University of Mila

Faculty of Science and Technology

Department of Process Engineering

Practical Work 03

Course : Introduction to Programming

Level : 1st year ST - ENG & LMD

Semester 02

By :

Dr. KECITA Farouk

Academic Year : 2025/2026

Question 01 : Basic Multiplication Function

Program : multiplyNumbers - Basic Version

```c
#include <stdio.h>

// Function prototype
int multiplyNumbers(int a, int b);

int main() {
    int num1, num2, result;

    printf("Enter first number: ");
    scanf("%d", &num1);
    printf("Enter second number: ");
    scanf("%d", &num2);

    // Call the function
    result = multiplyNumbers(num1, num2);

    printf("Product: %d * %d = %d\n", num1, num2, result);

    return 0;
}

// Function definition
int multiplyNumbers(int a, int b) {
    return a * b;
}
```

Example Output

```
Enter first number: 7
Enter second number: 8
Product: 7 * 8 = 56

Another execution:
Enter first number: 12
Enter second number: 5
Product: 12 * 5 = 60
```

Question 02 : Multiplication Using Pointers

Program : multiplyNumbers - Pointer Version

```c
#include <stdio.h>

// Function prototype using pointers
int multiplyNumbers(int *a, int *b);

int main() {
    int num1, num2, result;

    printf("Enter first number: ");
    scanf("%d", &num1);
    printf("Enter second number: ");
    scanf("%d", &num2);

    // Call the function with addresses
    result = multiplyNumbers(&num1, &num2);

    printf("Product: %d * %d = %d\n", num1, num2, result);

    return 0;
}

// Function definition using pointers
int multiplyNumbers(int *a, int *b) {
    return (*a) * (*b);
}
```

Example Output

```
Enter first number: 15
Enter second number: 4
Product: 15 * 4 = 60

Another execution:
Enter first number: -8
Enter second number: 3
Product: -8 * 3 = -24
```

Exercise N 02 : Advanced Swap Functions

Question 01 : Basic Swap Three Function (Pass by Value)

Program : swapThree - Pass by Value

```c
#include <stdio.h>

// Function prototype
void swapThree(int a, int b, int c);

int main() {
    int x = 10, y = 20, z = 30;

    printf("Before swap: x=%d, y=%d, z=%d\n", x, y, z);
    swapThree(x, y, z);
    printf("After swap: x=%d, y=%d, z=%d\n", x, y, z);
    printf("(Note: This swaps values inside the function only
   . ");
    printf("For actual swapping, use pointers.)\n");

    return 0;
}

// Function definition
void swapThree(int a, int b, int c) {
    int temp = a;
    a = c;
    c = b;
    b = temp;
    printf("Inside function: a=%d, b=%d, c=%d\n", a, b, c);
}
```

Example Output

```
Before swap: x=10, y=20, z=30
Inside function: a=30, b=10, c=20
After swap: x=10, y=20, z=30
(Note: This swaps values inside the function only.For actual swapping,
 use pointers.)
```

Question 02 : Swap Three Using Pointers

Program : swapThree - Pointer Version

```c
#include <stdio.h>

// Function prototype using pointers
void swapThree(int *a, int *b, int *c);

int main() {
    int x = 10, y = 20, z = 30;

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

    return 0;
}

// Function definition using pointers
void swapThree(int *a, int *b, int *c) {
    int temp = *a;
    *a = *c;
    *c = *b;
    *b = temp;
}
```

Example Output

```
Before swap: x=10, y=20, z=30
After swap: x=30, y=10, z=20

Another execution with different values:
Before swap: x=5, y=15, z=25
After swap: x=25, y=5, z=15
```

Exercise N 03 : Statistical Functions

Question 01 : Max and Min of Three Numbers

Program : MaxMinThree

```c
#include <stdio.h>
// Function prototype using pointers to return multiple
   values
void MaxMinThree(int a, int b, int c, int *max, int *min);
int main() {
    int num1, num2, num3;
    int max, min;
    printf("Enter three integers: ");
    scanf("%d %d %d", &num1, &num2, &num3) ;
    // Call the function and pass addresses of max and min
    MaxMinThree(num1, num2, num3, &max, &min);
    printf("\n--- Results ---\n");
    printf("Numbers entered: %d, %d,%d\n", num1, num2, num3);
    printf("Maximum: %d\n", max);
    printf("Minimum: %d\n", min);
    return 0;
}
// Function definition
void MaxMinThree(int a, int b, int c, int *max, int *min) {
    // Logic for Maximum
    *max = a;
    if (b > *max) *max = b;
    if (c > *max) *max = c;
    // Logic for Minimum
    *min = a;
    if (b < *min) *min = b;
    if (c < *min) *min = c;
}
```

Example Output

```
Enter three integers: 42 18 35
--- Results ---
Numbers entered: 42, 18, 35
Maximum: 42
Minimum: 18
Another execution:
Enter three integers: -5 12 7
--- Results ---
Numbers entered: -5, 12, 7
Maximum: 12
Minimum: -5
```

Question 02 : Find Minimum Function

Program : findMin

```c
#include <stdio.h>

// Function prototype
int findMin(int a, int b, int c);

int main() {
    int num1, num2, num3, minimum;

    printf("Enter three numbers: ");
    scanf("%d %d %d", &num1, &num2, &num3);

    minimum = findMin(num1, num2, num3);

    printf("The smallest number is: %d\n", minimum);

    return 0;
}

// Function definition
int findMin(int a, int b, int c) {
    int min = a;
    if (b < min) min = b;
    if (c < min) min = c;
    return min;
}
```

Example Output

```
Enter three numbers: 25 17 42
The smallest number is: 17

Another execution:
Enter three numbers: -8 -3 -15
The smallest number is: -15
```

Exercise N 04 : Number Classification

Program : Prime Number Checker

```c
#include <stdio.h>

// Function prototype
int isPrime(int n);

int main() {
    int num;

    printf("Enter a number: ");
    scanf("%d", &num);

    if (isPrime(num)) {
        printf("%d is a prime number.\n", num);
    } else {
        printf("%d is not a prime number.\n", num);
    }

    return 0;
}

// Function to check if a number is prime
int isPrime(int n) {
    if (n <= 1) return 0; // Not prime

    for (int i = 2;  i < n; i++) {
        if (n % i == 0) {
            return 0; // Not prime
        }
    }
    return 1; // Prime
}
```

Example Output

```
Enter a number: 17
17 is a prime number.

Another execution:
Enter a number: 25
25 is not a prime number.
```

Exercise N 05 : Range Printing with Functions

Question 01 : Print All Prime Numbers Between 1 to n

Program : Print Prime Numbers in Range

```c
#include <stdio.h>

// Function prototype
int isPrime(int n);

int main() {
    int n;

    printf("Enter a number n: ");
    scanf("%d", &n);

    printf("Prime numbers between 1 and %d are:\n", n);
    for (int i = 2; i <= n; i++) {
        if (isPrime(i)) {
            printf("%d ", i);
        }
    }
    printf("\n");

    return 0;
}

// Function to check if a number is prime
int isPrime(int n) {
    if (n <= 1) return 0;
    for (int i = 2; i < n; i++) {
        if (n % i == 0) return 0;
    }
    return 1;
}
```

Example Output

```
Enter a number n: 20
Prime numbers between 1 and 20 are:
2 3 5 7 11 13 17 19

Another execution:
Enter a number n: 30
Prime numbers between 1 and 30 are:
2 3 5 7 11 13 17 19 23 29
```

Exercise N 06 : Advanced Recursion

Question 01 : Recursive Power Function

Program : power - Recursive Version

```c
#include <stdio.h>

// Function prototype
int power(int base, int exponent);

int main() {
    int base = 2;
    int exponent = 10;
    int result = power(base, exponent);

    printf("%d^%d = %d\n", base, exponent, result);

    return 0;
}

// Recursive function to calculate power
int power(int base, int exponent) {
    if (exponent == 0) {
        return 1;
    } else {
        return base * power(base, exponent - 1);
    }
}
```

Example Output

```
2^10 = 1024

Testing other values:
3^4 = 81
5^3 = 125
10^0 = 1
```

Question 02(Optional) : Fibonacci Series Using Recursion

Program : Fibonacci Series - Recursive Version

```c
#include <stdio.h>

// Function prototype
int fibonacci(int n);

int main() {
    int n;

    printf("Enter value of N: ");
    scanf("%d", &n);

    printf("Fibonacci series up to %d:\n", n);
    for (int i = 0; i < n; i++) {
        printf("%d ", fibonacci(i));
    }
    printf("\n");

    return 0;
}

// Recursive function to get nth Fibonacci number
int fibonacci(int n) {
    if (n <= 1) {
        return n;
    }
    return fibonacci(n - 1) + fibonacci(n - 2);
}
```

Example Output

```
Enter value of N: 10
Fibonacci series up to 10:
0 1 1 2 3 5 8 13 21 34
```

Question 03 : Sum of Natural Numbers Using Recursion

Program : Sum of Natural Numbers - Recursive Version

```c
#include <stdio.h>

// Function prototype
int sumNatural(int n);

int main() {
    int n;

    printf("Enter value of N: ");
    scanf("%d", &n);

    int result = sumNatural(n);

    printf("Sum of natural numbers from 1 to %d = %d\n", n,
   result);

    return 0;
}

// Recursive function to calculate sum
int sumNatural(int n) {
    if (n == 0) {
        return 0;
    } else {
        return n + sumNatural(n - 1);
    }
}
```

Example Output

```
Enter value of N: 10
Sum of natural numbers from 1 to 10 = 55

Another execution:
Enter value of N: 5
Sum of natural numbers from 1 to 5 = 15
```