Week 8

1# Write a C++ program that uses functions to perform the following operations:

i. To insert a sub string into a given main string from a given position.
ii. ii. To delete n characters from a given position in a given string

main.cpp

Run

Output

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

// Function to insert a substring into a main string at a given position
    string insert_str(string m_str, string sub_str, int position) {
    m_str.insert(position, sub_str);
    return m_str;
}

// Function to delete n characters from a given position in a given string
    string del_char(string m_str, int position, int n) {
    m_str.erase(position, n);
    return m_str;
}

int main() {
    string m_str = "Hello, World!";
    string sub_str = "C++ ";
    int position = 7;

    // Insert the substring into the main string at the given position
    string result = insert_str(m_str, sub_str, position);
    cout << "After insertion: " << result << endl;
```

```
/tmp/JvJkzf7zJV.o
After insertion: Hello, C++ World!
After deletion: Hello, World!
```

2# Write a C++ program to determine if the given string is a palindrome or not.

main.cpp

Run

Output

```cpp
#include <algorithm>
using namespace std;

// Function to check if a string is a palindrome
bool is_palindrome(string str)
{
    string rev = str;
    reverse(rev.begin(), rev.end());
    return (str == rev);
}

int main()
{
    string str;
    cout << "Enter a string: ";
    cin >> str;

    if (is_palindrome(str)) {
        cout << str << " is a palindrome." << endl;
    } else {
        cout << str << " is not a palindrome." << endl;
    }

    return 0;
```

```
/tmp/JvJkzf7zJV.o
Enter a string: Hello
Hello is not a palindrome.
```

3# Write a C++ program to find a string within a sentence and replace it with another string.

main.cpp

Run

Output

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

// Function for demonstration
void replaceDemo(string s1, string s2, string s3, string s4)
{
    // Replaces 7 characters from 0th index by s2
    s1.replace(0, 7, s2);
    cout << s1 << endl;

    // Replaces 3 characters from 0th index with "Hello"
    s4.replace(0, 3, "Hello ");
    cout << s4 << endl;

    // Replaces 5 characters from 6th index of s4 with
    // 5 characters from 0th of s3
    s4.replace(6, 5, s3, 0, 5);
    cout << s4 << endl;

    // Replaces 5 characters from 6th index of s4 with
    // 6 characters from string "to all"
    s4.replace(6, 5, "to all", 6);
    cout << s4 << endl;
```

```
/tmp/JvJkzf7zJV.o
Demonstration of replace
Hello World !
Hello Geeks !
Hello to all !
Hello to all!!!!
```

4# Write a C++ program that reads a line of text and counts all occurrence of a particular word.

main.cpp

Run

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

int main() {
    string text;
    string word;
    int count = 0;

    // Input the text and word to search for
    cout << "Enter a line of text: ";
    getline(std::cin, text);

    cout << "Enter the word to count: ";
    cin >> word;

    // Search for the word and count its occurrences
    size_t pos = 0;
    while ((pos = text.find(word, pos)) != std::string::npos) {
        count++;
        pos += word.length();
    }

    // Display the count
```

Output

```
/tmp/JvJkzf7zJV.o
Enter a line of text: Hello I Am Manas
Enter the word to count: Manas
The word "Manas" occurs 1 times in the given text.
```

5# Write a C++ program that displays the position or index in the string S where the string T begins, or 1 if S doesn't contain T.

main.cpp

Run

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

int main() {
    string S, T;
    cout << "Enter the string S: ";
    cin >> S;
    cout << "Enter the string T: ";
    cin >> T;

    size_t found = S.find(T); // Search for T in S

    if (found != std::string::npos) {
        // If T is found in S, display the position (index) where it begins
        cout << "String T begins at position " << found + 1 << " in string S." <<
            endl;
    } else {
        // If T is not found in S, display 1
        cout << "String T is not found in string S, so the result is 1." << endl;
    }

    return 0;
}
```

Output

```
/tmp/JvJkzf7zJV.o
Enter the string S: ManasMaheshwari
Enter the string T: ana
String T begins at position 2 in string S.
```

Week 9

1# Write C programs that use both recursive and non-recursive functions to find:

a) The factorial of a given integer.

main.cpp

Run

```c
#include <stdio.h>

// Non-recursive function to find factorial
int factorialNonRecursive(int n) {
    int result = 1;
    for (int i = 1; i <= n; i++) {
        result *= i;
    }
    return result;
}

// Recursive function to find factorial
int factorialRecursive(int n) {
    if (n == 0 || n == 1) {
        return 1;
    } else {
        return n * factorialRecursive(n - 1);
    }
}

int main() {
    int num;
    printf("Enter a non-negative integer: ");
```

Output

```
/tmp/JvJkzf7zJV.o
Enter a non-negative integer: 23
Factorial (Non-recursive): 862453760
Factorial (Recursive): 862453760
```

b) To find the greatest common divisor of two given integers.

main.cpp

Run

```c
#include <stdio.h>

// Non-recursive function to find GCD
int gcdNonRecursive(int a, int b) {
    int temp;
    while (b != 0) {
        temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

// Recursive function to find GCD
int gcdRecursive(int a, int b) {
    if (b == 0) {
        return a;
    } else {
        return gcdRecursive(b, a % b);
    }
}

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

Output

```
/tmp/JvJkzf7zJV.o
Enter two positive integers: 23
15
GCD (Non-recursive): 1
GCD (Recursive): 1
```

2# Write C programs that use both recursive and non-recursive functions to solve towers of Hanoi problem.

main.cpp

Run

```c
#include <stdio.h>

void towersOfHanoi(int n, char source, char auxiliary, char target) {
    if (n == 1) {
        printf("Move disk 1 from %c to %c\n", source, target);
        return;
    }

    towersOfHanoi(n - 1, source, target, auxiliary);
    printf("Move disk %d from %c to %c\n", n, source, target);
    towersOfHanoi(n - 1, auxiliary, source, target);
}

int main() {
    int numDisks;
    printf("Enter the number of disks: ");
    scanf("%d", &numDisks);

    if (numDisks < 1) {
        printf("Number of disks should be at least 1.\n");
    } else {
        towersOfHanoi(numDisks, 'A', 'B', 'C');
    }

    return 0;
```

Output

```
/tmp/JvJkzf7zJV.o
Enter the number of disks: 3
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```

3# Write a C++ program to print the transpose of a given matrix using function.

main.cpp

```
25      int matrix[100][100];
26      int rows, cols;
27
28      std::cout << "Enter the number of rows: ";
29      std::cin >> rows;
30      std::cout << "Enter the number of columns: ";
31      std::cin >> cols;
32
33      if (rows <= 0 || cols <= 0) {
34          std::cout << "Invalid matrix dimensions. Please enter positive values
                for rows and columns.\n";
35          return 1;
36      }
37
38      std::cout << "Enter the elements of the matrix:\n";
39      for (int i = 0; i < rows; i++) {
40          for (int j = 0; j < cols; j++) {
41              std::cin >> matrix[i][j];
42          }
43      }
44
45      transposeMatrix(matrix, rows, cols);
46
47      return 0;
```

Output

```
/tmp/JvJkzf7zJV.o
Enter the number of rows: 3
Enter the number of columns: 3
Enter the elements of the matrix:
1
3
4
5
6
8
9
2
4
5
Transpose of the matrix:
1 6 2
3 8 4
4 9 5
```

4# Write a C++ program to swap two number by both call by value and call by reference mechanism, using two functions swap_value() and swap_reference respectively, by getting the choice from the user and executing the user's choice by switch-case.

main.cpp

```
1   #include <iostream>
2
3   // Function to swap two numbers using call by value
4   void swap_value(int a, int b) {
5       int temp = a;
6       a = b;
7       b = temp;
8       std::cout << "Swapped values using Call by Value: " << a << " " << b << std
            ::endl;
9   }
10
11  // Function to swap two numbers using call by reference
12  void swap_reference(int &a, int &b) {
13      int temp = a;
14      a = b;
15      b = temp;
16      std::cout << "Swapped values using Call by Reference: " << num1 << " " <<
            num2 << std::endl;
17  }
18
19  int main() {
20      int num1, num2;
21      int choice;
22
```

Output

```
/tmp/JvJkzf7zJV.o
Enter two numbers to swap: 3
4
Choose the swapping mechanism:
1. Call by Value
2. Call by Reference
1
Swapped values using Call by Value: 4 3
```

5# Write a C++ program to display all array elements using recursion.

main.cpp

```
1   #include <iostream>
2
3   void displayArray(int arr[], int size, int index) {
4       if (index >= size) {
5           return; // Base case: stop when the index is out of bounds
6       }
7
8       // Print the element at the current index
9       std::cout << arr[index] << " ";
10
11      // Recursively call the function for the next index
12      displayArray(arr, size, index + 1);
13  }
14
15  int main() {
16      int arr[] = {1, 2, 3, 4, 5};
17      int size = sizeof(arr) / sizeof(arr[0]);
18
19      std::cout << "Array elements using recursion: ";
20      displayArray(arr, size, 0); // Start from index 0
21
22      return 0;
23  }
```

Output

```
/tmp/JvJkzf7zJV.o
Array elements using recursion: 1 2 3 4 5
```

6# Write a C++ program to find sum of elements of array using recursion.

main.cpp

Run

Output

```cpp
#include <iostream>

int sumOfArray(int arr[], int size, int index) {
    if (index >= size) {
        return 0; // Base case: when the index is out of bounds, return 0
    }

    // Add the current element to the sum of the rest of the array
    return arr[index] + sumOfArray(arr, size, index + 1);
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);

    int sum = sumOfArray(arr, size, 0);

    std::cout << "Sum of elements in the array: " << sum << std::endl;

    return 0;
}
```

```
/tmp/JvJkzf7zJV.o
Sum of elements in the array: 15
```

7# Write a C++ program to find maximum and minimum elements in array using recursion.

main.cpp

Run

Output

```cpp
#include <iostream>

// Function to find the maximum element in an array using recursion
int findMax(int arr[], int size) {
    if (size == 1) {
        return arr[0];  // Base case: if there's only one element, it's the maximum
    }

    int maxRest = findMax(arr, size - 1);  // Find the maximum in the rest of the array

    return (maxRest > arr[size - 1]) ? maxRest : arr[size - 1];
}

// Function to find the minimum element in an array using recursion
int findMin(int arr[], int size) {
    if (size == 1) {
        return arr[0];  // Base case: if there's only one element, it's the minimum
    }

    int minRest = findMin(arr, size - 1);  // Find the minimum in the rest of the array
```

```
/tmp/JvJkzf7zJV.o
Maximum element in the array: 9
Minimum element in the array: 1
```

Week 10

1# Write a C++ program that uses functions to perform the following operations:

i. Reading a complex number
ii. ii. Writing a complex number
iii. iii. Addition and subtraction of two complex numbers iv. Multiplication of two complex numbers. Note: represent complex number using a structure.

main.cpp

```cpp
#include <iostream>

using namespace std;

// Structure to represent a complex number
struct Complex {
    double real;
    double imag;
};

// Function to read a complex number
void readComplexNumber(Complex &num) {
    cout << "Enter real part: ";
    cin >> num.real;
    cout << "Enter imaginary part: ";
    cin >> num.imag;
}

// Function to write a complex number
void writeComplexNumber(const Complex &num) {
    cout << "Complex Number: " << num.real << " + " << num.imag << "i" << endl;
}

// Function to add two complex numbers
```

Output

```
/tmp/JvJkzf7zJV.o
Enter details for the first complex number:
Enter real part: 2
Enter imaginary part: 3
Enter details for the second complex number:
Enter real part:
5
Enter imaginary part: 7
Sum of the two complex numbers:
Complex Number: 7 + 10i

Difference of the two complex numbers:
Complex Number: -3 + -4i

Product of the two complex numbers:
Complex Number: -11 + 29i
```

2# Write a C++ program to compute the monthly pay of 100 employees using each employee's name, basic pay. The DA is computed as 52% of the basic pay. Gross-salary (basic pay + DA). Print the employees name and gross salary

main.cpp

```cpp
double computeDA(double basicPay) {
    return 0.52 * basicPay;
}

// Function to compute the gross salary
double computeGrossSalary(double basicPay, double da) {
    return basicPay + da;
}

// Function to print employee details and gross salary
void printEmployeeDetails(const Employee &employee, double grossSalary) {
    cout << "Employee Name: " << employee.name << "\tGross Salary: " <<
        grossSalary << endl;
}

int main() {
    const int numEmployees = 3;
    Employee employees[numEmployees];

    // Input basic pay and name for each employee
    for (int i = 0; i < numEmployees; ++i) {
        cout << "Enter details for Employee " << i + 1 << ":\n";
        cout << "Name: ";
        getline(cin >> ws, employees[i].name); // Use getline to allow spaces
```

Output

```
/tmp/JvJkzf7zJV.o
Enter details for Employee 1:
Name: Manas
Basic Pay: 90000
Enter details for Employee 2:
Name: Anamika
Basic Pay: 55000
Enter details for Employee 3:
Name: Arushi
Basic Pay: 60000
Employee Name: Manas    Gross Salary: 136800
Employee Name: Anamika  Gross Salary: 83600
Employee Name: Arushi   Gross Salary: 91200
```

3# Create a Book structure containing book_id, title, author name and price. Write a C++ program to pass a structure as a function argument and print the book details.

main.cpp

```cpp
int main() {
    // Create a Book structure
    Book myBook;

    // Input book details
    cout << "Enter Book ID: ";
    cin >> myBook.book_id;

    cout << "Enter Title: ";
    getline(cin >> ws, myBook.title); // Allowing spaces in the title using
        getline

    cout << "Enter Author: ";
    getline(cin >> ws, myBook.author); // Allowing spaces in the author's name
        using getline

    cout << "Enter Price: $";
    cin >> myBook.price;

    // Print book details using the function
    cout << "\nBook Details:\n";
    printBookDetails(myBook);

    return 0;
```

Output

```
/tmp/9z3nIl5i5k.o
Enter Book ID: 1
Enter Title: Harry Potter
Enter Author: J K Rowling
Enter Price: $29.9
Book Details:
Book ID: 1
Title: Harry Potter
Author: J K Rowling
Price: $29.9
```

4# Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip. Write a C++ program to display your present address

main.cpp

Run

Output

```cpp
#include <iostream>
#include <cstring> // for strcpy function

using namespace std;

// Define a union containing C-style strings
union Address {
    char name[50];
    char home_address[100];
    char hostel_address[100];
    char city[50];
    char state[50];
    char zip[10];
};

int main() {
    // Create an instance of the Address union
    Address myAddress;
```

```
/tmp/yIoS1EI6WJ.o
Present Address:
Name: 12345
Home Address: 12345
Hostel Address: 12345
City: 12345
State: 12345
Zip: 12345
```

5# Write a C++ program to define a structure named D.O.B., which contains name, day, month and year. Using the concept of nested structures display your name and date of birth.

main.cpp

Run

Output

```cpp
#include <iostream>
#include <string>

using namespace std;

// Structure to represent date of birth
struct Date {
    int day;
    int month;
    int year;
};

// Structure to represent name and date of birth
struct DOB {
    string name;
    Date birthDate;
};

// Function to display name and date of birth
void displayDOB(const DOB &person) {
    cout << "Name: " << person.name << endl;
    cout << "Date of Birth: " << person.birthDate.day << "/"
         << person.birthDate.month << "/"
         << person.birthDate.year << endl;
```

```
/tmp/yIoS1EI6WJ.o
Enter your name: Manas
Enter your date of birth (day month year): 10 02 2003
Your Information:
Name: Manas
Date of Birth: 10/2/2003
```

Week 11

1# Write a program in C++ to display your name, Branch, Year on to the computer screen without using classes and object. All information should be displayed in the separate line.

main.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

int main() {
    string name, branch, year;
    // Displaying name, branch, and year on separate lines
    cout << "Name: ";
    cin >> name;
    cout << "Branch: ";
    cin >> branch;
    cout << "Year: ";
    cin >>year;

    cout << "Name :"<<name<<"\n";
    cout <<"Branch :"<<branch<<"\n";
    cout <<"Year :"<<year<<"\n";

    return 0;
}
```

Output

```
/tmp/yIoS1EI6WJ.o
Name: Manas
Branch: CS
Year: 2023
Name :Manas
Branch :CS
Year :2023
```

2# Write a menu driven program in C++ to perform all basic arithmetic operation addition, subtraction, multiplication, and division of two given values. Program receives two values and required operation to be performed from the keyboard and display particular result of the required operation.

main.cpp

```cpp
#include <iostream>

using namespace std;

// Function to perform addition
double add(double a, double b) {
    return a + b;
}

// Function to perform subtraction
double subtract(double a, double b) {
    return a - b;
}

// Function to perform multiplication
double multiply(double a, double b) {
    return a * b;
}

// Function to perform division
double divide(double a, double b) {
    if (b != 0) {
        return a / b;
    } else {
        cout << "Error: Division by zero!\n";
```

Output

```
/tmp/yIoS1EI6WJ.o
Enter the first number: 23
Enter the second number: 42
Choose an operation:
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
Enter the operation number (1-4): 1
Result of addition: 65
```

3# Write a menu driven program in C++ that receives 4-digit integer value the keyboard and perform following operations:

i) Reverse of that no.
ii) sum of number with its reverse.
iii) sum of alternative digits (1 digit+3 digit and 2 digit+4 digit)

main.cpp

```cpp
#include <iostream>

using namespace std;

// Function to reverse a 4-digit number
int reverseNumber(int num) {
    int reversedNum = 0;
    while (num > 0) {
        reversedNum = reversedNum * 10 + num % 10;
        num /= 10;
    }
    return reversedNum;
}

// Function to calculate the sum of digits
int sumOfDigits(int num) {
    int sum = 0;
    while (num > 0) {
        sum += num % 10;
        num /= 10;
    }
    return sum;
```

Output

```
/tmp/yIoS1EI6WJ.o
Enter a 4-digit integer: 1234
Choose an operation:
i) Reverse of the number
ii) Sum of number with its reverse
iii) Sum of alternative digits
Enter your choice (i/ii/iii): i
Reverse of 1234 is: 4321
```

4# Write a menu driven program in C++ to receive integer number and convert equivalent binary, octal, hexadecimal number.

main.cpp

Run

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

// Function to convert decimal to binary
string decimalToBinary(int decimal) {
    string binary = "";
    while (decimal > 0) {
        binary = char('0' + decimal % 2) + binary;
        decimal /= 2;
    }
    return binary.empty() ? "0" : binary;
}

// Function to convert decimal to octal
string decimalToOctal(int decimal) {
    string octal = "";
    while (decimal > 0) {
        octal = char('0' + decimal % 8) + octal;
        decimal /= 8;
    }
    return octal.empty() ? "0" : octal;
```

Output

```
/tmp/yIoS1EI6WJ.o
Enter an integer number: 4
Choose a base for conversion:
1. Binary
2. Octal
3. Hexadecimal
Enter your choice (1-3): 1
Binary representation: 100
```

5# Write a menu driven program in C++ to perform all basic arithmetic operation addition, subtraction, multiplication, and division of two given values using function and switch case. Program receives two values and required operation to be performed from the keyboard and display particular result of the required operation.

main.cpp

Run

```cpp
#include <iostream>

using namespace std;

// Function to perform addition
double add(double a, double b) {
    return a + b;
}

// Function to perform subtraction
double subtract(double a, double b) {
    return a - b;
}

// Function to perform multiplication
double multiply(double a, double b) {
    return a * b;
}

// Function to perform division
double divide(double a, double b) {
    if (b != 0) {
        return a / b;
```

Output

```
/tmp/yIoS1EI6WJ.o
Enter the first number: 4
Enter the second number: 2
Choose an operation:
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
Enter the operation number (1-4): 1
Result of addition: 6
```

6# Define a class Bank Account to represent a bank account. Include the following members: Data Members:

o Name of the depositor o Account Number o Type of account o Balance amount in the account Member Functions: o To assign initial value o To deposit an amount o To withdraw an amount after checking

main.cpp

Run

```cpp
#include <iostream>
#include <string>

using namespace std;

class BankAccount {
private:
    string depositorName;
    long long accountNumber;
    char accountType;
    double balance;

public:
    // Constructor to initialize data members
    BankAccount(const string& name, long long accNumber, char accType, double
        initialBalance) {
        depositorName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = initialBalance;
    }

    // Function to deposit an amount
    void deposit(double amount) {
```

Output

```
/tmp/yIoS1EI6WJ.o
Deposit successful. Updated balance: $1500.5
Withdrawal successful. Updated balance: $1299.75
Account Information:
Depositor Name: John Doe
Account Number: 123456789
Account Type: S
Current Balance: $1299.75
```

Week 12

1# Write a program in C++ to demonstrate default constructor. Create a class having two data members in the private section. Define a default constructor to initialize these data members to initial value and display these values with the help of member function

main.cpp

Run

Output

```cpp
#include <iostream>

using namespace std;

class MyClass {
private:
    int num1;
    double num2;

public:
    // Default constructor
    MyClass() {
        // Initialize data members to initial values
        num1 = 10;
        num2 = 5.5;
    }

    // Member function to display data members
    void displayValues() {
        cout << "Value of num1: " << num1 << endl;
        cout << "Value of num2: " << num2 << endl;
    }
};
```

```
/tmp/yIoS1EI6WJ.o
Initialized values using the default constructor:
Value of num1: 10
Value of num2: 5.5
```

2# Write a program in C++ to demonstrate parameterized/constructor overloading constructor. Create a class calculator that contains four data members in it. Initialize data members with different values using parameterized constructor and perform various arithmetic operation over these values and display result on to the computer screen.

main.cpp

Run

Output

```cpp
#include <iostream>

using namespace std;

class Calculator {
private:
    int num1;
    int num2;
    double num3;
    double num4;

public:
    // Parameterized constructor with integer values
    Calculator(int a, int b) : num1(a), num2(b), num3(0), num4(0) {}

    // Parameterized constructor with double values
    Calculator(double c, double d) : num1(0), num2(0), num3(c), num4(d) {}

    // Function to perform addition
    double add() {
        return num1 + num2 + num3 + num4;
    }

    // Function to perform subtraction
```

```
/tmp/yIoS1EI6WJ.o
ERROR!
Arithmetic operations using integer values:
Addition: 15
Subtraction: 5

Arithmetic operations using double values:
Multiplication: 0
Division: Error: Division by zero!
0
```

3# Create a class called Triangle that stores the length of the base and height of a right triangle in two private instance variables. Include a constructor that sets these values. Define two functions. The first is hypo( ), which returns the length of the hypotenuse. The second is area ( ), which returns the area of the triangle

main.cpp

Run

Output

```cpp
#include <iostream>
#include <cmath>

using namespace std;

class Triangle {
private:
    double base;
    double height;

public:
    // Constructor to set the base and height
    Triangle(double baseValue, double heightValue) : base(baseValue), height
        (heightValue) {}

    // Function to calculate and return the length of the hypotenuse
    double hypo() const {
        return sqrt(base * base + height * height);
    }

    // Function to calculate and return the area of the triangle
    double area() const {
        return 0.5 * base * height;
    }
};
```

```
/tmp/yIoS1EI6WJ.o
Hypotenuse length: 5
Area of the triangle: 6
```

4# Create a class for counting the number of objects created and destroyed within various block using constructor and destructors.

main.cpp

Run

```cpp
#include <iostream>

using namespace std;

class ObjectCounter {
private:
    static int objectCount; // Static variable to count the number of objects
    int objectId; // Unique ID for each object

public:
    ObjectCounter() {
        objectId = ++objectCount; // Increment object count and assign ID
        cout << "Object " << objectId << " created." << endl;
    }

    ~ObjectCounter() {
        cout << "Object " << objectId << " destroyed." << endl;
    }

    static int getObjectCount() {
        return objectCount;
    }
};

// Initialize the static variable
```

Output

```
/tmp/yIoS1EI6WJ.o
Object 1 created.
Object 2 created.
Number of objects in block 1: 2
Object 2 destroyed.
Object 1 destroyed.
Object 3 created.
Object 4 created.
Object 5 created.
Number of objects in block 2: 5
Object 5 destroyed.
Object 4 destroyed.
Object 3 destroyed.
Number of objects outside any block: 5
```