1. Write a program in C/C++/Java/Python to implement “Array” data structure. Perform operations like add or insert, delete or remove and display. Your program should be able to add/insert at any position in an array or remove/delete any element from an array. Take care of extreme conditions such as empty array or full array and display appropriate message to user.

#include<stdio.h>

#include<stdlib.h>

int a[100], n;

void create()

{

int i,j;

printf("enter number of elements");

scanf("%d", &n);

printf("Enter the elements");

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

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

return ;

}

void display()

{

int i;

printf("Elements are:");

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

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

return;

}

void insert()

{

int ele, pos, i;

printf("enter the element to be inserted");

scanf("%d", &ele);

printf("enter the position in which u want to insert");

scanf("%d", &pos);

if(pos>=n+1||pos<0)

{

printf("Invalid position! Please enter position between 0 to %d", n);

return;

}

else

{

for(i=n; i>=pos; i--)

a[i] = a[i-1];

n++;

a[pos] = ele;

printf("Array elements after insertion : ");

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

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

}

}

void delete()

{

int ele, pos,i;

printf("enter the position from which u want to delete");

scanf("%d", &pos);

if(pos>=n+1 || pos<0)

{

printf("Invalid position! Please enter position between 0 to %d", n-1);

return;

}

else

{

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

a[i] = a[i+1];

}

printf("\nElements of array after delete are : ");

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

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

n--;

}

void main()

{

int choice;

while(1)

{

printf("\n1 : create array\n2 : display array\n3 : insert element\n4 : delete element\n5 : exit");

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

scanf("%d",&choice);

switch(choice)

{

case 1: create();

break;

case 2: display();

break;

case 3: insert();

break;

case 4: delete();

break;

case 5: exit(0);

default:printf("invalid option");

break;

}

}

}

1. Write a program in C/C++/Java/Python to implement “Doubly Linked List (DLL)” data structure. Perform operations like insert or add, remove, or delete and display. Your program should be able to add/insert at any position in DLL or remove/delete any element from an DLL. Take care of extreme conditions such as empty DLL or full DLL and display appropriate message to user.

#include <stdio.h>

#include <stdlib.h>

struct node {

int info;

struct node \*prev, \*next;

};

struct node\* start = NULL;

void traverse()

{

if (start == NULL) {

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

return;

}

struct node\* temp;

temp = start;

while (temp != NULL) {

printf("Data = %d\n", temp->info);

temp = temp->next;

}

}

void insertAtFront()

{

int data;

struct node\* temp;

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

printf("\nEnter number to be inserted: ");

scanf("%d", &data);

temp->info = data;

temp->prev = NULL;

temp->next = start;

start = temp;

}

void insertAtEnd()

{

int data;

struct node \*temp, \*trav;

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

temp->prev = NULL;

temp->next = NULL;

printf("\nEnter number to be inserted: ");

scanf("%d", &data);

temp->info = data;

temp->next = NULL;

trav = start;

if (start == NULL) {

start = temp;

}

else {

while (trav->next != NULL)

trav = trav->next;

temp->prev = trav;

trav->next = temp;

}

}

void insertAtPosition()

{

int data, pos, i = 1;

struct node \*temp, \*newnode;

newnode = malloc(sizeof(struct node));

newnode->next = NULL;

newnode->prev = NULL;

printf("\nEnter position : ");

scanf("%d", &pos);

if (start == NULL) {

start = newnode;

newnode->prev = NULL;

newnode->next = NULL;

}

else if (pos == 1) {

/\* newnode->next = start;

newnode->next->prev = newnode;

newnode->prev = NULL;

start = newnode; \*/

insertAtFront();

}

else {

printf("\nEnter number to be inserted: ");

scanf("%d", &data);

newnode->info = data;

temp = start;

while (i < pos - 1) {

temp = temp->next;

i++;

}

newnode->next = temp->next;

newnode->prev = temp;

temp->next = newnode;

temp->next->prev = newnode;

}

}

void deleteFirst()

{

struct node\* temp;

if (start == NULL)

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

else {

temp = start;

start = start->next;

if (start != NULL)

start->prev = NULL;

free(temp);

}

}

void deleteEnd()

{

struct node\* temp;

if (start == NULL)

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

temp = start;

while (temp->next != NULL)

temp = temp->next;

if (start->next == NULL)

start = NULL;

else {

temp->prev->next = NULL;

free(temp);

}

}

void deletePosition()

{

int pos, i = 1;

struct node \*temp, \*position;

temp = start;

if (start == NULL)

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

else {

printf("\nEnter position : ");

scanf("%d", &pos);

if (pos == 1) {

deleteFirst(); // im,proved by Jay Ghughriwala on GeeksforGeeks

if (start != NULL) {

start->prev = NULL;

}

free(position);

return;

}

while (i < pos - 1) {

temp = temp->next;

i++;

}

position = temp->next;

if (position->next != NULL)

position->next->prev = temp;

temp->next = position->next;

free(position);

}

}

int main()

{

int choice;

while (1) {

printf("\n\t1 To see list\n");

printf("\t2 For insertion at"

" starting\n");

printf("\t3 For insertion at"

" end\n");

printf("\t4 For insertion at "

"any position\n");

printf("\t5 For deletion of "

"first element\n");

printf("\t6 For deletion of "

"last element\n");

printf("\t7 For deletion of "

"element at any position\n");

printf("\t8 To exit\n");

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

scanf("%d", &choice);

switch (choice) {

case 1:

traverse();

break;

case 2:

insertAtFront();

break;

case 3:

insertAtEnd();

break;

case 4:

insertAtPosition();

break;

case 5:

deleteFirst();

break;

case 6:

deleteEnd();

break;

case 7:

deletePosition();

break;

case 8:

exit(1);

break;

default:

printf("Incorrect Choice. Try Again \n");

continue;

}

}

return 0;

}

1. Write a program in C/C++/Java/Pythton to implement “stack” of integers using array.Take care of extreme conditions such as empty stack or full stack and display appropriate message to user.

#include<stdio.h>

#define MAX 5

int stack[MAX], item;

int ch, top = -1;

void push(int stack[], int item)

{

if (top == (MAX-1))

printf("\n\nStack is Overflow");

else

{

stack[++top] = item;

}

}

void pop(int stack[])

{

int ret;

if(top == -1)

printf("\n\nStack is Underflow");

else

{

ret = stack[top--];

printf("\nPopped element is %d", ret);

}

}

void palindrome(int stack[])

{

int i, count=0;

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

{

if(stack[i] == stack[top-i]) // or if(stack[i] == stack [top-i])

count++;

}

if(top==count)

printf("\nStack contents are Palindrome");

else

printf("\nStack contents are not palindrome");

}

void display(int stack[])

{

int i;

printf("\nThe stack contents are:");

if(top == -1)

printf("\nStack is Empty");

else{

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

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

printf("\n");

}

}

int main()

{

do{

printf("\n\n----MAIN MENU----\n");

printf("\n1. PUSH (Insert) in the Stack");

printf("\n2. POP (Delete) from the Stack");

printf("\n3. PALINDROME check using Stack");

printf("\n4. Exit (End the Execution)");

printf("\nEnter Your Choice: ");

scanf("%d", &ch);

switch(ch){

case 1: printf("\nEnter a element to be pushed: ");

scanf("%d", &item);

push(stack, item);

display(stack);

break;

case 2: pop(stack);

display(stack);

break;

case 3:

palindrome(stack);

break;

case 4:

exit(0); break;

default:

printf("\nEND OF EXECUTION");

}//end switch

}while (ch != 4);

return 0;

}

1. Write a program in C/C++/Java/Pythton to implement “Queue” of Strings using Singly Linked List. Take care of extreme conditions such as empty Queue or full Queue and display appropriate message to user.

#include <stdio.h>

#include <stdlib.h>

struct Node

{

str data; // integer data

struct Node\* next; // pointer to the next node

}\*rear = NULL, \*front = NULL;

struct Node\* newNode(int item)

{

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

if (node != NULL)

{

node->data = item;

node->next = NULL;

return node;

}

else {

printf("\nHeap Overflow");

exit(EXIT\_FAILURE);

}

}

int dequeue() // delete at the beginning

{

if (front == NULL)

{

printf("\nQueue Underflow");

exit(EXIT\_FAILURE);

}

struct Node\* temp = front;

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

front = front->next;

if (front == NULL) {

rear = NULL;

}

int item = temp->data;

free(temp);

return item;

}

void enqueue(int item) // insertion at the end

{

struct Node\* node = newNode(item);

printf("Inserting %d\n", item);

if (front == NULL)

{

front = node;

rear = node;

}

else {

rear->next = node;

rear = node;

}

}

int peek()

{

if (front != NULL) {

return front->data;

}

else {

exit(EXIT\_FAILURE);

}

}

int isEmpty() {

return rear == NULL && front == NULL;

}

int main()

{

enqueue(1);

enqueue(2);

enqueue(3);

enqueue(4);

printf("The front element is %d\n", peek());

dequeue();

dequeue();

dequeue();

dequeue();

if (isEmpty()) {

printf("The queue is empty");

}

else {

printf("The queue is not empty");

}

return 0;

}

1. Write a program in C/C++/Java/Pythton to implement bubbel sort or insertion sort.

#include<stdio.h>

void print(int a[], int n) //function to print array elements

{

int i;

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

{

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

}

}

void bubble(int a[], int n) // function to implement bubble sort

{

int i, j, temp;

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

{

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

{

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

{

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

}

void main ()

{

int i, j,temp;

int a[5] = { 10, 35, 32, 13, 26};

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

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

print(a, n);

bubble(a, n);

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

print(a, n);

}

6.Write a program in C/C++/Java/Python to implement binary search algorithm.

#include<stdio.h>

main()

{

int a[10],i,n,beg,mid,end,key,flag=0;

printf("Enter value for N:");

scanf("%d",&n);

printf("Enter %d values [ascending Order]:",n);

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

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

printf("Enter key value:");

scanf("%d",&key);

beg=0;

end=n-1;

while(beg<=end)

{

mid=(beg+end)/2;

if(a[mid]==key)

{

flag=1;

break;

}

else if(a[mid]<key)

beg=mid+1;

else

end=mid-1;

}

if(flag==1)

printf("key value is found at pos %d",mid+1);

else

printf("key value not found");

}

7) Write down answers for following with reference to TREE given below

a. What is the height of this tree?

Ans: 4

b. Is this a binary tree? Yes or No

Ans: Yes

c. Is this a Binary Search Tree? (Yes or No)

Ans: Yes

d. When you use BFS to traverse this tree, how will output look like?

Ans: 84,41,96,24,50,98,13,37

e. When you use DFS to traverse this tree, how will output look like in case of

i. Pre-order traversal

Ans:84,41,24,13,37,50,96,98

ii. In-order traversal

Ans: 13,24,37,41,50,84,96,98

iii. post-order traversal

Ans:13,37,24,50,41,98,96,84

f. If you want to add “47” under which NODE it will get added and will it added at left child of that node or right child of that node

Ans: 50 left child

g. What is LEFT SKEWED or RIGHT SKEWED Binary Serrach Tree? What is the worst cast time complexity of searching an element in such BST?

Ans: Left skewed:84,41,24,13

Time complexity: O(n)