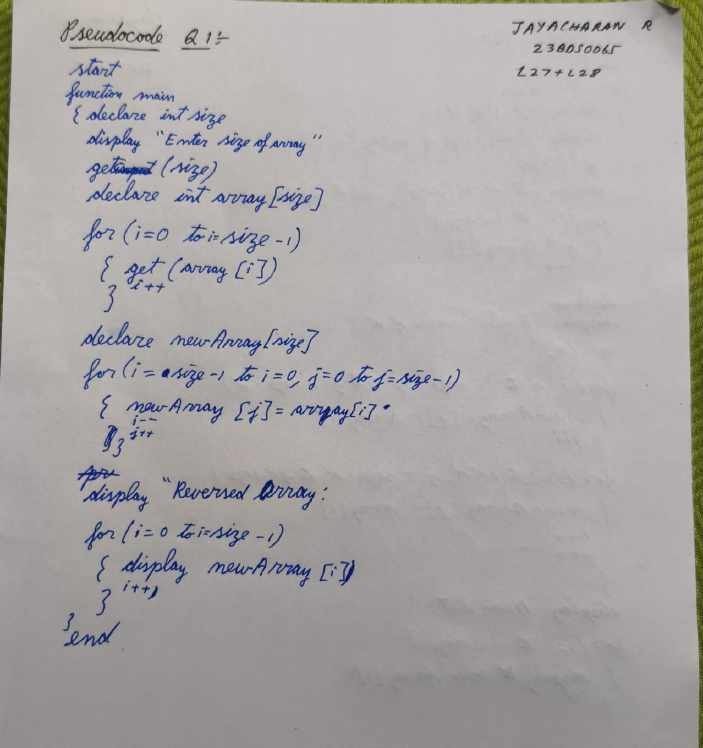
**DSA Lab Assignment**

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Q1. Write a C Program to reverse the given array. Objective: To reverse the given array.

Pseudocode: Code:

#include <stdio.h>

int main()

{

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); int size;

printf("Enter the size of the array:"); scanf("%d", &size);

int array[size];

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

{

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

}

int NewArray[size];

for (int i = size - 1, j = 0; i >= 0, j < size; i--, j++)

{

NewArray[j] = array[i];

}

printf("Reversed list: "); for (int i = 0; i < size; i++)

{

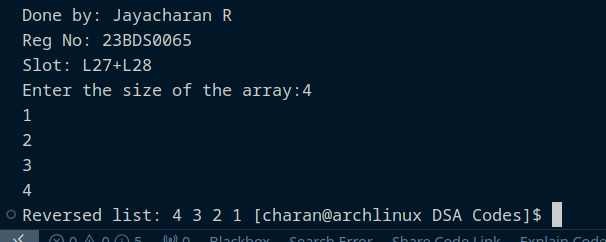
printf("%d ", NewArray[i]);

}

}

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Output:



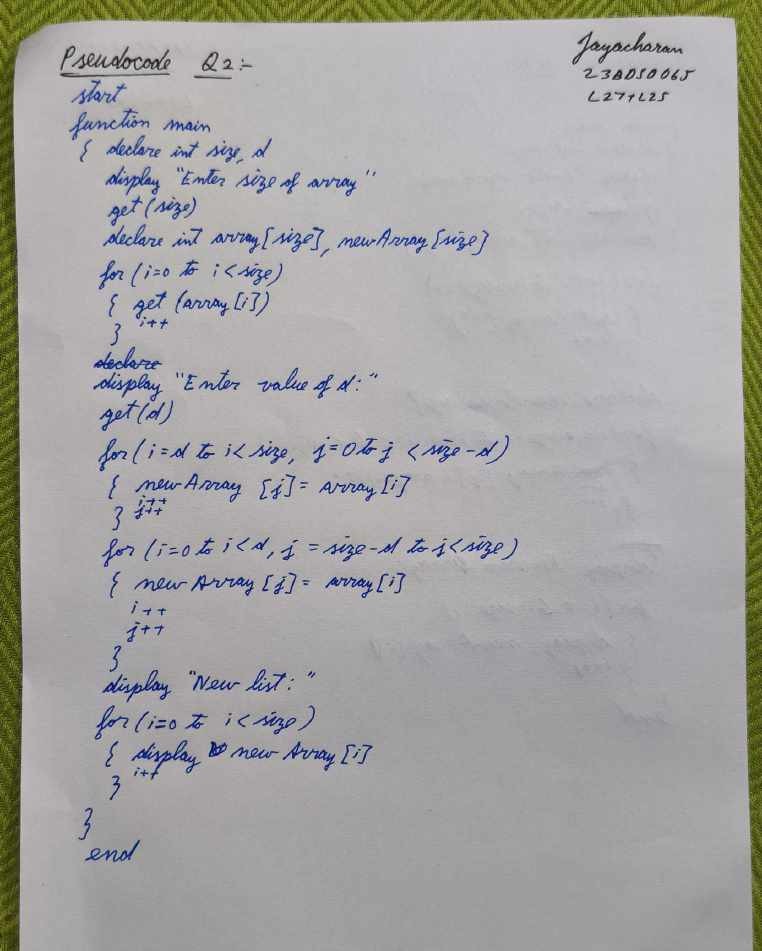
Q2. Write a C Program to swap for array rotation

Write a program to rotate (arr[], d, n) that rotates arr[] of size n by d elements. (Example d=2 , Shifting each elements by 2 location) Rotation of the above array by 2 will make array

Objective: To swap the given array for array rotation.

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Pseudocode:



# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Code:

#include <stdio.h>

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); int size;

printf("Enter the size of the array:"); scanf("%d",&size);

int array[size];

for(int i=0;i<size;i++){ scanf("%d",&array[i]);

}

int d;

printf("Enter the d value:"); scanf("%d",&d);

int NewArray[size];

for(int i=d,j=0;i<size,j<size-d;i++,j++){ NewArray[j]=array[i];

}

for(int i=0,j=size-d;i<d,j<size;i++,j++){ NewArray[j]=array[i];

}

printf("New list:"); for(int i=0;i<size;i++){

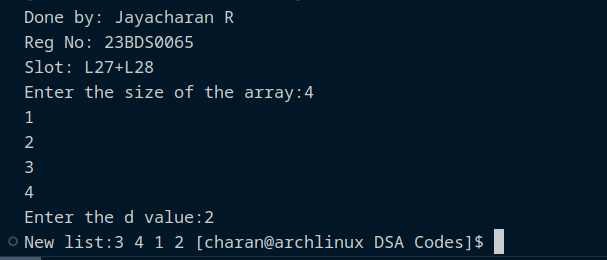
printf("%d ",NewArray[i]);

}

return 0;

}

Output:

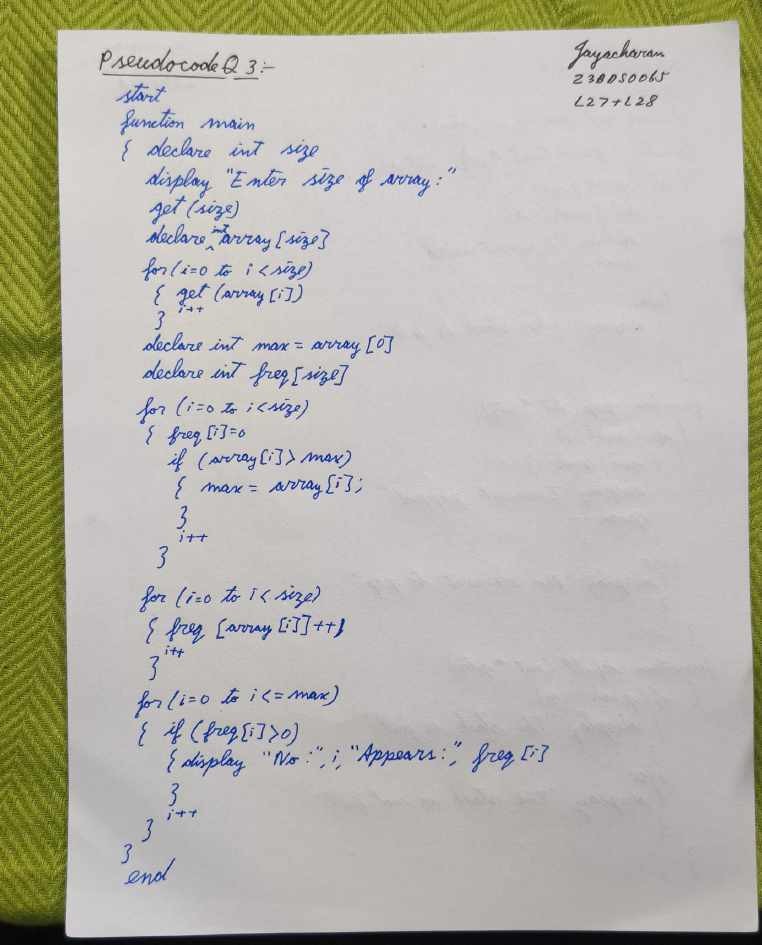


# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Q3. Write a C Program to Print All the Repeated Numbers with Frequency in an Array

Objective: To display frequency for repeated elements in an given array.

Pseudocode:



# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Code:

#include <stdio.h>

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); int size;

printf("Enter the size of the array:"); scanf("%d",&size);

int array[size];

for(int i=0;i<size;i++){ scanf("%d",&array[i]);

}

int max=array[0]; int feq[size];

for(int i=0;i<size;i++){ feq[i]=0;

if (array[i]>max){ max=array[i];

}

}

for(int i=0;i<size;i++){ feq[array[i]]++;

}

for(int i=0;i<=max;i++){ if (feq[i]>0){

printf("The No %d appears %d times \n",i,feq[i]);

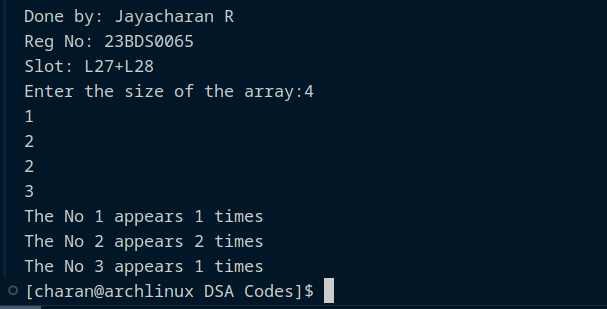
}

}

}

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Output:

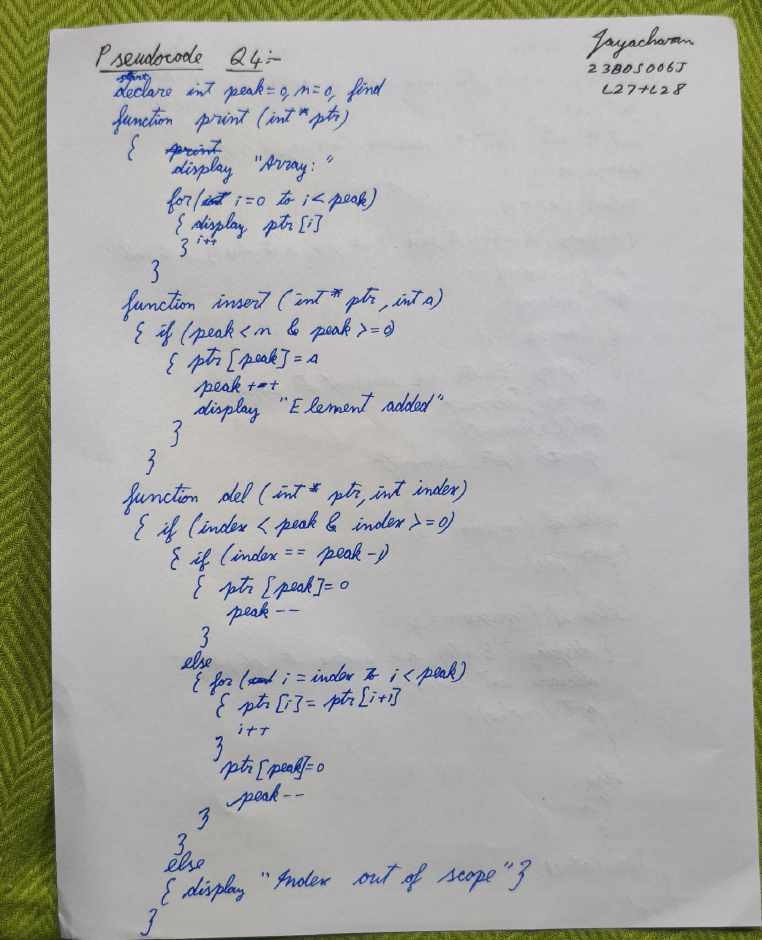
Q4.

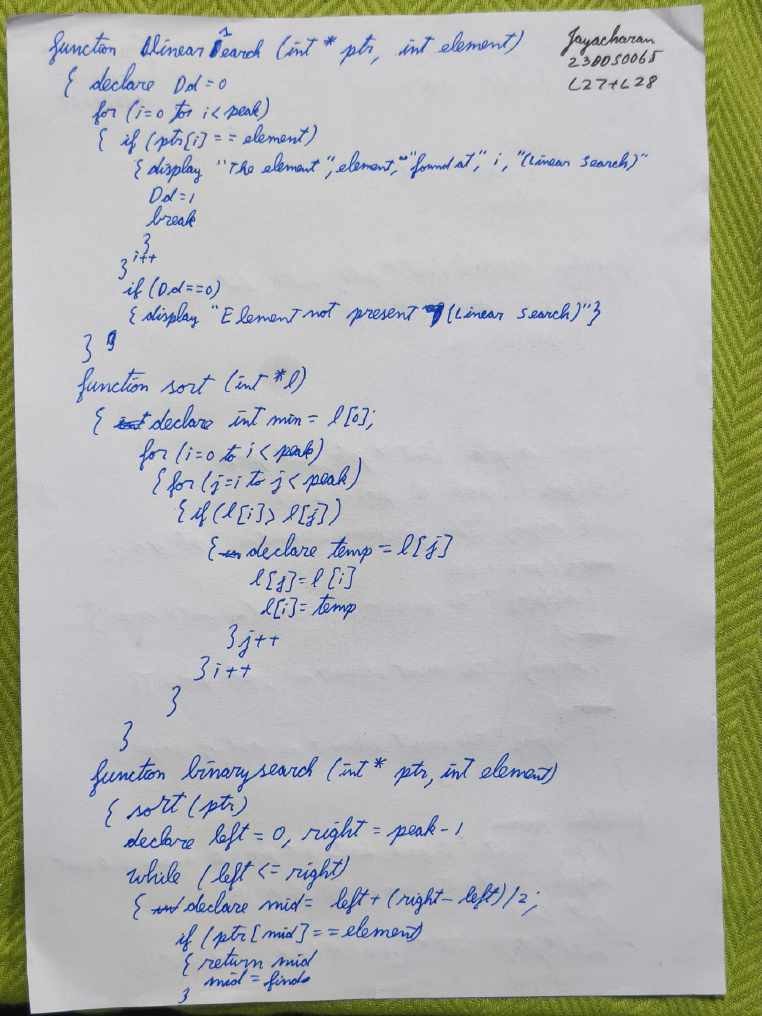
Write a C program to implement

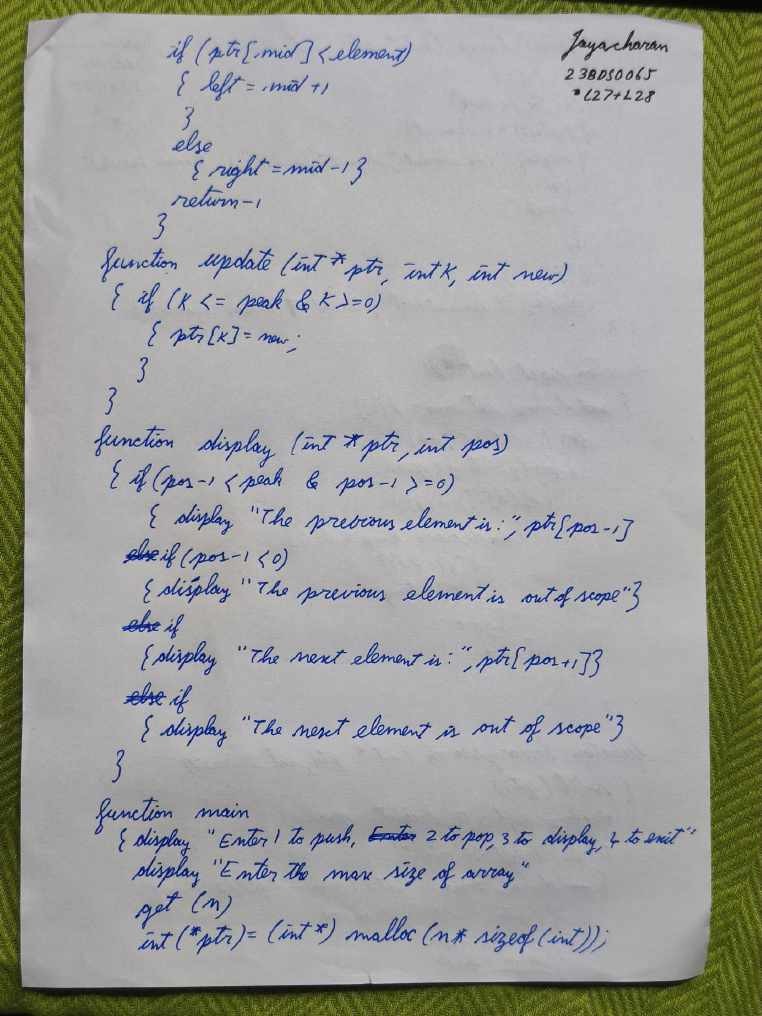
1. Insertion of an Element into the array (with position)
2. Delete an element from the array (With position)
3. Find or Search an Element in the given array.
   1. Binary search
   2. Linear search
4. Update the k th element in the given array (given position)
5. Display the next and previous element based on the position given.

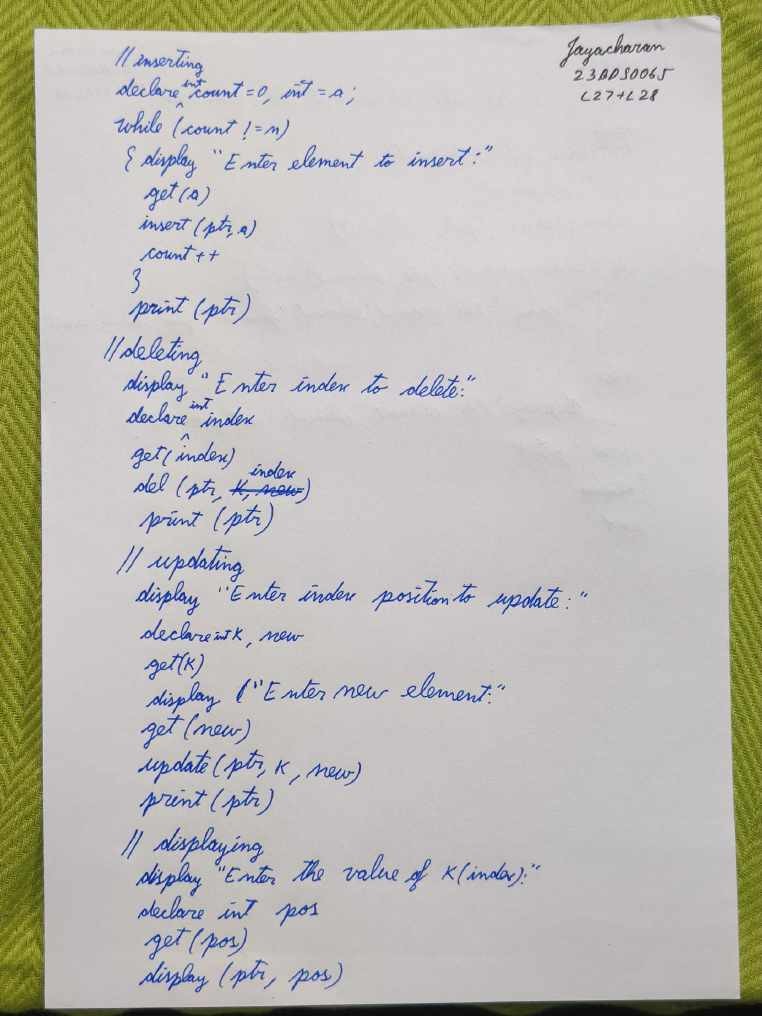
Objective: To implement insert, delete, find, binary search, linear search, update, next and previous element operations in a given array.

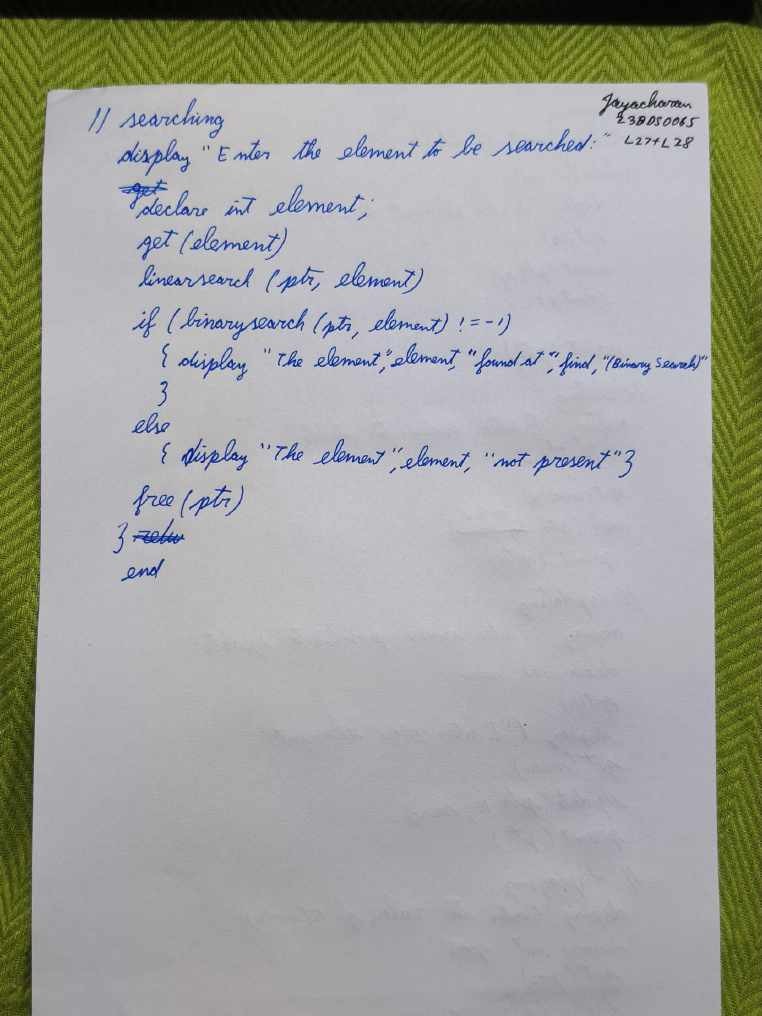
Pseudocode:











# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Code:

#include<stdio.h> #include<stdlib.h> int peak=0,n=0,find;

void print(int\* ptr){ printf("Array: "); for(int i=0;i<peak;i++){

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

}

printf("\n");

}

void insert(int\* ptr,int a){ if(peak<n & peak>=0){

ptr[peak]=a; peak+=1;

printf("Element added \n");

