1. Write a function to find the factorial of a number

IPO:

Input: number n

Process: using the function fact(), by multiplying i with f recursively

Output: factorial of the number

Code:

#include<stdio.h>

int fact(int n)

{

int f = 1;

for(int i = 1 ; i <= n; i++)

f\*=i;

return f;

}

int main()

{

int n;

scanf("%d",&n);

printf("The factorial of %d is %d",n,fact(n));

}

Output:



1. Write a function to check whether a number is prime.

IPO:

input: number n

Process: using the function prime() and by using for loop checking how many numbers can divide n and incrementing the value of count.

Output : n is prime number or not

Code:

#include<stdio.h>

int prime(int n)

{

int count=0;

for(int i = 1 ; i <= n ; i++)

{

if(n% i == 0)

count++;

}

if(count == 2)

printf("%d is prime\n",n);

else

printf("%d is not prime\n",n);

}

int main()

{

int n;

scanf("%d",&n);

prime(n);

return 0;

}

Output:



1. Write a function to calculate power using recursion.

IPO:  
Input: base and power say b,e

Process: using for loop recursively multiplying b by itself

Output: Calculate power using recursion

Code:

#include<stdio.h>

void power(int b , int e)

{

for(int i = 1 ; i<e ; i++)

{

b\*=b;

}

printf("%d",b);

}

int main()

{

int b,e;

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

power(b,e);

return 0;

}

Output:



1. Write a function to check palindrome number using recursion.

IPO:

Input: number say n

Process: recursively multiplying rev with and the last digit of the number and removing the last digit from the number recursively

Output: n is palindrome or not

Code:

#include<stdio.h>

void pal(int n)

{

int rev=0,r ;

int m = n;

while(n>0)

{

r = n % 10;

rev = rev\*10+r;

n = n / 10;

}

if(rev == m)

printf("Palindrome");

else

printf("not a palindrome");

}

void main()

{

int n;

scanf("%d",&n);

pal(n);

}

Output:



1. Write a function to calculate nCr (combinations).

IPO:

Input : numbers n and c

Process: using the formula n!/r!x (n-r)!

Output: value of nCr combination

Code:

#include<stdio.h>

int fact(int n)

{

int f = 1;

for(int i = 1 ; i <= n; i++)

f\*=i;

return f;

}

int main()

{

int n,r;

scanf("%d %d",&n,&r);

int value = fact(n)/(fact(r) \* fact(n-r));

printf("%dC%d = %d",n,r,value);

return 0;

}

Output:



1. Write a program to demonstrate call by value and call by reference.

IPO:  
Input: two numbers say a and b

Process: change the values of a and b using the functions callByValue and callByReference

Output: demonstrate call by value and call by reference

Code:

#include <stdio.h>

void callByValue(int x)

{

x = x + 10;

printf("Inside callByValue: x = %d\n", x);

}

void callByReference(int \*y) {

\*y = \*y + 10;

printf("Inside callByReference: y = %d\n", \*y);

}

int main() {

int a = 5, b = 5;

printf("Before callByValue: a = %d\n", a);

callByValue(a);

printf("After callByValue: a = %d\n\n", a);

printf("Before callByReference: b = %d\n", b);

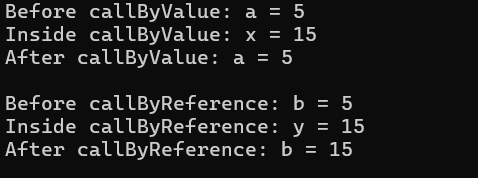
callByReference(&b);

printf("After callByReference: b = %d\n", b);

return 0;

}

Output:



1. Write a program using function to swap two numbers.

IPO

Input:

Code: two numbers a,b

Process: using a temporary variable swap the values of a and b

Output: swap a and b

Code:

#include<stdio.h>

void swap(int a , int b)

{

printf("Before swapping a = %d and b = %d\n",a,b);

int temp = a;

a = b;

b = temp;

printf("After swapping a = %d and b = %d",a,b);

}

int main()

{

int a,b;

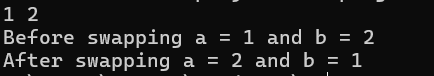
scanf("%d%d", &a,&b);

swap(a,b);

return 0;

}

Output:



1. Write a recursive function to find the nth Fibonacci number.

IPO:

Input: nth fibonacci number n

Process: using function fibo() and by using for loop by adding b and c and assigning it to a and assigning b to c and a to b . This occurs till the loop ends

Output:nth fibonacci number

Code:

#include<stdio.h>

void fibo(int n)

{

int a, b = 1, c = - 1;

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

{

a= b + c;

c = b;

b = a;

}

printf("%d nth element = %d",n,a);

}

void main()

{

int n;

scanf("%d",&n);

fibo(n);

}

Output:



1. Write a program to find GCD and LCM using functions

IPO:

Input: two numbers a and b

Process: using the function GCD\_LCM() and calculating its greatest common divisor and least common multiple

Output: GCD and LCM of a and b

Code:

#include<stdio.h>

void GCD\_LCM(int n,int m)

{

//GCD

int c;

if(n >m)

c = m;

else

c = n;

for(int i = c; i > 0; i--)

{

if(m% i == 0 && n % i ==0)

{

printf("GCD of %d and %d is %d\n",m,n,i);

break;

}

}

//LCM

if(n > m)

c = n;

else

c = m;

for(int i = c; i <= m\*n; i++)

{

if(i%n == 0 && i % m == 0)

{

printf("The LCM of %d and %d is %d",m,n,i);

break;

}

}

}

void main()

{

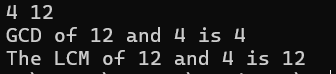
int a,b;

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

GCD\_LCM(a,b);

}

Output:



1. Write a program to demonstrate global and local variables

IPO:

Input: global and local variable

Process: declaring global variable and declaring local variable in the two functions and showing how the its values are affected when changed inside different functions

Output: How values of global variable and local variables change in different functions

Code:

.#include <stdio.h>

int globalVar = 10;

void display()

{

int localVar = 20;

printf("Inside display():\n");

printf("Global variable: %d\n", globalVar);

printf("Local variable: %d\n\n", localVar);

}

int main()

{

int localVar = 5;

printf("Inside main():\n");

printf("Global variable: %d\n", globalVar);

printf("Local variable: %d\n\n", localVar);

display();

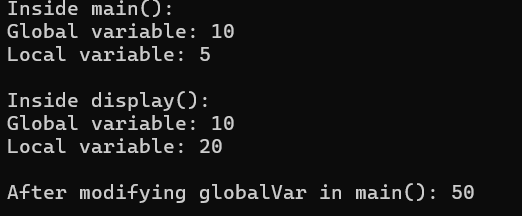
globalVar = 50;

printf("After modifying globalVar in main(): %d\n", globalVar);

return 0;

}

Output:



.