Practical 1: Introduction to pointers. Implement Call by value and Call by Reference.

A pointer is defined as a derived data type that can store the address of other C variables or a memory location. We can access and manipulate the data stored in that memory location using pointers

Code Snippet:

```
//call by value
#include <stdio.h>
void swap(int, int);
int main()
{
   printf("Name- Rupesh kumar \n Enrollment No. 92201703182\n");
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n", a, b);
   swap(a, b);
   printf("After swapping values in main a = %d, b = %d\n", a, b);
}
void swap(int a, int b)
{
   int temp;
   temp = a;
   a = b;
   b = temp;
   printf("After swapping values in function a = %d, b = %d\n", a, b);
}
```

```
//call by reference
#include <stdio.h>
void swap(int *, int *);
int main()
{
   printf("Name- Rupesh kumar \n Enrollment No. 92201703182\n");
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n", a, b);
   swap(&a, &b);
   printf("After swapping values in main a = %d, b = %d\n", a, b);
}
void swap(int *a, int *b)
{
   int temp;
   temp = *a;
   *a = *b;
   *b = temp;
   printf("After swapping values in function a = %d, b = %d\n", *a, *b);
```

```
}
```

Output:

1. Call by Value

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output>
Name- Rupesh kumar
 Enrollment No. 92201703182
Before swapping the values in main a = 10, b = 20
After swapping values in function a = 20, b = 10
After swapping values in main a = 10, b = 20
```

2. Call by Reference

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output> &
Name- Rupesh kumar
 Enrollment No. 92201703182
Before swapping the values in main a = 10, b = 20
After swapping values in function a = 20, b = 10
After swapping values in main a = 20, b = 10
```

Practical 2: Introduction to Dynamic Memory Allocation and use of DMA functions malloc(), calloc(), free(), etc.

Code Snippet:

1.Calloc

```
#include<stdio.h>
#include<stdlib.h>
int main(){
        printf("Name=Rupesh kumar\nEnrollment no.=92201703182\n");
        int n,i,*ptr;
        printf("enter the size of data \n");
        scanf("%d",&n);
        ptr=(int*)calloc(n,sizeof(int));
        printf("enter data \n");
        for(i=0;i<n;i++){
                scanf("%d",ptr+i);
        }
        printf("Your data is : \n");
        for(i=0;i<n;i++){
                printf("%d",*(ptr+i));
        }
        return 0;
}
```

Output:

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output> &
Name=Rupesh kumar
Enrollment no.=92201703182
enter the size of data
4
enter data
2
5
8
9
Your data is :
2       5       8       9
```

2.Malloc

```
#include<stdio.h>
#include<stdlib.h>
int main(){
        printf("Name=Rupesh kumar\nEnrollment no.=92201703182\n");
        int n,i,*ptr;
        printf("enter the size of data \n");
        scanf("%d",&n);
        ptr=(int*)malloc(n*sizeof(int));
        printf("enter data \n");
```

```
        for(i=0;i<n;i++){
                scanf("%d",ptr+i);
        }
        printf("Your data is : \n");
        for(i=0;i<n;i++){
                printf("%d",*(ptr+i));
        }
        return 0;
}
```

Output:

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output>
Name=Rupesh kumar
Enrollment no.=92201703182
enter the size of data
4
enter data
5
8
7
9
Your data is :
5       8       7       9
```

3.Realloc()

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
   printf("Name- Rupesh kumar \n Enrollment No. 92201703182\n");
   int *ptr = (int *)malloc(3 * sizeof(int));
   ptr[0] = 1;
   ptr[1] = 2;
   ptr[2] = 3;

   // resize the memory block to hold 5 integers
   ptr = (int *)realloc(ptr, 5 * sizeof(int));
   ptr[3] = 4;
   ptr[4] = 5;
   for (int i = 0; i < 5; i++)
   {
      printf("%d\n ", ptr[i]);
   }
   free(ptr);
   return 0;
}
```

Output:

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output>
Name- Rupesh kumar
 Enrollment No. 92201703182
1
 2
 3
 4
 5
 memory get free
```