}

}

void del(int\* ptr,int index){ if(index<peak & index>=0){

if(index==peak-1){ ptr[peak]=0; peak-=1;

}

else{

for(int i=index;i<peak;i++){ ptr[i]=ptr[i+1];

}

ptr[peak]=0; peak-=1;

}

}

else{

printf("index out of scope \n");

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

}

void LinearSearch(int\* ptr,int element){ int Dd=0;

for(int i=0;i<peak;i++){ if(ptr[i]==element){

printf("The Element %d found at %d (LINEAR SEARCH) \n",element,i); Dd=1;

break;

}

}

if(Dd==0){

printf("The Element not present (LINEAR SEARCH) \n");

}

}

int sort(int\* l){ int min=l[0];

for(int i=0;i<peak;i++){ for(int j=i;j<peak;j++){

if(l[i]>l[j]){

int temp=l[j]; l[j]=l[i]; l[i]=temp;

}

}

}

}

int BinarySearch(int\* ptr,int element){ sort(ptr);

int left = 0;

int right = peak - 1;

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

while (left <= right) {

int mid = left + (right - left) / 2; if (ptr[mid] == element) {

return mid; mid=find;

}

if (ptr[mid] < element) { left = mid + 1;

}

else {

right = mid - 1;

}

}

return -1;

}

void update(int\* ptr,int k,int new){ if(k<=peak & k>=0){

ptr[k]=new;

}

}

void display(int\* ptr,int pos){ if(pos-1<peak & pos-1>=0){

printf("The previous element is: %d \n",ptr[pos-1]);

}

if(pos-1<0){

printf("The previous element is out of scope \n");

}

if(pos+1<peak){

printf("The next element is: %d \n",ptr[pos+1]);

}

if(pos+1>=peak){

printf("The next element is out of scope \n");

}

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); printf("Enter 1 to push \n2 to pop \n3 to display \n4 to EXIT \n"); printf("Enter the max size of the array:");

scanf("%d",&n);

int\* ptr=(int \*)malloc(n\*sizeof(int));

//inserting element int count=0;

int a; while(count!=n)

{printf("Enter element to be entered:"); scanf("%d",&a);

insert(ptr,a); count++;

}

print(ptr);

//deleting element printf("Enter index to del:"); int index; scanf("%d",&index); del(ptr,index);

print(ptr);

//updating

printf("Enter the index position to update:"); int k,new;

scanf("%d",&k);

printf("Enter the new element:"); scanf("%d",&new); update(ptr,k,new);

print(ptr);

//display

printf("Enter the value of k(index):"); int pos;

scanf("%d",&pos); display(ptr,pos);

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

//searching int element;

printf("Enter the element to be searched:"); scanf("%d",&element);

//linear search LinearSearch(ptr,element);

//binary search

if (BinarySearch(ptr,element) != -1){

printf("The Element %d found at %d (BINARY SEARCH)",element,find);

}

else{

printf("The Element %d not present (BINARY SEARCH)",element);

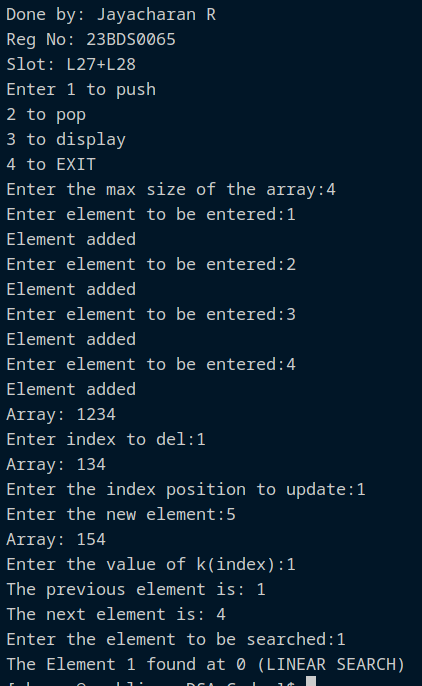
}

free(ptr); return 0;

}

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Output:



# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

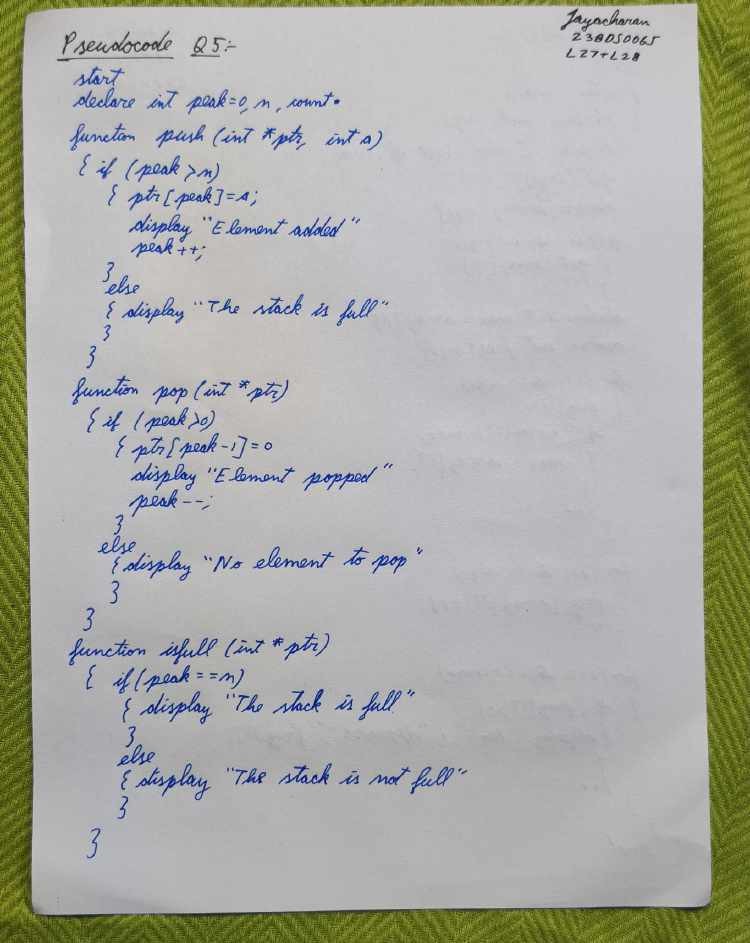
Q5. Write C program to implement Stack Using an array with the operation :

