```c
#include <stdio.h>

int main() {
    int max;
    printf("enter the max size of stack");
    scanf("%d",&max);
    int stack[max];
    int top=-1;
    int value;
    int opt=0;

    while(opt!=7)
    {
    printf("enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to
check IfFull and 7 to exit");
    scanf("%d",&opt);

    if (opt==1)
    {
        if (top==max-1)
        {
            printf("stack overflow\n");
        }
        else
        {
            printf("enter value to be pushed");
            scanf("%d",&value);
            top=top+1;
            stack[top]=value;
        }
    }
    else if (opt==2)
    {
        if (top==-1)
        {
            printf("stack underflow\n");
        }
        else
        {
            value= stack[top];
            top=top-1;
            printf("%d was popped\n",value);
        }

    }
```

```
    else if(opt==3)
    {
        if(top==-1)
        {
            printf("stack is empty\n");
        }
        else
        {
            value = stack[top];
            printf("top- most element is %d \n",value);
        }
    }
    else if (opt==4)
    {
        if(top==-1)
        {
            printf("stack is empty\n");
        }
        else
        {
            printf("stack is not empty\n");
        }
    }
    else if (opt==5)
    {
         if (top==max-1)
        {
            printf("stack is full\n");
        }
        else
        printf("stack is not full\n");
    }
    else if(opt==6)
    {
        for(int i=0;i<=top;i++)
        {
            printf("%d,",stack[i]);
        }
        printf("\n");
    }
    }
}
```

Output:

```
sammj@SAM_LAPTOP MINGW64 ~/DS LAB
$  /usr/bin/env c:\\Users\\sammj\\.vscode\\extensions\\ms-vscode.cpptools-1.22.11-win32-x64\\debugAd
soft-MIEngine-In-pbspe0bl.mwj --stdout=Microsoft-MIEngine-Out-1nlrwtnl.r2n --stderr=Microsoft-MIEngi
v4vwmg1f.b2h --dbgExe=C:\\msys64\\ucrt64\\bin\\gdb.exe --interpreter=mi
enter the max size of stack5
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
enter value to be pushed1
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
enter value to be pushed1
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
enter value to be pushed2
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
enter value to be pushed3
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
enter value to be pushed4
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
stack overflow
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit5
stack is full
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit1
stack overflow
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit4
stack is not empty
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit5
stack is full
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit2
4 was popped
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit2
3 was popped
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit2
2 was popped
enter 1 to PUSH, 2 to POP, 3 to SEEK, 4 to check IfEmpty and 5 to check IfFull and 7 to exit
```

PAGE :
DATE : / /

C program to implement stack operations

```
#include <stdio.h>
void main()
{ int v
  int max_size;
  printf("Enter the maximum size of the array");
  scanf("%d", &max_size);
  int stack[max_size];
  int top = -1;
  while(opt!=6)
  { printf("Enter 1 to push, 2 to POP, 3 to seek, 4 to check if empty, and 5 to check if full and 6 to exit");
  int opt;
  scanf("%d", &opt);
  if(opt==1)
  {
      if(top == max_size-1)
      {
          printf("stack overflow");
      }
      else
      {
          printf("Enter element: ");
          scanf("%d", &v);
          top = top+1;
          stack[top] = v;
      }
}
```

```
else if (opt == 2)
{
    if (stack.top == -1)
    {
        printf("Stack Overflow\n");
    }
    else
    {
        value = stack[top];
        v = a.stack[top];
        top = top - 1;
    }   printf("%d was popped\n", &v);
}

else if (opt == 3)
{
    if (top == -1)
    {
        printf("Stack is empty\n");
    }
    else
    {
        printf f
        v = stack[top];
        printf("%d\n", v);
    }
}
else if (opt == 4)
{
    if (st.stack top == -1)
        printf("empty\n");
    else
        printf("Not empty\n");
}
```

```
else if (opt==5)
{
    if (top == max_size-1)
        printf("Full\n");
    else
        printf("Not full\n");
}
else
{
    printf("Invalid Input\n")
}
}
```

Seen

O/p: Output

Enter the max size of stack 2
enter 1 to PUSH, 2 to POP, 3 to seek, 4 to check if empty
5 to check if full and 7 to exit : 1
enter value to be pushed : 1
Enter 1 to PUSH, 2 to POP, 3 to seek ... : 2
1 was popped
Enter 1 to PUSH, 2 to POP ... : 4
Stack is empty
Enter 1 to PUSH, 2 to POP ... : 2
Stack underflow
Enter 1 to PUSH, 2 to POP ... : 1
Enter value to be pushed : 1
Enter 1 to PUSH, 2 to POP ... : 1
Enter value to be pushe : 2
enter 1 to PUSH, 2 to POP ... : 1
Enter value to be pushed : 3
Stack overflow
Enter 1 to Push, 2 to POP ... : 6
1, 2,