4.Free

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int *ptr;
   int n = 5;
    printf("Name=Rupesh kumar\nEnrollment no.=92201703182\n");
   printf("Enter number of Elements: %d\n", n);

   scanf("%d", &n);

   ptr = (int *)calloc(n, sizeof(int));

   if (ptr == NULL)
   {
      printf("Memory not allocated \n");
      exit(0);
   }
   printf("Successfully allocated the memory using "
         "calloc(). \n");
   free(ptr);

   printf("Calloc Memory Successfully freed.");

   return 0;
}
```

Output:

```
PS C:\Users\rupes\Desktop\desktop1\facereconition\output> & .\'thir
Name=Rupesh kumar
Enrollment no.=92201703182
Enter number of Elements:
5
Successfully allocated the memory using calloc().
Calloc Memory Successfully freed.
```

Practical 3: Write a program to implement STACK using array that performs following operations: (a) PUSH (b) POP (c) PEEP (d) CHANGE (e) DISPLAY

```
#include<stdio.h>
#define MAX 5
int S[MAX], top=-1;
int isFull(){
   if(top==MAX-1)
      return 1;
   else
      return 0;
}
void push(int x)
{
   if(isFull())
      printf("Stack is OverFlow.");
   else{
      top++;
      S[top]=x;
      printf("Value pushed successfully.");
   }
}
int isEmpty(){
   if(top==-1)
      return 1;
   else
      return 0;
}
void pop(){
   if(isEmpty())
```

```
        printf("\nStack is Underflow.");
    else{
        printf("%d is deleted.",S[top]);
        top--;
    }
}
void display(){
    int i;
    if(isEmpty())
        printf("Stack is Empty.");
    else{
        printf("Stack is: ");
        for(i=0;i<=top;i++)
            printf("%d ",S[i]);
    }
}
void peep(){
   if(isEmpty())
        printf("Stack is Empty.");
    else
        printf("Topmost element is %d.",S[top]);
}
void change(){
    int index, value;
    printf("Enter an Index:");
    scanf("%d",&index);
    printf("Enter a value:");
    scanf("%d",&value);
    if(top-index+1 < 0)
        printf("Invalid Index.");
```

```
    else{
        S[top-index+1] = value;
        printf("Value changed Successfully.");
    }
}
void main()
{
    int choice,v;
    while(1)
    {
        printf("\n\nStack Operations:");
        printf("\n1. Push\t2. Pop\t3. Peep\t4. Change");
        printf("\t5. Display\t6. isEmpty\t7. isFull\t 8. Exit");
        printf("\nEnter your choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("Enter a Value:");
                    scanf("%d",&v);
                    push(v);
                    break;
            case 2: pop(); break;
            case 3: peep(); break;
            case 4: change(); break;
            case 5: display(); break;
            case 6: if(isEmpty())
                        printf("Yes, Stack is Empty.");
                    else
                        printf("No, Stack is Not Empty.");
                        break;
```

```
        case 7:
            if(isFull())
                printf("Yes, Stack is Full.");
            else
                printf("No, Stack is Not Full.");
                break;
        case 8: exit(0);
        default: printf("\nInvalid Choice!");
    }
  }
}
```

Output: :

```
PS C:\Users\rupes\Desktop\desktop1\facereconition> cd 'c:\Users\rupes\Desktop\desktop1\facereconition
PS C:\Users\rupes\Desktop\desktop1\facereconition\output> & .\'thirdpracticalds.exe'


Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:1
Enter a Value:10
Value pushed successfully.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:1
Enter a Value:10
Value pushed successfully.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:2
10 is deleted.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:3
Topmost element is 10.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:4
Enter an Index:0
Enter a value:20
Value changed Successfully.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:5
Stack is: 10

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:6
No, Stack is Not Empty.

Stack Operations:
1. Push 2. Pop  3. Peep 4. Change      5. Display     6. isEmpty     7. isFull       8. Exit
Enter your choice:7
No, Stack is Not Full.
```