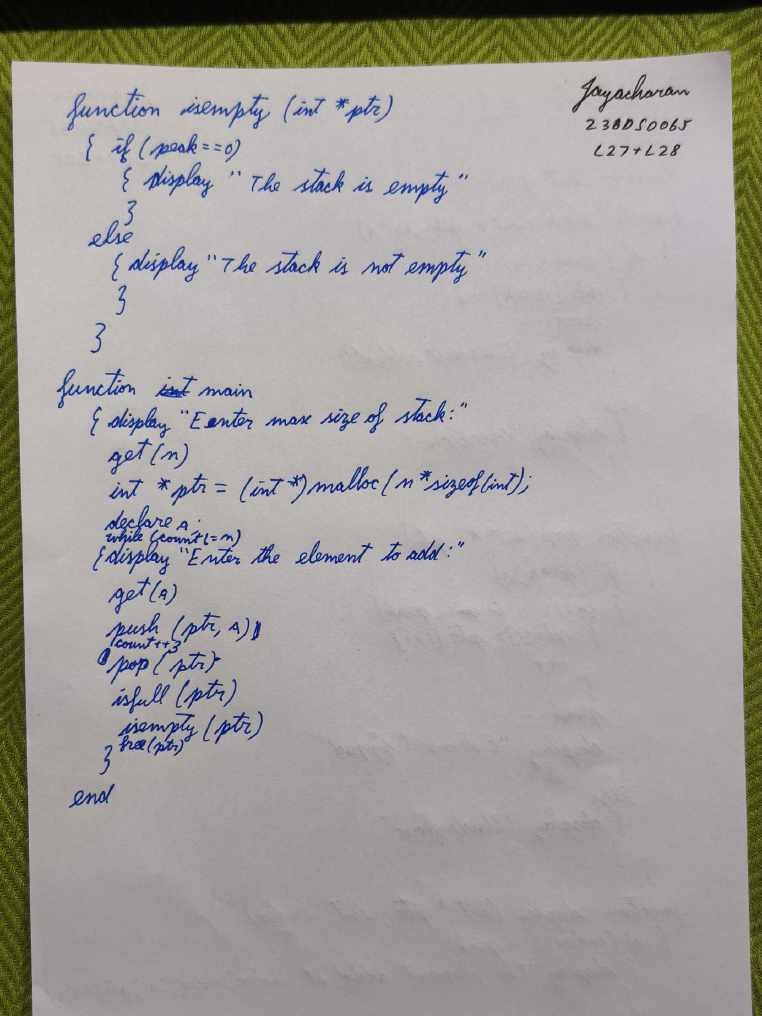
* 1. Push
  2. Pop
  3. isEmpty()
  4. isFull()

Objective: To implement stack using operations of push, pop, isempty, isfull operations in a given array.

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Pseudocode:





# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Code:

#include<stdio.h> #include<stdlib.h> int peak=0,n;

void push(int\* ptr,int a){ if(peak<n){

ptr[peak]=a; printf("Element added\n"); peak+=1;

}

else{

printf("The stack is full\n");

}

}

void pop(int\* ptr){ if(peak>0){

ptr[peak-1]=0; printf("Element poped\n"); peak-=1;

}

else{

printf("No element to pop\n");

}

}

void IsFull(int\* ptr){ if(peak==n){

printf("The stack is full\n");

}

else{

printf("The stack is not Full\n");

}

}

void IsEmpty(int\* ptr){ if(peak==0){

printf("The stack is empty\n");

}

else{

printf("The stack is not empty\n");

}

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); printf("Enter the max size of the stack:\n");

scanf("%d",&n);

int\* ptr=(int \*)malloc(n\*sizeof(int));

//push

int count=0; int a;

while (count!=n)

{printf("Enter the element to add: \n"); scanf("%d",&a);

push(ptr,a); count++;

}

//pop pop(ptr);

//IsFull IsFull(ptr);

//IsEmpty IsEmpty(ptr);

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

{

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

}

free(ptr);

}

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Output:

# Done By:

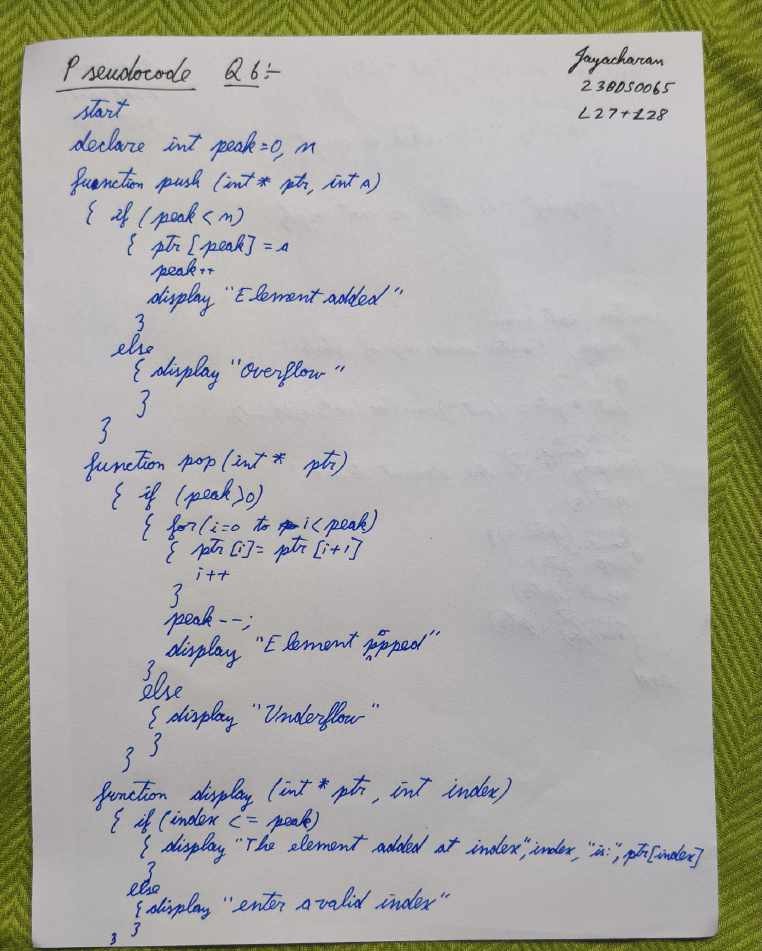
**Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28**

