Lab 9

Introduction:

In this lab we will analyze Python project dependencies using **pydeps** for python projects to compute the **fan-in** and **fan-out** for each module. We also **detect cycles** in the dependency graph. In this lab we also identify **isolated modules**, which are defined but not used. This analysis aids in improving code organization, reducing coupling, and enhancing modular design.

We will also work with Java to understand how modularity affects object-oriented programs. As part of this, we will use the **LCOM** (Lack of Cohesion in Methods) metric—a measure of how related the methods in a class are to each other. Using a **Java-based LCOM tool**, we will calculate the **LCOM1**, **LCOM2**, **LCOM3**, **LCOM4**, **LCOM5** and **YALCOM** scores of an open-source repository.

Setup and Tools:

1. **Setting up a virtual environment.**

```
sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ python3 -m venv Lab9
sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ source Lab9/bin/activate
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$
```

2. **Installing “pydeps”.**

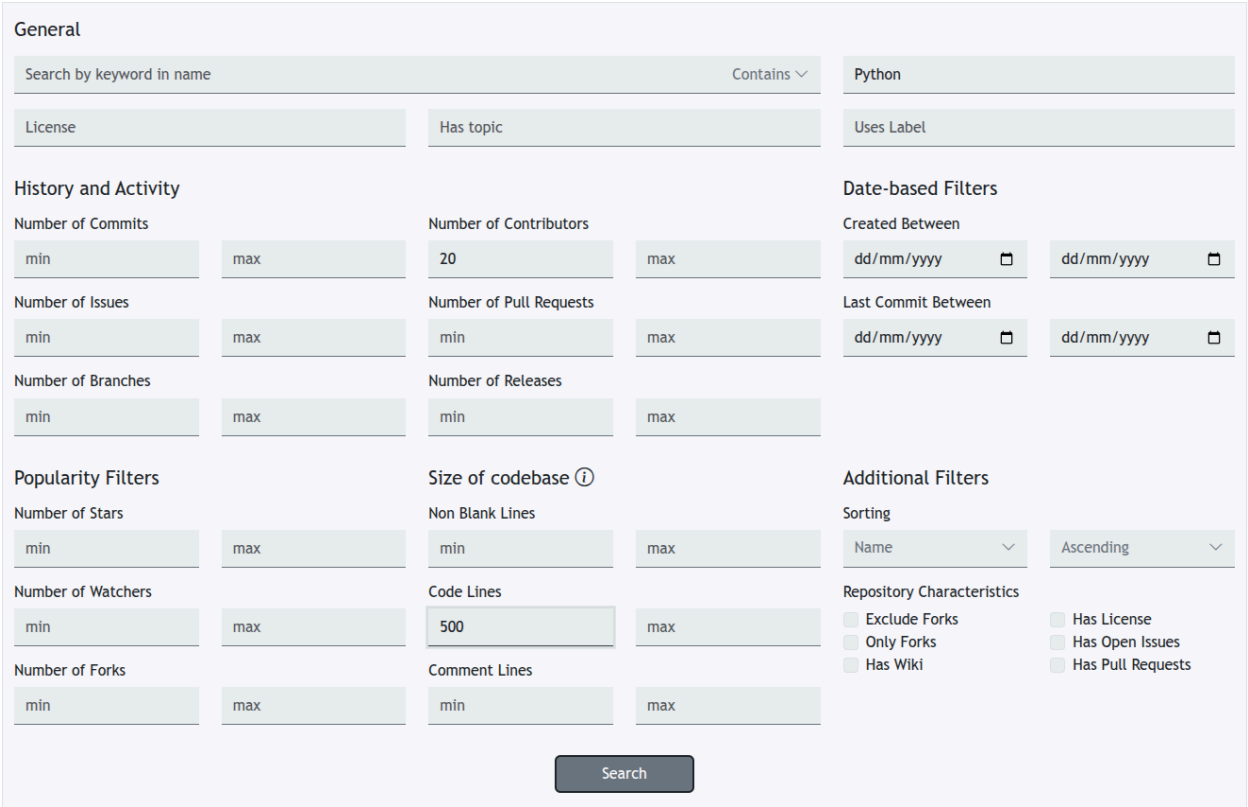
```
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ pip install pydeps
Collecting pydeps
  Downloading pydeps-3.0.1-py3-none-any.whl.metadata (22 kB)
Collecting stdlib_list (from pydeps)
  Downloading stdlib_list-0.11.1-py3-none-any.whl.metadata (3.3 kB)
Downloading pydeps-3.0.1-py3-none-any.whl (47 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 47.6/47.6 kB 7.6 MB/s eta 0:00:00
Downloading stdlib_list-0.11.1-py3-none-any.whl (83 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 83.6/83.6 kB 2.9 MB/s eta 0:00:00
Installing collected packages: stdlib_list, pydeps
Successfully installed pydeps-3.0.1 stdlib_list-0.11.1
```

3. **Installing “java”**

```
sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/LCOM/src$ sudo apt install default-jre
[sudo] password for sumeet:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  acpid blt libnsl2 libtk8.6 mailcap tk8.6-blt2.5
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fonts-dejavu-extra java-common
  libatk-wrapper-java libatk-wrapper-java-jni openjdk-21-jre
  openjdk-21-jre-headless
```

Methodology:

1. Finding a real-world repository using GitHub SEART.



- **Language = Python.**
- **Minimum 20 contributors.**
- **Minimum 500 lines of code.**

2. Selected repository [Link]

3. Running pydeps and storing output in “out.json”

```
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ pydeps --show-deps spe
ctree/ > out.json
```

4. Python script for computing fan-in fan-out

analysis.py U ×

analysis.py > ...

```python
import json

with open('out.json') as f:
    data = json.load(f)

fan_in = {}
fan_out = {}

for module, details in data.items():
    deps = details.get('imports', [])
    fan_out[module] = len(deps)
    for dep in deps:
        fan_in[dep] = fan_in.get(dep, 0) + 1

all_modules = set(fan_in.keys()) | set(fan_out.keys())

print(f"{'Module':<35} | {'Fan-in':>7} | {'Fan-out':>8}")
print("-" * 60)
for module in sorted(all_modules):
    fi = fan_in.get(module, 0)
    fo = fan_out.get(module, 0)
    print(f"{module:<35} | {fi:>7} | {fo:>8}")


with open("fan_analysis.json", "w") as f:
    json.dump({"fan_in": fan_in, "fan_out": fan_out}, f, indent=4)
```

5. **Python script for cycle detection:**

cycle_detection.py

spectree > cycle_detection.py > cycle_detection

```python
import json

with open('out.json') as f:
    data = json.load(f)

cycle = None

def cycle_detection(path, cur_node):

    global cycle

    if cur_node in path:
        cycle = path + [cur_node]
        return True

    if 'imports' not in data[cur_node].keys():
        return False

    for node in data[cur_node]['imports']:
        if cycle_detection(path + [cur_node], node):
            return True

    return False

if cycle_detection([], '__main__'):
    print("Found Cycle")
    print(cycle)
else:
    print("No Cycle found")
```

