Day 6: Functions (8-8-2025)

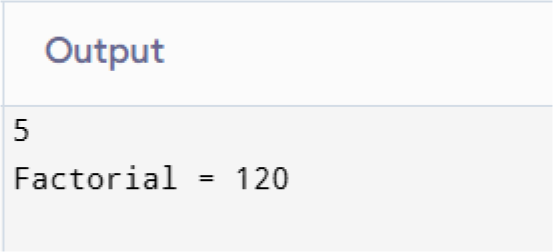
1. Function to find factorial of a number

IPO:
INPUT: An integer n
PROCESS: Multiply numbers from 1 to n
OUTPUT: Factorial of n

CODE;

```
#include <stdio.h>
int factorial(int n)
{
   int i,fact = 1;
   for( i = 1; i <= n; i++)
      fact =fact* i;
   return fact;
}
void main()
{
   int num;
   scanf("%d", &num);
   printf("Factorial = %d", factorial(num));
}
```

OUTPUT;



```
5
Factorial = 120
```

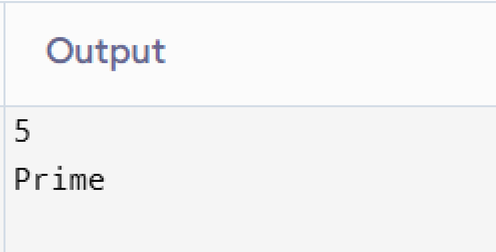
2. Function to check prime

IPO:
INPUT: An integer ‘n’
PROCESS: Check divisibility from 2 to n/2
OUTPUT: Prime or not

CODE;

```
#include <stdio.h>
int Prime(int n)
{
    int i;
    if(n < 2) return 0;
    for(i = 2; i <= n/2; i++)
        if(n % i == 0)
            return 0;
    return 1;
}
void main()
{
    int num;
    scanf("%d", &num);
    if(Prime(num))
        printf("Prime");
    else
        printf("Not Prime");
}
```

OUTPUT;



```
5
Prime
```

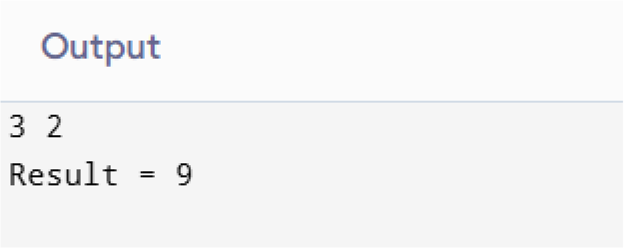
3. Power using recursion

IPO:
 INPUT: Base a, exponent b
 PROCESS: Multiply base recursively
 OUTPUT: a^b

CODE;

```
#include <stdio.h>
int power(int a, int b)
{
    if(b == 0) return 1;
    return a * power(a, b-1);
}
void main()
{
    int base, exp;
    scanf("%d%d", &base, &exp);
    printf("Result = %d", power(base, exp));
}
```

OUTPUT;



```
3 2
Result = 9
```

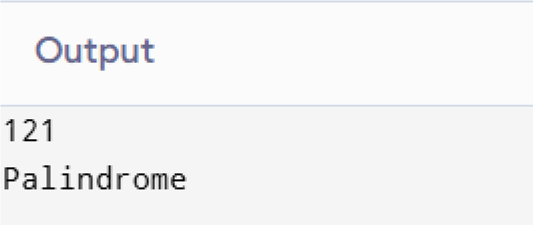
4. Palindrome number using recursion

IPO:
INPUT: A number
PROCESS: Reverse digits recursively and compare
OUTPUT: Palindrome or not

CODE;

```
#include <stdio.h>
int reverseNum(int num, int rev)
{
    if(num == 0)
    return rev;
    return reverseNum(num / 10, rev * 10 + num % 10);
}
void main()
{
    int n;
    scanf("%d", &n);
    if(n == reverseNum(n, 0))
        printf("Palindrome");
    else
        printf("Not Palindrome");
}
```

OUTPUT;



```
Output
121
Palindrome
```

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

INPUT: Two integers `n` and `r` from the user
PROCESS:

Find factorial of `n`, `r`, and `(n - r)`

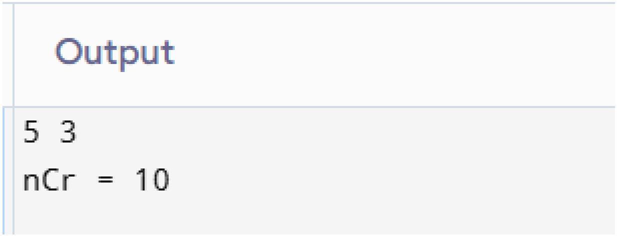
Apply formula `nCr = n! / (r! × (n - r)!)`

OUTPUT: Value of nCr (number of combinations)

