21/12/23

> write a program with a function to swap two numbers using pointers.

->

```
#include <stdio.h>
int main() {
int a, b, temp;
int *ptr1, *ptr2;
printf("enter the value of a and b: ");
scanf("%d %d", &a, &b);
printf("\n before swapping the numbers a=%d and
                                b=%d", a, b);

ptr1 = &a;
ptr2 = &b;
temp = *ptr1;
*ptr1 = *ptr2;
*ptr2 = temp;
printf("\n after swapping a=%d and b=%d", a, b);
return 0;
}
```

output:

enter the value of a and b: 2 3
before swapping the numbers a=2 and b=3
after swapping a=3 and b=2

2) Write a program to implement dynamic memory allocation functions like malloc, calloc, free, realloc

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
int *ptr, *ptr1;
int n, i;

n = 5;
printf("Enter number of elements : %d \n", n);
ptr = (int *) malloc (n * sizeof(int));
ptr1 = (int *) calloc (n, sizeof(int));
if (ptr == NULL || ptr1 == NULL) {
        printf("Memory not allocated. \n");
        exit(0);
}
else {
   printf ("Memory Successfully allocated using
    malloc. \n");

free (ptr);
printf("Malloc Memory Successfully freed. \n");
print("\n Memory Successfully allocated using
      Calloc. \n");
free (ptr1);
print f("Calloc Memory Successfully freed. \n");
}
return 0;
}
```

realloc :-

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

int main() {
  int *pts, i, n1, n2;
  printf("Enter size:");
  scanf("%d", &n1);
  pts = (int *) malloc (n1 * sizeof(int));
  printf("Addresses of previously allocated memory:\n");
  for (i = 0; i<n1; ++i)
    printf("%pc\n", pts + i);
  printf("\n Enter the new size:");
  scanf("%d", &n2);
  pts = realloc(pts, n2 * sizeof(int));
  printf("Addresses of newly allocated memory:\n");
  for (i = 0; i<n2; ++i)
    printf("%pc\n", pts + i);
  free(pts);
  return 0;
}
```

output :-

```
Enter size: 5
Addresses of previously allocated Memory:
000000000736F50C
000000000736F54C
000000000736F58C
000000000736F5CC
000000000736F60C
Enter the new size: 3
Addres of newly allocated memory:
000000000736F50C
000000000736F54C
000000000736F58C
```

3) Stack implementation:

→

```
#include <stdio.h>
# define SIZE 5
int stack[SIZE];
int top = -1;
void push (int elements);
void pop();
void display();
int main() {
    int choice, element;
    do {
        printf("\n stack operations.: \n");
        printf("1. push \n");
        printf("2. pop\n");
        printf("3. Display \n");
        printf("4. Exit \n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
    switch (choice){
        case 1:
            printf("Enter the element to push: ");
            scan("%d" & element);
            push (element);
            break;
        case 2:
            pop();
            break;
        Case 3:
            display();
            break;
```

```
        case 4:
            printf ("Exiting the program.\n");
            break;
        default:
            printf("Invalid choice. please enter a valid
                    option.\n");
    }
} while (choice != 4);
return 0;
}
void push(int element){
    if (top == SIZE - 1) {
        printf("Stack overflow! Cannot push element.\n");
    } else {
        top++;
        stack[top] = element;
        print("%d pushed onto the stack.\n",
                element);
    }
}
void pop() {
    if (top == -1) {
        printf ("Stack underflow! Cannot pop
                element.\n");
    } else {
        printf ("%d popped from the stack.\n",
                stack[top]);
        top--;
    }
}
```

```
void display () {
    if (top == -1) {
        printf (" Stack is empty. Nothing to
            display. \n");
    } else {
        printf ("Elements in the Stack: \n");
        for (int i = 0; i <= top; i++) {
            printf ("%d ", stack[i]);
        }
        printf ("\n");
    }
}
```

output :

Stack operations :

1. push
2. pop
3. display
4. Exit

Enter your choice: 1.
Enter the element to push : 34
34 pushed onto the stack.

Stack operations:
1. push
2. pop
3. display
4. Exit
Enter your choice = 1
Enter the element to push : 9
9 pushed onto the stack

Stack operations:
1. push
2. pop
3. display
4. Exit
Enter your choice : 3
Elements in the stack:

3H
Stack operations:
1. push
2. pop
3. display
4. Exit

Enter your choice : 2
9 popped from the stack.

Stack operations:
1. push
2. pop
3. Display
4. Exit

Enter your choice : 4
Exiting the program.

21/12/2023