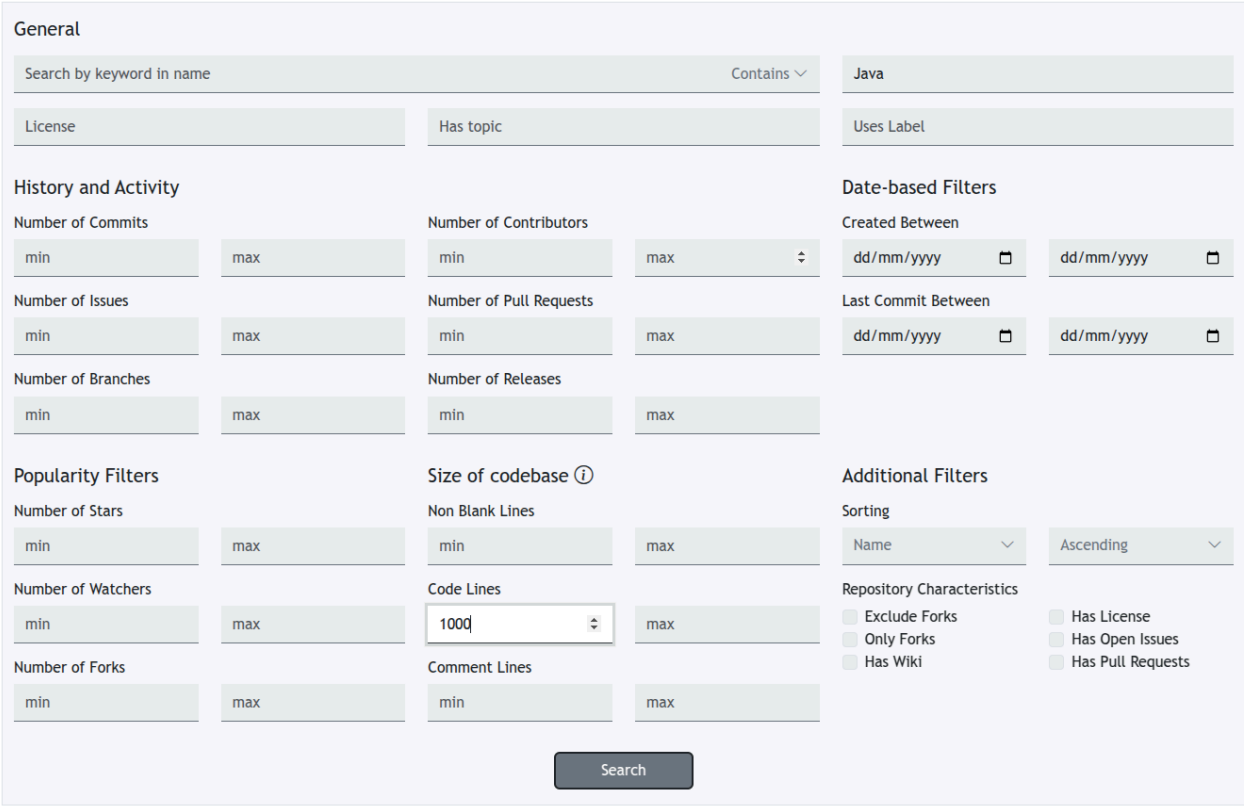
6. Python script for isolated modules

isolated.py U

~/Sumeet/Study/STT/spectree/isolated.py • Untracked

```python
import json
import os

with open('out.json') as f:
    data = json.load(f)

project_root = '/home/sumeet/Sumeet/Study/STT/spectree/spectree'
python_files = []
for root, dirs, files in os.walk(project_root):
    for file in files:
        if file.endswith('.py'):
            python_files.append(os.path.join(root, file))

paths = []
info = data.values()
for i in info:
    paths.append(i['path'])


num_isolated = 0
for file in python_files:
    if file not in paths :
        print(file)
        num_isolated += 1

print("Number of isolated files: ", num_isolated)
```

7. Selecting a JAVA project



- **Language = Java**
- **Min 1000 lines of code**

8. Selected repository

9. Running LCOM.jar

```
sumeet@sumeet-G5-5505:~/Sumeet/Study/STT$ java -jar LCOM.jar -i NocturneSpy-Clie
nt/ -o .
Parsing the source code ...
Resolving symbols...
Computing metrics...
Done.
```

10. **Python script to arrange the packages according to average LCOM value and take the top 5.**

