Lab 1

1. Write a program to simulate the working of stack using an array with the following:
a) Push
b) Pop
c) Display. The program should print appropriate messages for stack overflow, stack underflow.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#define max 3
int st[max], top=-1;
void push(int st[], int val);
int pop(int st[]);
int peek(int st[]);
void display(int st[]);
int main(int argc, char *argv[])
{
  int val, option;
  do
  {
   printf("\n***** MAIN MENU *****");
   printf("\n 1. PUSH");
   printf("\n 2. POP");
   printf("\n 3. PEEK");
   printf("\n 4. DISPLAY");
   printf("\n 5. EXIT");
```

DATE: / /

```
printf("\n Enter your option:");
scanf("%d", &option);
switch(option)
{
 case 1:
        printf("\n Enter the number
         to be pushed on stack:");
        scanf("%d", &val);
        push(st, val);
        break;
case 2:
        val=pop(st);
        if(val!=-1)
        printf("\n The value deleted
                from stack is:%d", val);
        break;
case 3:
        val = peek(st);
        if(val!=-1)
         printf("\n The value stored
                at top of stack is:%d", val);
        break;
case 4:
        display(st);
        break;
}
}while(option!=5);
return 0;
}
void push(int st[], int val)
{
```

```
if(top==max-1)
  printf("\n STACK OVERFLOW");
else
{
 top++;
 st[top]=val;
}
}
int pop(int st[])
{
 int val;
 if(top==-1)
 {
  printf("\n STACK UNDERFLOW");
  return -1;
 }
 else
 {
  val=st[top];
  top--;
  return val;
 }
}
void display(int st[])
{
 int i;
 if(top==-1)
  printf("\n STACK IS EMPTY");
 else
 {
 for(i=top; i>=0; i--)
  printf("\n %d", st[i]);
```

```
        printf("\n");
    }
}
int peek(int st[])
{
  if(top == -1)
  {
    printf("\n STACK IS EMPTY");
    return -1;
  }
  else
    return (st[top]);
  }
```

Output:

```
**** MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 2

STACK UNDERFLOW
**** MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 3
```

STACK IS EMPTY

*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1

Enter the number to be pushed on stack: 4

*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1

Enter the number to be pushed on stack: 6

*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1

Enter the number to be pushed on stack: 8

PAGE NO.:
DATE: / /

*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1

Enter the number to be pushed on stack: 7

STACK OVERFLOW
*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 3
The value stored at top of stack is: 8

*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 4
8
6
4

***** MAIN MENU *****

1. PUSH

2. POP

3. PEEK

4. DISPLAY

5. EXIT

Enter your option : 5

DATE: / /

Programs on Dynamic Memory allocation

1. Program to calculate the sum of n numbers entered by the user using malloc and free

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int n, i, *ptr, sum=0;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    ptr = (int *) malloc (n * sizeof(int));
    // if memory cannot be allocated
    if (ptr==NULL)
    {
    printf("Error! memory not allocated.");
    exit(0);
    }
    printf("Enter elements: ");
    for (i=0 ; i<n ; ++i)
    {
    scanf("%d", ptr+i);
    sum+= *(ptr+i);
    }
    printf("Sum = %d", sum);
    //deallocating the memory
    free(ptr);
    return 0;
}
```

Output:
Enter number of elements: 3
Enter elements: 12 34 56
Sum = 102

2. Program to calculate the sum of n numbers entered by the user using calloc and free.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int n, i, *ptr, sum=0;
  printf("Enter number of elements: ");
  scanf("%d", &n);
  ptr = (int*) calloc (n, sizeof(int));
  if(ptr==NULL)
  {
    printf("Error! memory not allocated.");
    exit(0);
  }
  printf("Enter elements: ");
  for(i=0; i<n; ++i)
  {
    scanf("%d", ptr+i);
    sum += *(ptr+i);
  }
  printf("Sum = %d", sum);
  free(ptr);
  return 0;
}
```

Output:
Enter number of elements: 4
Enter elements: 24 23 28 20
Sum = 95

3. Program to demonstrate dynamic memory allocation using realloc

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
int *ptr, i, n1, n2;
printf("Enter size: ");
scanf("%d", &n1);
ptr=(int*) malloc (n1 * sizeof(int));
printf("Addresses of previously
        allocated memory: \n");
for (i=0 ; i<n1; ++i)
  printf ("%pc\n", ptr+i);
printf("\nEnter the new size: ");
scanf ("%d", &n2);
// reallocating the memory
ptr = realloc (ptr, n2 * sizeof(int));
printf("Addresses of newly allocated
        memory: \n");
for (i=0 ; i<n2 ; ++i)
  printf ("%pc\n", ptr+i);

free (ptr);
return 0;
}
```

Enter size: 2
Addresses of previously allocated memory:
0000000009914300
0000000009914340

Enter the new size: 3
Addresses of newly allocated memory:
0000000009914300
0000000009914340
0000000009914380