Assignment

Reg no:192511101

S.Harish

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

IPO:

* Input: A number n
* Process: Multiply all integers from 1 to n
* Output: Factorial of n

Program code:

#include <stdio.h>

Void factorial(int n)

{

if (n == 0)

return 1;

return n \* factorial(n - 1);

}

int main()

{

int num = 5;

printf("Factorial of %d is %d\n", num, factorial(num));

}

Sample output:

Factorial of 5 is 120

2) Write a function to check whether a number is prime.

IPO:

* Input: A number n
* Process: Check divisibility by numbers from 2 to √n
* Output: Whether n is prime or not

Program code:

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int n)

{

if (n <= 1) return false;

for (int i = 2; i \* i <= n; i++)

if (n % i == 0)

return false;

return true;

}

Void main()

{

int num = 17;

if (isPrime(num))

printf("%d is a prime number.\n", num);

else

printf("%d is not a prime number.\n", num);

}

Sample output:

17 is a prime number.

3) Write a function to calculate power using recursion.

IPO:

* Input: Base and exponent
* Process: Multiply base x recursively y times
* Output: Result of x^y

Program code:

#include <stdio.h>

Int power(int x, int y)

{

if (y == 0)

return 1;

return x \* power(x, y - 1);

}

Void main()

{

printf("2^4 = %d\n", power(2, 4));

}

Sample output:

2^4 = 16

4) Write a function to check palindrome number using recursion.

IPO:

* Input: Number n
* Process: Reverse the number using recursion and compare
* Output: Whether it’s a palindrome

Program code:

#include <stdio.h>

int reverse(int n, int rev)

{

if (n == 0) return rev;

return reverse(n / 10, rev \* 10 + n % 10);

}

Void main()

{

int num = 121;

if (num == reverse(num, 0))

printf("%d is a palindrome.\n", num);

else

printf("%d is not a palindrome.\n", num);

}

Sample output:

121 is a palindrome.

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

IPO:

* Input: n and r
* Process: Use formula nCr = n! / (r! \* (n-r)!)
* Output: nCr value

Program code:

#include <stdio.h>

int factorial(int n)

{

if (n == 0) return 1;

return n \* factorial(n - 1);

}

int nCr(int n, int r)

{

return factorial(n) / (factorial(r) \* factorial(n - r));

}

Void main()

{

printf("5C3 = %d\n", nCr(5, 3));

}

Sample output:

5C3 = 10

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

IPO:

* Input: Two variables
* Process: Show difference in call by value and reference
* Output: Effect on original variables

Program code:

#include <stdio.h>

void callByValue(int a)

{

a = a + 10;

}

void callByReference(int \*a)

{

\*a = \*a + 10;

}

Int main()

{

int x = 5;

callByValue(x);

printf("After call by value: %d\n", x);

callByReference(&x);

printf("After call by reference: %d\n", x);

Return 0;

}

Sample output:

After call by value: 5

After call by reference: 15

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

IPO:

* Input: Two numbers
* Process: Swap their values using a function
* Output: Swapped values

Program code:

#include <stdio.h>

void swap(int \*a, int \*b)

{

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main()

{

int x = 10, y = 20;

swap(&x, &y);

printf("After swap: x = %d, y = %d\n", x, y);

return 0;

}

Sample output:

After swap: x = 20, y = 10

8) Write a recursive function to find the nth Fibonacci number.

IPO:

* Input: n
* Process: Return nth term using fib(n-1) + fib(n-2)
* Output: Fibonacci number

Program code:

#include <stdio.h>

int fibonacci(int n)

{

if (n <= 1)

return n;

return fibonacci(n - 1) + fibonacci(n - 2);

}

int main()

{

int n = 7;

printf("Fibonacci(%d) = %d\n", n, fibonacci(n));

return 0;

}

Sample output:

Fibonacci(7) = 13

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

IPO:

* Input: Two numbers
* Process: Calculate GCD and LCM
* Output: GCD and LCM

Program 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);

}

int main()

{

int a = 12, b = 18;

printf("GCD = %d\n", gcd(a, b));

printf("LCM = %d\n", lcm(a, b));

return 0;

}

Sample output:

GCD = 6

LCM = 36

10) Write a program to demonstrate global and local variables.

IPO:

* Input: None
* Process: Demonstrate scope
* Output: Values of global and local variables

Program code:

#include <stdio.h>

int global = 100; // Global variable

void show()

{

int local = 50; // Local variable

printf("Global: %d, Local: %d\n", global, local);

}

int main()

{

show();

return 0;

}

Sample output:

Global: 100, Local: 50