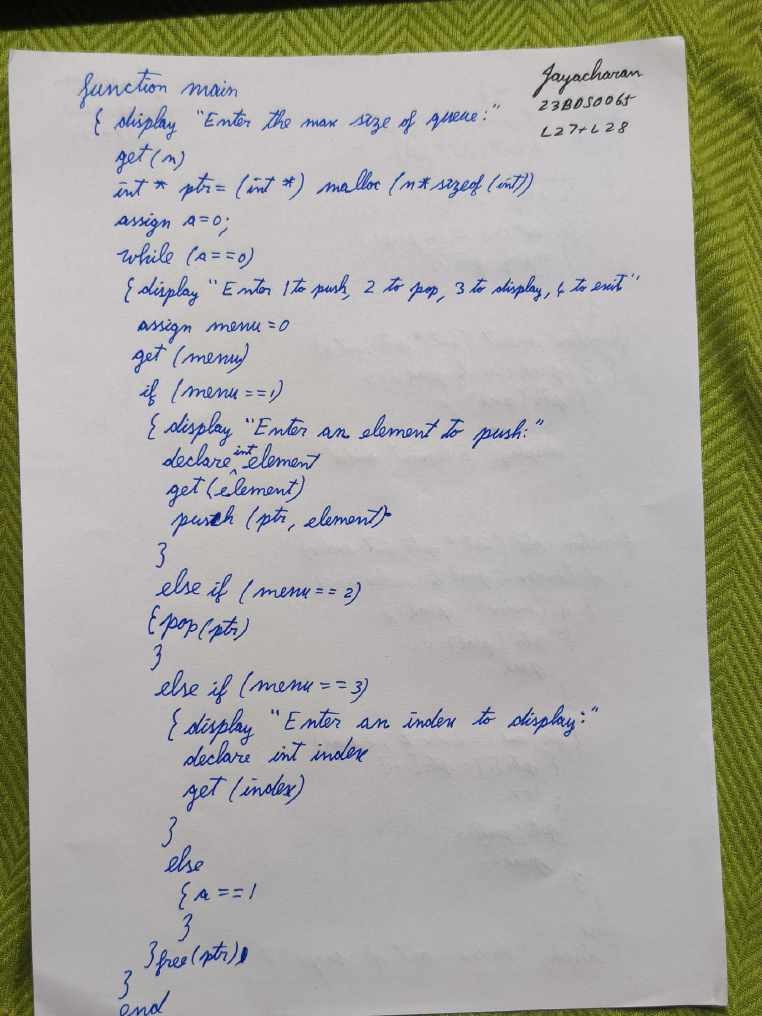
Q6. Write a C Program to implement Queue with the following operation:(Design a Menu driven program)

1. Inserting and Element
2. Deleting an Element
3. Display the Element
4. Exit

Objective: To design a menu driven C Program to implement queue with insert, delete, display and exit operations.

Pseudocode:





# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Code: #include<stdio.h> #include<stdlib.h> int peak=0,n;

void push(int\* ptr,int a){ if(peak<n){

ptr[peak]=a; peak+=1;

printf("Element added \n");

}

else{

printf("Over Flow \n");

}

}

void pop(int\* ptr){ if(peak>0){

for(int i=0;i<peak;i++){ ptr[i]=ptr[i+1];

}

peak-=1;

printf("Element poped \n");

}

else{

printf("Under Flow \n");

}

}

void display(int\* ptr,int index){ if(index<=peak){

printf("The element at index %d is: %d \n",index,ptr[index]);

}

else{

printf("enter valid index \n");

}

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); printf("Enter the max size of the Queue:");

scanf("%d",&n);

int\* ptr=(int \*)malloc(n\*sizeof(int)); int a=0;

while (a==0)

{

printf("Enter 1 to push \n2 to pop \n3 to display \n4 to EXIT \n"); int menu=0;

scanf("%d",&menu); if(menu==1){

printf("Enter a element to push:"); int element; scanf("%d",&element); push(ptr,element);

}

else if(menu==2){ pop(ptr);

}

else if(menu==3){

printf("Enter a index to display:"); int index;

scanf("%d",&index); display(ptr,index);

}

else if(menu==4){ a=1;

}

}

free(ptr);

