1BM22CS219 - RANJAN DEVI

WEEK 2

1.Swapping using pointers

#include <stdio.h>

void swap(int \*, int\*);

void main()

{

int a,b;

printf("Enter values of a and b:\n");

scanf("%d%d",&a,&b);

printf("Values of a and b before swapping: a=%d and b=%d",a,b);

swap(&a,&b);

printf("Values of a and b after swapping: a=%d and b=%d",a,b);

}

void swap(int \*p,int \*q)

{

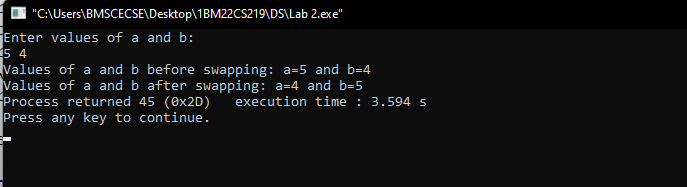
int temp;

temp=\*p;

\*p=\*q;

\*q=temp;

}

OUTPUT: 

2.Dynamic memory allocation

#include <stdio.h>

#include<stdlib.h>

void main()

{

int \*p,\*q,\*r;

int i,m,n;

printf("Enter no. of elements for p:");

scanf("%d",&m);

printf("Enter no. of elements for q:");

scanf("%d",&n);

p=(int\*)malloc(m\*sizeof(int));

q=(int\*)calloc(n,sizeof(int));

if(p==NULL && q==NULL)

printf("Memory is not allocated");

else

printf("Memory allocated successfully\n\n");

printf("Elements of p:\n");

for(i=0;i<m;i++)

printf("%d\t",i+1);

printf("\nElements of q:\n");

for(i=0;i<n;i++)

printf("%d\t",i);

free(p);

printf("\nMalloc Memory successfully freed.");

printf("\n\nEnter the new size of the array: ");

scanf("%d",&m);

r= (int\*)realloc(q, n \* sizeof(int));

if(r !=NULL)

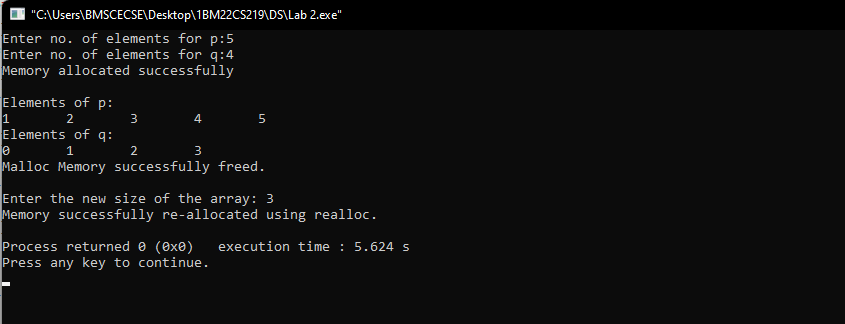
printf("Memory successfully re-allocated using realloc.\n");

else

printf("Memory not re-allocated");

}

OUTPUT:



3.Stack Implementation

#include <stdio.h>

int stack[100] , i , j , ch , n , top=-1;

void push();

void pop();

void display();

void main ()

{

printf("Enter the number of elements in the stack ");

scanf("%d",&n);

printf("Choose 1.Push 2.Pop 3.Show 4.Exit\n ");

while(ch != 4)

{

printf("Enter your choice :");

scanf("%d",&ch);

switch(ch)

{

case 1:

push();

break;

case 2:

pop();

break;

case 3:

display();

break;

case 4:

printf("Exited");

break;

default: printf("Please enter valid choice”);

};

}

}

void push ()

{

int val;

if (top == n )

printf("Overflow\n");

else

{

printf("Enter the value:");

scanf("%d",&val);

top = top +1;

stack[top] = val;

}

}

void pop ()

{

if(top == -1)

printf("Underflow\n");

else

top = top -1;

}

void display()

{ printf(“Stack:\n”);

for (i=top;i>=0;i--)

printf("%d\n",stack[i]);

if(top == -1)

printf("Stack is empty");

}

OUTPUT:

