Name: Vanshaj Rana

UID: 22BCS15417

Section: 620 B

DOMAIN WINTER WINNING CAMP(Day-1)

1) Sum of Natural Numbers up to

N Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;

    if (n > 0) {
        int sum = n * (n + 1) / 2;  // Using the formula
        cout << "The sum of natural numbers from 1 to " << n << " is: " << sum <<
endl;
    } else {
        cout << "Please enter a positive integer!" << endl;
    }

    return 0;
}
```

Output:

```
Enter a positive integer: 5
The sum of natural numbers from 1 to 5 is: 15
```

2) Count Digits in a Number

```
#include <iostream>
using namespace std;

int countDigits(int n) {
    int count = 0;
    while (n > 0) {
        n /= 10; // Remove the last digit
        count++;
    }
    return count;
}

int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;

    if (n > 0) {
        int digitCount = countDigits(n);
        cout << "The number of digits in " << n << " is: " << digitCount << endl;
    } else {
        cout << "Please enter a positive integer!" << endl;
    }
return 0;
}
```

Output:

```
Enter a positive integer: 12345
The number of digits in 12345 is: 5
```

3) Function Overloading for finding maximum of two numbers, three numbers and two floating number.

```
#include <iostream>
using namespace std;

// Overloaded function to find the maximum of two integers int
max(int a, int b) {
    return (a > b) ? a : b;
}

// Overloaded function to find the maximum of three integers
int max(int a, int b, int c) {
    return (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);
}

// Overloaded function to find the maximum of two floating-point numbers
float max(float a, float b) {
    return (a > b) ? a : b;
}
```

```
int main() {
    int choice;
    cout << "Choose the operation:\n";
    cout << "1. Maximum of two integers\n";
    cout << "2. Maximum of three integers\n";
    cout << "3. Maximum of two floating-point numbers\n";
    cin >> choice;

    if (choice == 1) {
        int a, b;
        cout << "Enter two integers: ";
        cin >> a >> b;
        cout << "Maximum of " << a << " and " << b << " is: " << max(a, b) <<
        endl;
    } else if (choice == 2) {
        int a, b, c;
        cout << "Enter three integers: ";
        cin >> a >> b >> c;
        cout << "Maximum of " << a << ", " << b << " and " << c << " is: " <<
max(a, b, c) << endl;
    } else if (choice == 3) {
        float x, y;
        cout << "Enter two floating-point numbers: ";
        cin >> x >> y;
        cout << "Maximum of " << x << " and " << y << " is: " << max(x, y) <<
        endl;
    } else {
        cout << "Invalid choice!" << endl;
    }

    return 0;
}
```

Output:

```
Choose the operation:
1. Maximum of two integers
2. Maximum of three integers
3. Maximum of two floating-point numbers
1
Enter two integers: 5 10
Maximum of 5 and 10 is: 10
```

4) Function Overloading for Calculating Area.

```
#include <iostream>
#include <cmath>
using namespace std;

// Function to calculate the area of a circle
double area(double radius) {
    return M_PI * radius * radius; // M_PI is a constant for π
}

// Function to calculate the area of a rectangle
double area(double length, double width) {
    return length * width;
```

```
}

// Function to calculate the area of a triangle
double area(double base, double height, bool isTriangle) {
    if (isTriangle) {
        return 0.5 * base * height;
    }
    return 0; // Fallback, not used in practice
}

int main() {
    int choice;
    cout << "Choose a shape to calculate the area:\n";
    cout << "1. Circle\n";
    cout << "2. Rectangle\n";
    cout << "3. Triangle\n";
    cin >> choice;

    if (choice == 1) {
        double radius;
        cout << "Enter the radius of the circle: ";
        cin >> radius;
        cout << "Area of the circle: " << area(radius) << endl;
    } else if (choice == 2) {
        double length, width;
        cout << "Enter the length and width of the rectangle: ";
        cin >> length >> width;
        cout << "Area of the rectangle: " << area(length, width) << endl;
    } else if (choice == 3) {
        double base, height;
        cout << "Enter the base and height of the triangle: ";
        cin >> base >> height;
        cout << "Area of the triangle: " << area(base, height, true) << endl;
    } else {
        cout << "Invalid choice!" << endl;
    }

    return 0;
}
```

Output:

```
Choose a shape to calculate the area:
1. Circle
2. Rectangle
3. Triangle
1
Enter the radius of the circle: 5
Area of the circle: 78.5398
```

5) Matrix Multiplication Using Function Overloading

```
#include <iostream>
#include <vector>
using namespace std;
```

```
// Function to perform matrix addition
vector<vector<int>>    operate(const    vector<vector<int>>&    A,    const
vector<vector<int>>& B, int operationType) {
   int m = A.size(), n = A[0].size();
   vector<vector<int>> result(m, vector<int>(n,
   0)); if (operationType == 1) { // Matrix Addition
      for (int i = 0; i < m; i++)
         { for (int j = 0; j < n;
         j++) {
            result[i][j] = A[i][j] + B[i][j];
         }
      }
   }
   return result;
}

// Function to perform matrix multiplication
vector<vector<int>>    operate(const    vector<vector<int>>&    A,    const
vector<vector<int>>& B) {
   int m = A.size(), n = A[0].size(), p = B[0].size();
   vector<vector<int>> result(m, vector<int>(p, 0));
   for (int i = 0; i < m; i++) {
      for (int j = 0; j < p; j++) {
         for (int k = 0; k < n; k++) {
            result[i][j] += A[i][k] * B[k][j];
         }
      }
```

```
    }
    return result;
}

int main() {
    int m, n, p, operationType;

    cout << "Enter dimensions of Matrix A (m n): ";
    cin >> m >> n;
    vector<vector<int>> A(m, vector<int>(n));

    cout << "Enter elements of Matrix A:\n";
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cin >> A[i][j];
        }
    }

    cout << "Enter dimensions of Matrix B (n p): ";
    cin >> n >> p;
    vector<vector<int>> B(n, vector<int>(p));

    cout << "Enter elements of Matrix B:\n";
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < p; j++) {
            cin >> B[i][j];
        }
    }

    cout << "Choose operation type (1 for Addition, 2 for Multiplication): ";
    cin >> operationType;

    if (operationType == 1) {
        if (A.size() == B.size() && A[0].size() == B[0].size()) {
            vector<vector<int>> result = operate(A, B, 1);
            cout << "Result of Matrix Addition:\n";
            for (const auto& row : result) {
                for (int elem : row) {
                    cout << elem << " ";
                }
                cout << endl;
            }
        } else {
            cout << "Invalid dimensions for operation." << endl;
        }
    } else if (operationType == 2)
        { if (A[0].size() == B.size())
        {
            vector<vector<int>> result = operate(A, B);
            cout << "Result of Matrix Multiplication:\n";
            for (const auto& row : result) {
                for (int elem : row) {
                    cout << elem << " ";
                }
                cout << endl;
            }
        } else {
            cout << "Invalid dimensions for operation." << endl;
        }
    } else {
        cout << "Invalid operation type." << endl;
    }

    return 0;
}
```

Output:

```
Enter dimensions of Matrix A (m n): 2 2
Enter elements of Matrix A:
1 2
3 4
Enter dimensions of Matrix B (n p): 2 2
Enter elements of Matrix B:
5 6
7 8
Choose operation type (1 for Addition, 2 for Multiplication): 1
Result of Matrix Addition:
6 8
10 12
```