CODE;

```
#include <stdio.h>
int factorial(int n)
{
    int fact = 1;
    for(int i = 1; i <= n; i++)
        fact =fact* i;
    return fact;
}
int nCr(int n, int r)
{
    return factorial(n) / (factorial(r) * factorial(n - r));
}
void main()
{
   int n, r;
   scanf("%d%d", &n, &r);
   printf("nCr = %d", nCr(n, r));

}
```

OUTPUT;



```
5 3
nCr = 10
```

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

IPO:

INPUT: Two integers

PROCESS: Show how changes affect variables in call by value & reference

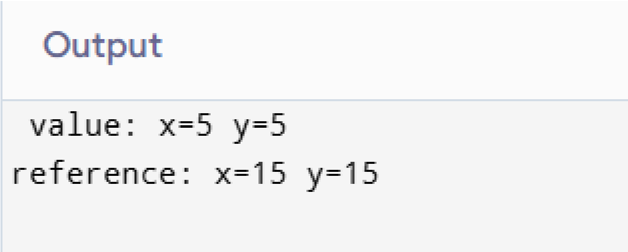
OUTPUT: Demonstration

CODE;

```
#include <stdio.h>
void Value(int a, int b)
{
   a = a + 10;
   b = b + 10;
}
void Reference(int *a, int *b)
```

```
{
   *a = *a + 10;
   *b = *b + 10;
}
void main()
{
   int x = 5, y = 5;
   Value(x, y);
   printf(" value: x=%d y=%d\n", x, y);
   Reference(&x, &y);
   printf("reference: x=%d y=%d\n", x, y);
}
```

OUTPUT;



Output

```
 value: x=5 y=5
reference: x=15 y=15
```

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

IPO:

INPUT: Two integers

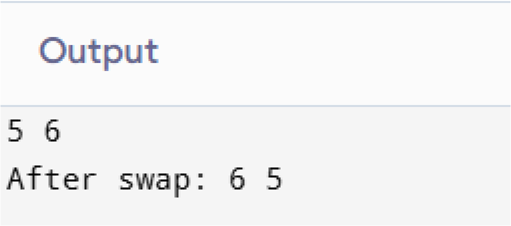
PROCESS: Swap values using a function

OUTPUT: Swapped numbers

CODE:

```
#include <stdio.h>
void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
void main()
{
    int x, y;
    scanf("%d%d", &x, &y);
    swap(&x, &y);
    printf("After swap: %d %d", x, y);
}
```

OUTPUT:



```
5 6
After swap: 6 5
```

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

IPO:

INPUT: Two integers

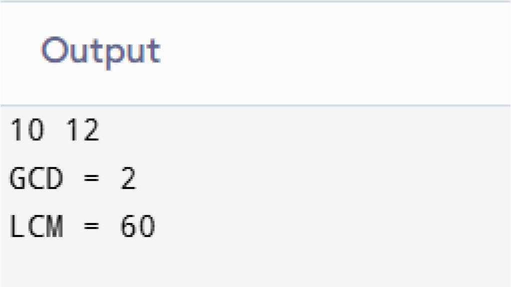
PROCESS: Find GCD using Euclidean method, LCM = (a*b)/GCD

OUTPUT: GCD and LCM

CODE;

```
#include <stdio.h>
int gcd(int a, int b)
{
   if(b == 0) return a;
   return gcd(b, a % b);
}
int lcm(int a, int b)
{
   return (a * b) / gcd(a, b);
}
void main()
{
   int a, b;
   scanf("%d%d", &a, &b);
   printf("GCD = %d\n", gcd(a, b));
   printf("LCM = %d", lcm(a, b));
}
```

OUTPUT;



```
10 12
GCD = 2
LCM = 60
```

10.Write a program to demonstrate global and local variables

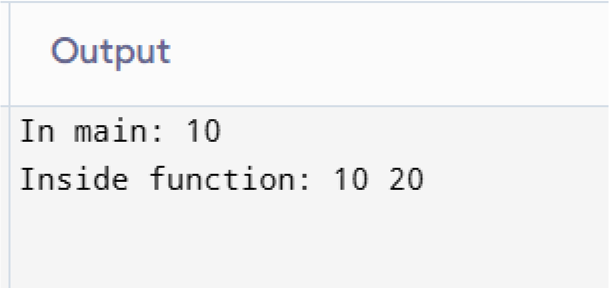
IPO:
INPUT: two integers

PROCESS: Show difference between local & global scope
OUTPUT: Variable values

CODE;

```
#include <stdio.h>

int globalVar = 10;

void show()

{

   int localVar = 20;

   printf("Inside function: %d %d\n", globalVar, localVar);

}

void main()

{

   printf("In main: %d\n", globalVar);

   show();

}
```

OUTPUT;



```
In main: 10
Inside function: 10 20
```