}

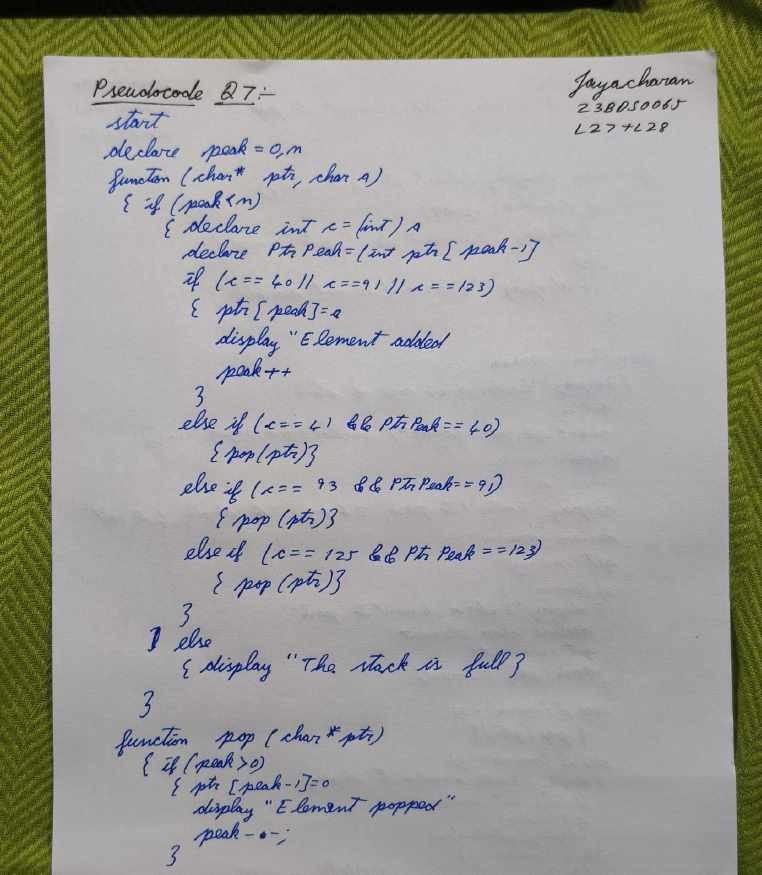
Output:

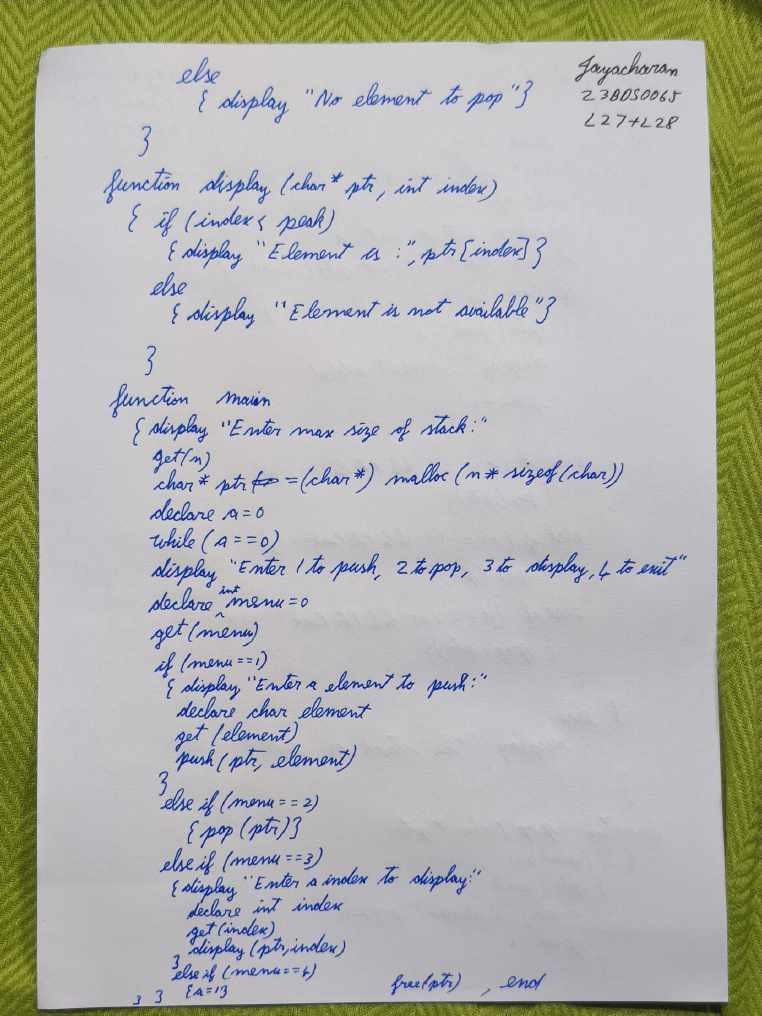
# 23BDS0065, Slot: L27+L28

**Done By: Jayacharan R, Reg no:**

Q7. Write a C program to implement balancing the parenthesis using Stack. Note: Push all types of open parenthesis into the stack and pop whenever a close parenthesis encountered.

Objective: To balance the parenthesis using stack. Pseudocode:





Code: #include<stdio.h> #include<stdlib.h> int peak=0,n;

void pop(char\* ptr);

void push(char\* ptr,char a){ if(peak<n){

int c=(int)a;

int PtrPeak=(int)ptr[peak-1]; if(c==40 || c==91 || c==123){

ptr[peak]=a; printf("Element added \n"); peak+=1;

}

else if(c==41 && PtrPeak==40){ pop(ptr);

}

else if(c==93 && PtrPeak==91){ pop(ptr);

}

else if(c==125 && PtrPeak==123){ pop(ptr);

}

}

else{

printf("The stack is full \n");

}

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

void pop(char\* ptr){ if(peak>0){

ptr[peak-1]=0; printf("Element poped \n"); peak-=1;

}

else{

printf("No element to pop \n");

}

}

void display(char\* ptr,int index){ if(index<peak){

printf("The element is %c \n",ptr[index]);

}

else{

printf("Element not avaliable \n");

}

}

# //Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

int main(){

printf("Done by: Jayacharan R\nReg No: 23BDS0065\nSlot: L27+L28\n"); printf("Enter the max size of the stack:");

scanf("%d",&n);

char\* ptr=(char \*)malloc(n\*sizeof(char));

int a=0; while (a==0)

{

printf("Enter 1 to push \n2 to pop \n3 to display \n4 to EXIT \n"); int menu=0;

scanf("%d",&menu);

if(menu==1){

printf("Enter a element to push:"); char element; scanf("%s",&element); push(ptr,element);

}

else if(menu==2){ pop(ptr);

}

else if(menu==3){

printf("Enter a index to display:"); int index;

scanf("%d",&index); display(ptr,index);

}

else if(menu==4){ a=1;

}

}

free(ptr);

}

# Done By: Jayacharan R, Reg no: 23BDS0065, Slot: L27+L28

Output:

