)
20              swap(&arr[min_idx], &arr[i]);
21      }
22  }
```

```
23  void printArray(int arr[], int size)
24  {
25      int i;
26      for (i=0; i < size; i++)
27          cout << arr[i] << " ";
28      cout << endl;
29  }
30  int main()
31  {
32      int arr[] = {64, 25, 12, 22, 11};
33      int n = sizeof(arr)/sizeof(arr[0]);
34      selectionSort(arr, n);
35      cout << "Sorted array: \n";
36      printArray(arr, n);
37      return 0;
38  }
39
40
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Sorted array:
11 12 22 25 64
PS D:\DSA PRACTICE\DAA_PRACTICAL>
```

2)Bubble sort

DAA_PRACTICAL > bubble_sort.cpp > ...

```cpp
#include <bits/stdc++.h>
using namespace std;
void bubbleSort(int arr[], int n)
{
    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 printArray(int arr[], int size)
{
    int i;
    for (i = 0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}
int main()
{
    int arr[] = { 5, 1, 4, 2, 8};
    int N = sizeof(arr) / sizeof(arr[0]);
    bubbleSort(arr, N);
    cout << "Sorted array: \n";
    printArray(arr, N);
    return 0;
}
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

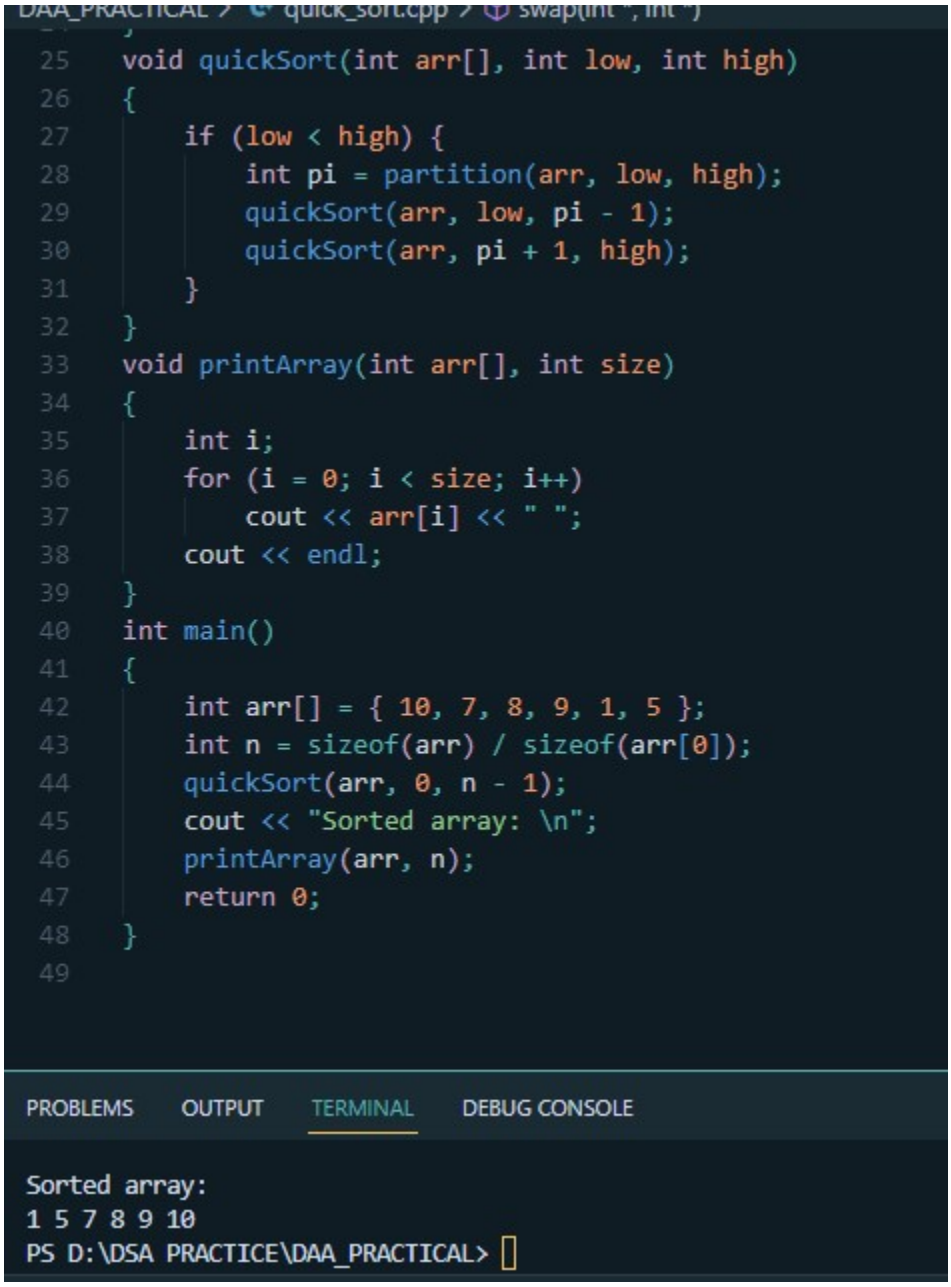
```
Sorted array:
1 2 4 5 8
PS D:\DSA PRACTICE\DAA_PRACTICAL>
```

3)Quick sort

DAA_PRACTICAL > quick_sort.cpp > swap(int *, int *)

```cpp
#include <bits/stdc++.h>
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);
}
```



```
DAA_PRACTICAL > quick_sort.cpp > swap(int *, int *)
25  void quickSort(int arr[], int low, int high)
26  {
27      if (low < high) {
28          int pi = partition(arr, low, high);
29          quickSort(arr, low, pi - 1);
30          quickSort(arr, pi + 1, high);
31      }
32  }
33  void printArray(int arr[], int size)
34  {
35      int i;
36      for (i = 0; i < size; i++)
37          cout << arr[i] << " ";
38      cout << endl;
39  }
40  int main()
41  {
42      int arr[] = { 10, 7, 8, 9, 1, 5 };
43      int n = sizeof(arr) / sizeof(arr[0]);
44      quickSort(arr, 0, n - 1);
45      cout << "Sorted array: \n";
46      printArray(arr, n);
47      return 0;
48  }
49

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

Sorted array:
1 5 7 8 9 10
PS D:\DSA PRACTICE\DAA_PRACTICAL>
```