**Rawal Institute of Engineering & Technology**

**Faridabad**

**Department of Computer Science & Engineering**

**Course: B.Tech 3rd Semester**

**Data Structure & Algorithms Lab (PCC-CS-303)**

(Session: Aug - Dec, 2023)

**Submitted By Submitted To**

(Student Name) (Mr. Raj Kumar)

(Roll No) (Assistant Professor)

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No.** | **Programs** | **Date** | **Signature** |
| 1 | Write programs on String |  |  |
| 2 | Write programs on Array |  |  |
| 3 | Write a program to search an element from an array using Linear Search |  |  |
| 4 | Write a program to search an element from an array using Binary Search |  |  |
| 5 | Write a program to push, pop and display the elements in a stack using array |  |  |
| 6 | Write a program to insert, delete and display the elements in a queue using array |  |  |
| 7 | Write a program to insert, delete and display the elements in a linked list |  |  |
| 8 | Write a program to insert, delete and display the elements in a binary search tree |  |  |
| 9 | Write a program to sort elements of an array using selection sort |  |  |
| 10 | Write a program to sort elements of an array using insertion sort |  |  |
| 11 | Write a program to sort elements of an array using bubble sort |  |  |
| 12 | Write a program to sort elements of an array using quick sort |  |  |
| 13 | Write a program to sort elements of an array using merge sort |  |  |
| 14 | Write a program to sort elements using heap sort |  |  |

Program-1

/\*Write a programs to print a String as char array & string literal\*/

#include<stdio.h>

#include <string.h>

**int** main()

{

**char** ch[11]={'w', 'e', 'l', 'c', 'o', 'm', 'e'', '\0'};   // By char array

**char** ch2[11]="welcome";  // By string literal

   printf("Char Array Value is: %s\n", ch);

  printf("String Literal Value is: %s\n", ch2);

**return** 0;

}

Output:- welcome

Welcome

/\*Accepting string as input & find the frequency of a character\*/

#include <stdio.h>

int main() {

char str[1000], ch;

int count = 0;

printf("Enter a string: ");

fgets(str, sizeof(str), stdin);

printf("Enter a character to find its frequency: ");

scanf("%c", &ch);

for (int i = 0; str[i] != '\0'; ++i) {

if (ch == str[i])

++count;

}

printf("Frequency of %c = %d", ch, count);

return 0;

}

Output:

Enter a string: This is awesome.

Enter a character to find its frequency: e

Frequency of e = 2

/\*Write a program toFind the length of a string\*/

#include<stdio.h>

#include <string.h>

**int** main(){

**char** ch[20]= {'w', 'e', 'l', 'c', 'o', 'm', 'e'', '\0'};

   printf("Length of string is: %d",strlen(ch));

**return** 0;

}

Output:

Length of the string: 7

# /\*Write a program to Copy one string to other\*/

#include <stdio.h>

int main()

{

char s1[100], s2[100], i;

printf("Enter string s1: ");

fgets(s1, sizeof(s1), stdin);

for (i = 0; s1[i] != '\0'; ++i) {

s2[i] = s1[i];

}

s2[i] = '\0';

printf("String s2: %s", s2);

return 0;

}

Output:

Enter string s1: c programmer.

String s2: c programmer.

# /\*Write a program to concatenate of two string\*/

#include<stdio.h>

#include <string.h>

**int** main(){

**char** ch[10]={'h', 'e', 'l', 'l', 'o', '\0'};

**char** ch2[10]=”world”;

   strcat(ch,ch2);

   printf("The concatenated string is:- %s",ch);

**return** 0;

}

Output:

The concatenated string is :- hello world

‘

Program-2

/\*Write a Program to input and output one dimensional array\*/

#include <stdio.h>

int main()

{

int values[5];

printf("Enter 5 integers: ");

for(int i = 0; i < 5; ++i) {

scanf("%d", &values[i]);

}

printf("Displaying integers: ");

for(int i = 0; i < 5; ++i) {

printf("%d\n", values[i]);

}

return 0;

}

Output:-

Enter 5 integers: 1

-3

34

0

3

Displaying integers: 1

-3

34

0

3

/\*Write a Program to input and output two dimensional array\*/

#include<stdio.h>

int main(){

int i=0,j=0,m,n;

int arr[5][4];

printf("enter size of array:-");

scanf("%d%d",&m,&n);

printf("enter the elements of array:-");

for(i=0;i<m;i++)

{

for(j=0;j<n;j++)

{

scanf("%d",&arr[i][j]);

}

}

for(i=0;i<m;i++)

{

for(j=0;j<n;j++)

{

printf("arr[%d] [%d] = %d \n",i,j,arr[i][j]);

}

}

return 0;

}

Output:-

enter the size of array:-

2

2

enter the elements of array:-

56

10

30

34

arr[0] [0]=56

arr[0] [1]=10

arr[0] [2]= 30

arr[1] [0]= 34

Program-3

/\*Write a Program for Linear search in an Array\*/

#include <stdio.h>

int main()

{

int array[100], search, c, n;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter elements of array", n);

for (c = 0; c < n; c++)

{

scanf("%d", &array[c]);

}

for (c = 0; c < n; c++)

{

printf("%d\n", array[c]);

}

printf("Enter an element you want to search\n");

scanf("%d", &search);

for (c = 0; c < n; c++)

{

if (array[c] == search) /\* If required element is found \*/

{

printf("%d is present at location %d.\n", search, c);

break;

}

}

if (c == n)

printf("%d isn't present in the array.\n", search);

return 0;

}

Output:-

enter the size of array 4

enter elements of array 23

45

67

78

Enter an element you want to search 67

67 is present at location 2

Program-4

/\*Write a Program for Binary search in an Array\*/

#include <stdio.h>

int main()

{

int c, first, last, middle, n, search, array[100];

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter elements of array", n);

for (c = 0; c < n; c++)

{

scanf("%d", &array[c]);

}

printf("Enter value to find\n");

scanf("%d", &search);

first = 0;

last = n - 1;

middle = (first+last)/2;

while (first <= last) {

if (array[middle] < search)

first = middle + 1;

else if (array[middle] == search) {

printf("%d found at location %d.\n", search, middle+1);

break;

}

else

last = middle - 1;

middle = (first + last)/2;

}

if (first > last)

printf("Not found! %d isn't present in the list.\n", search);

return 0;

}

Output:-

enter the size of array 4

enter elements of array 32

45

56

78

Enter an element you want to search 78

78 is present at location 3

Program-5

/\*Write a Program to implement stack operations\*/

#include <stdio.h>

int stack[100],i,j,choice=0,n,top=-1;

void push();

void pop();

void show();

int main ()

{

printf("Enter the number of elements in the stack ");

scanf("%d",&n);

while(choice!=4)

{

printf("Chose one from the below options...\n");

printf("\n1.Push\n2.Pop\n3.Show\n4.Exit");

printf("\n Enter your choice \n");

scanf("%d",&choice);

switch(choice)

{

case 1:

{

push();

break;

}

case 2:

{

pop();

break;

}

case 3:

{

show();

break;

}

case 4:

{

printf("Exiting....");

break;

}

default:

{

printf("Please Enter valid choice ");

}

}

}

}

void push()

{

int val;

if (top == n )

{

printf("\n Overflow");

}

else

{

printf("Enter the value?");

scanf("%d",&val);

top = top +1;

stack[top] = val;

}

}

void pop()

{

if(top == -1)

printf("Underflow");

else

top = top -1;

}

void show()

{

for (i=top;i>=0;i--)

{

printf("%d\n",stack[i]);

}

if(top == -1)

{

printf("Stack is empty");

}

}

Output:-

Enter the number of elements in the stack 4

Chose one from the below options...

1.Push

2.Pop

3.Show

4.Exit

Enter your choice 1

Enter the value 35

Chose one from the below options...

1.Push

2.Pop

3.Show

4.Exit

Enter your choice 1

Enter the value 55

Chose one from the below options...

1.Push

2.Pop

3.Show

4.Exit

Enter your choice 3

55

35

Chose one from the below options...

1.Push

2.Pop

3.Show

4.Exit

Enter your choice 2

Chose one from the below options...

1.Push

2.Pop

3.Show

4.Exit

Enter your choice 3

35

Program-6

/\*Write a Program to implement Queue operations\*/

#include<stdio.h>

#include<stdlib.h>

void inst();

void del();

void display();

int front = -1, rear = -1,max;

int queue[20];

int main ()

{

printf("\nEnter size of queue ?");

scanf("%d",&max);

int choice;

while(choice != 4)

{

printf("\n1.insert an element\n2.delete an element\n3.display the queue\n4.Exit\n");

printf("\nEnter your choice ?");

scanf("%d",&choice);

switch(choice)

{

case 1:

inst();

break;

case 2:

del();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("\nEnter valid choice??\n");

}

}

}

void inst()

{

int item;

printf("\nEnter the element\n");

scanf("\n%d",&item);

if(rear == max-1)

{

printf("\nOVERFLOW\n");

return;

}

if(front == -1 && rear == -1)

{

front = 0;

rear = 0;

}

else

{

rear = rear+1;

}

queue[rear] = item;

printf("\nValue inserted ");

}

void del()

{

int item;

if (front == -1 || front > rear)

{

printf("\nUNDERFLOW\n");

}

else

{

item = queue[front];

if(front == rear)

{

front = -1;

rear = -1 ;

}

else

{

front = front + 1;

}

printf("\nvalue deleted ");

}

}

void display()

{

int i;

if(rear == -1)

{

printf("\nEmpty queue\n");

}

else

{ printf("\nprinting values .....\n");

for(i=front;i<=rear;i++)

{

printf("\n%d\n",queue[i]);

}

}

}

Output:-

Enter the size of queue 5

Enter your choice

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 1

Enter the element 45

Enter your choice

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 1

Enter the value 78

Enter your choice

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 3

45

78

Enter your choice

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 2

Enter your choice

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 3

78

Program-7

/\*Write a Program to implement Linked List operations\*/

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head;

void insert ();

void del();

void display();

void search();

int main ()

{

int choice =0;

while(choice != 9)

{

printf("\n1.Insert\n2.Delete\n3.Show\n4.Exit\n");

printf("\nEnter your choice?\n");

scanf("\n%d",&choice);

switch(choice)

{

case 1:

insert();

break;

case 2:

del();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("Please enter valid choice..");

}}}

void insert()

{

struct node \*ptr;

int item;

ptr = (struct node \*) malloc(sizeof(struct node \*));

printf("\nEnter value\n");

scanf("%d",&item);

ptr->data = item;

ptr->next = head;

head = ptr;

printf("\nNode inserted");

}

void del()

{

struct node \*ptr;

if(head == NULL)

{

printf("\nList is empty\n");

}

else

{

ptr = head;

head = ptr->next;

free(ptr);

printf("\nNode deleted from the begining ...\n");

}

}

void display()

{

struct node \*ptr;

ptr = head;

if(ptr == NULL)

{

printf("Nothing to print");

}

else

{

printf("\nprinting values . . . . .\n");

while (ptr!=NULL)

{

printf("\n%d",ptr->data);

ptr = ptr -> next;

}

}

}

Output:-

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 1

Enter value 23

Node inserted

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 1

Enter value 56

Node inserted

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 3

Printing values

56

23

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 2

1.insert an element

2.delete an element

3.display the queue

4.Exit

Enter your choice 3

Printing values

23

Program-8

/\*Write a Program to implement Binary Search Tree operations\*/

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*rlink;

struct node \*llink;

}\*tmp=NULL;

typedef struct node NODE;

NODE \*create();

void preorder(NODE \*);

void inorder(NODE \*);

void postorder(NODE \*);

void insert(NODE \*);

int main()

{

int n,i,ch;

do

{

printf("\n\n1.Create\n\n2.Insert\n\n3.Preorder\n\n4.Postorder\n\n5.Inorder\n\n6.Exit\n\n");

printf("\n\nEnter Your Choice : ");

scanf("%d",&ch);

switch(ch)

{

case 1:

tmp=create();

break;

case 2:

insert(tmp);

break;

case 3:

printf("\n\nDisplay Tree in Preorder Traversal : ");

preorder(tmp);

break;

case 4:

printf("\n\nDisplay Tree in Postorder Traversal : ");

postorder(tmp);

break;

case 5:

printf("\n\nDisplay Tree in Inorder Traversal : ");

inorder(tmp);

break;

case 6:

exit(0);

default:

printf("\n Inavild Choice..");

}

}

while(n!=5);

}

void insert(NODE \*root)

{

NODE \*newnode;

if(root==NULL)

{

newnode=create();

root=newnode;

}

else

{

newnode=create();

while(1)

{

if(newnode->data<root->data)

{

if(root->llink==NULL)

{

root->llink=newnode;

break;

}

root=root->llink;

}

if(newnode->data>root->data)

{

if(root->rlink==NULL)

{

root->rlink=newnode;

break;

}

root=root->rlink;

}

}

}

}

NODE \*create()

{

NODE \*newnode;

int n;

newnode=(NODE \*)malloc(sizeof(NODE));

printf("\n\nEnter the Data ");

scanf("%d",&n);

newnode->data=n;

newnode->llink=NULL;

newnode->rlink=NULL;

return(newnode);

}

void postorder(NODE \*tmp)

{

if(tmp!=NULL)

{

postorder(tmp->llink);

postorder(tmp->rlink);

printf("%d->",tmp->data);

}

}

void inorder(NODE \*tmp)

{

if(tmp!=NULL)

{

inorder(tmp->llink);

printf("%d->",tmp->data);

inorder(tmp->rlink);

}

}

void preorder(NODE \*tmp)

{

if(tmp!=NULL)

{

printf("%d->",tmp->data);

preorder(tmp->llink);

preorder(tmp->rlink);

}

}

Program-9

/\*Write a Program to implement selection sort\*/

#include <stdio.h>

int main()

{

int a[100], n,small,i,j,temp;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter the elements");

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

printf("Before sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

for (i = 0; i < n; i++)

{

small = i;

for (j = i+1; j < n; j++)

if (a[j] < a[small])

small = j;

temp = a[small];

a[small] = a[i];

a[i] = temp;

}

printf("\nAfter sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

return 0;

}

Output:-

Enter number of elements in array 4

Enter the elements

67

90

56

12

Before sorting array elements are:-

67

90

56

12

After sorting array elements are:-

12

56

67

90

Program-10

/\*Write a Program to implement insertion sort\*/

#include <stdio.h>

int main()

{

int a[100], n,small,i,j,temp;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

printf("Before sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

for (i = 1; i < n; i++) {

temp = a[i];

j = i - 1;

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

{

a[j+1] = a[j];

j = j-1;

}

a[j+1] = temp;

}

printf("\nAfter sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

return 0;

}

Output:-

Enter number of elements in array 4

Enter the elements

56

89

53

17

Before sorting array elements are:-

56

89

53

17

After sorting array elements are:-

17

53

56

89

Program-11

/\*Write a Program to implement bubble sort\*/

#include <stdio.h>

int main()

{

int a[100], n,small,i,j,temp;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

printf("Before sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

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

{

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

{

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

temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

printf("\nAfter sorting array elements are - \n");

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

return 0;

}

Output:-

Enter number of elements in array 4

Enter the elements

56

67

63

19

Before sorting array elements are:-

56

67

63

19

After sorting array elements are:-

19

56

63

67

Program-12

/\*Write a Program to implement quick sort\*/

#include <stdio.h>

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\n", arr[i]);

}

int main() {

int arr[100], n,small,i,j,temp;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

for (i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

printf("Before sorting array elements are - \n");

for (i = 0; i < n; i++)

{

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

}

quickSort(arr, 0, n - 1);

printf("Sorted array:\n");

printArray(arr, n);

return 0;

}

Output:-

Enter number of elements in array 4

Enter the elements

59

34

78

23

Before sorting array elements are:-

59

34

78

23

After sorting array elements are:-

23

34

59

78

Program-13

/\*Write a Program to implement merge sort\*/

#include <stdio.h>

void printArray(int A[], int n)

{

for (int i = 0; i < n; i++)

{

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

}

printf("\n");

}

void merge(int A[], int mid, int low, int high)

{

int i, j, k, B[100];

i = low;

j = mid + 1;

k = low;

while (i <= mid && j <= high)

{

if (A[i] < A[j])

{

B[k] = A[i];

i++;

k++;

}

else

{

B[k] = A[j];

j++;

k++;

}

}

while (i <= mid)

{

B[k] = A[i];

k++;

i++;

}

while (j <= high)

{

B[k] = A[j];

k++;

j++;

}

for (int i = low; i <= high; i++)

{

A[i] = B[i];

}

}

void mergeSort(int A[], int low, int high)

{

int mid;

if(low<high){

mid = (low + high) /2;

mergeSort(A, low, mid);

mergeSort(A, mid+1, high);

merge(A, mid, low, high);

}

}

int main()

{

int A[] = {9, 14, 4, 8, 7, 5, 6};

int n = 7;

printf("array before sorting:-");

printArray(A, n);

mergeSort(A, 0, 6);

printf("array after sorting:-");

printArray(A, n);

return 0;

}

Output:-

array before sorting:- 9 14 4 8 7 5 6

array after sorting:- 4 5 6 7 8 9 14

Program-14

/\*Write a Program to implement Heap sort\*/

#include <stdio.h>

void heapify(int a[], int n, int i)

{

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if (left < n && a[left] > a[largest])

largest = left;

if (right < n && a[right] > a[largest])

largest = right;

if (largest != i) {

int temp = a[i];

a[i] = a[largest];

a[largest] = temp;

heapify(a, n, largest);

}

}

void heapSort(int a[], int n)

{

for (int i = n / 2 - 1; i >= 0; i--)

heapify(a, n, i);

for (int i = n - 1; i >= 0; i--) {

int temp = a[0];

a[0] = a[i];

a[i] = temp;

heapify(a, i, 0);

}

}

void printArr(int arr[], int n)

{

for (int i = 0; i < n; ++i)

{

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

printf(" ");

}

}

int main()

{

int a[] = {48, 10, 23, 43, 28, 26, 1};

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

printf("Before sorting array elements are - \n");

printArr(a, n);

heapSort(a, n);

printf("\nAfter sorting array elements are - \n");

printArr(a, n);

return 0;

}

Output:-

Before sorting array elements are-

48 10 23 43 28 26 1

After sorting array elements are –

1 10 23 26 28 43 48