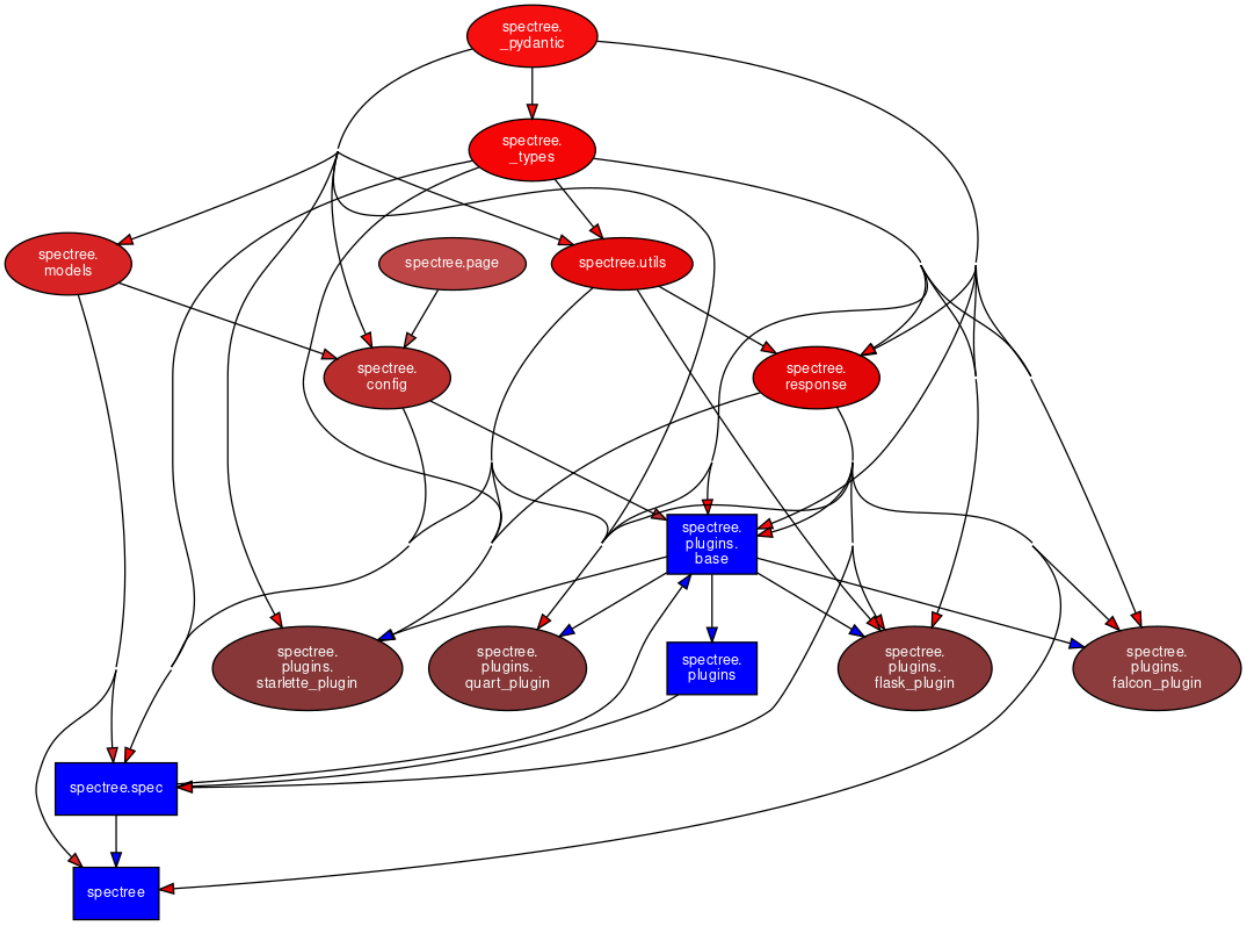
analysis.py

analysis.py > ...

```python
import pandas as pd

data = pd.read_csv('TypeMetrics.csv')

data['average'] = data[['LCOM1', 'LCOM2', 'LCOM3', 'LCOM4', 'LCOM5', 'YALCOM']].mean(axis=1)
data = data.sort_values(by='average', ascending=False, ignore_index=True).head()

print(data)

data.to_csv('TypeMetrics_sorted.csv', index=False)
```

Result and Analysis:

1. Pydeps dependency graph



2. Fan-in Fan-out

```
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ python3 analysis.py
Module                                  |  Fan-in |  Fan-out
------------------------------------------------------------
__main__                                |       0 |       16
spectree                                |       6 |        3
spectree._pydantic                      |      11 |        0
spectree._types                         |       9 |        1
spectree.analysis                       |       1 |        0
spectree.config                         |       3 |        3
spectree.models                         |       4 |        1
spectree.page                           |       2 |        0
spectree.plugins                        |       6 |        1
spectree.plugins.base                   |       6 |        6
spectree.plugins.falcon_plugin          |       1 |        6
spectree.plugins.flask_plugin           |       1 |        7
spectree.plugins.quart_plugin           |       1 |        7
spectree.plugins.starlette_plugin       |       1 |        7
spectree.response                       |       8 |        3
spectree.spec                           |       3 |        6
spectree.utils                          |       6 |        2
```

3. Analyzing the generated dependency graph
   - **Highly coupled modules.**
     "**sprectree._pydantic**" has highest fan-in of 11

```
Module                                  |  Fan-in |  Fan-out
------------------------------------------------------------
spectree._pydantic                      |      11 |        0
```

   - **Cyclic dependencies.**

```
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ python3 cycle_detection.py
Found Cycle
['__main__', 'spectree', 'spectree.spec', 'spectree.plugins', 'spectree.plugins.base',
 'spectree']
```

   - **Isolated modules**

```
(Lab9) sumeet@sumeet-G5-5505:~/Sumeet/Study/STT/spectree$ python3 isolated.py
Number of isolated files:  0
```

4. Modules with high fan-in are at high risk of breaking of system since many modules depend on them.
    - **spectree._pyndatic**
    - **spectree._types**
    - **spectree.response**

5. Output of LCOM.jar

Project Name	Package Name	Type Name	LCOM1	LCOM2	LCOM3	LCOM4	LCOM5	YALCOM
NocturneSpy-Client	com.android.adobot.utils	CalandarUtils	1	1	2	1	1	0
NocturneSpy-Client	com.android.adobot.http	Http	1	0	1	1	0.5	0
NocturneSpy-Client	com.android.adobot.http	HttpRequest	6	2	1	1	0.9375	0
NocturneSpy-Client	com.android.adobot.http	HttpCallback	0	0	1	1	0	-1
NocturneSpy-Client	com.android.adobot	CommonParams	28	20	1	1	0.888888888888889	0
NocturneSpy-Client	com.android.adobot	Constants	0	0	0	0	0	-1
NocturneSpy-Client	com.android.adobot	SMSWatcher	0	0	1	1	0	0
NocturneSpy-Client	com.android.adobot	NetworkWatcher	0	0	1	1	0	0
NocturneSpy-Client	com.android.adobot	CommandService	46	26	5	4	0.681818181818182	0.166666666666667
NocturneSpy-Client	com.android.adobot.activities	BaseActivity	10	0	5	5	0	1
NocturneSpy-Client	com.android.adobot.activities	SetupActivity	2	1	2	1	1	0
NocturneSpy-Client	com.android.adobot.activities	PermissionsActivity	20	19	6	1	0.958333333333333	0
NocturneSpy-Client	com.android.adobot.activities	UpdateActivity	3	0	2	1	0.666666666666667	0
NocturneSpy-Client	com.android.adobot.job	CheckJobCreator	0	0	1	1	0	0
NocturneSpy-Client	com.android.adobot.job	CheckJob	1	0	2	2	1	1
NocturneSpy-Client	com.android.adobot.tasks	GetSmsTask	2	1	2	1	0.5	0
NocturneSpy-Client	com.android.adobot.tasks	GetCallLogsTask	2	1	2	1	0.75	0
NocturneSpy-Client	com.android.adobot.tasks	GetCalandarTask	0	0	2	2	0.833333333333333	0
NocturneSpy-Client	com.android.adobot.tasks	LocationMonitor	22	16	4	1	0.821428571428571	0
NocturneSpy-Client	com.android.adobot.tasks	SendSmsTask	10	5	3	2	0.85	0.333333333333333
NocturneSpy-Client	com.android.adobot.tasks	GetContactsTask	0	0	2	2	0.833333333333333	0
NocturneSpy-Client	com.android.adobot.tasks	TransferBotTask	3	0	2	1	0.8	0
NocturneSpy-Client	com.android.adobot.tasks	BaseTask	3	0	1	1	0.5	0
NocturneSpy-Client	com.android.adobot.tasks	UpdateAppTask	2	1	2	1	0.5	0
NocturneSpy-Client	com.android.adobot.tasks	SmsForwarder	43	31	4	3	0.92	0.272727272727273
NocturneSpy-Client	com.android.adobot.tasks	SmsObserver	9	8	4	3	0.9	0.6
NocturneSpy-Client	com.android.adobot.tasks	SendSmsThread	0	0	1	1	0.5	0

6. Packages with high LCOM values:

Project Name	Package Name	Type Name	LCOM1	LCOM2	LCOM3	LCOM4	LCOM5	YALCOM
java	com.android.adobot.tasks	SmsForwarder	43	31	4	3	0.92	0.272727272727273
java	com.android.adobot	CommandService	46	26	5	4	0.681818181818182	0.166666666666667
java	com.android.adobot	CommonParams	28	20	1	1	0.888888888888889	0
java	com.android.adobot.activities	PermissionsActivity	20	19	6	1	0.958333333333333	0
java	com.android.adobot.tasks	LocationMonitor	22	16	4	1	0.821428571428571	0

Conclusion:

Through this lab, we have explored **fan-in** and **fan-out** analysis, detected **cyclic dependencies**, and identified **isolated or untracked** Python modules. The insights derived from this analysis can guide refactoring efforts, highlight central utility components, and flag tightly coupled or unused code. Overall, this exercise deepens our understanding of software architecture and dependency management, which are key to scalable and maintainable codebases.

By using **LCOM** analysis in Java, we deepened our understanding of internal class design and the significance of maintaining strong cohesion within classes. Overall, the lab highlighted the importance of both **inter-module** and **intra-module** analysis in software engineering, emphasizing clean architecture and modular design principles for building robust applications.

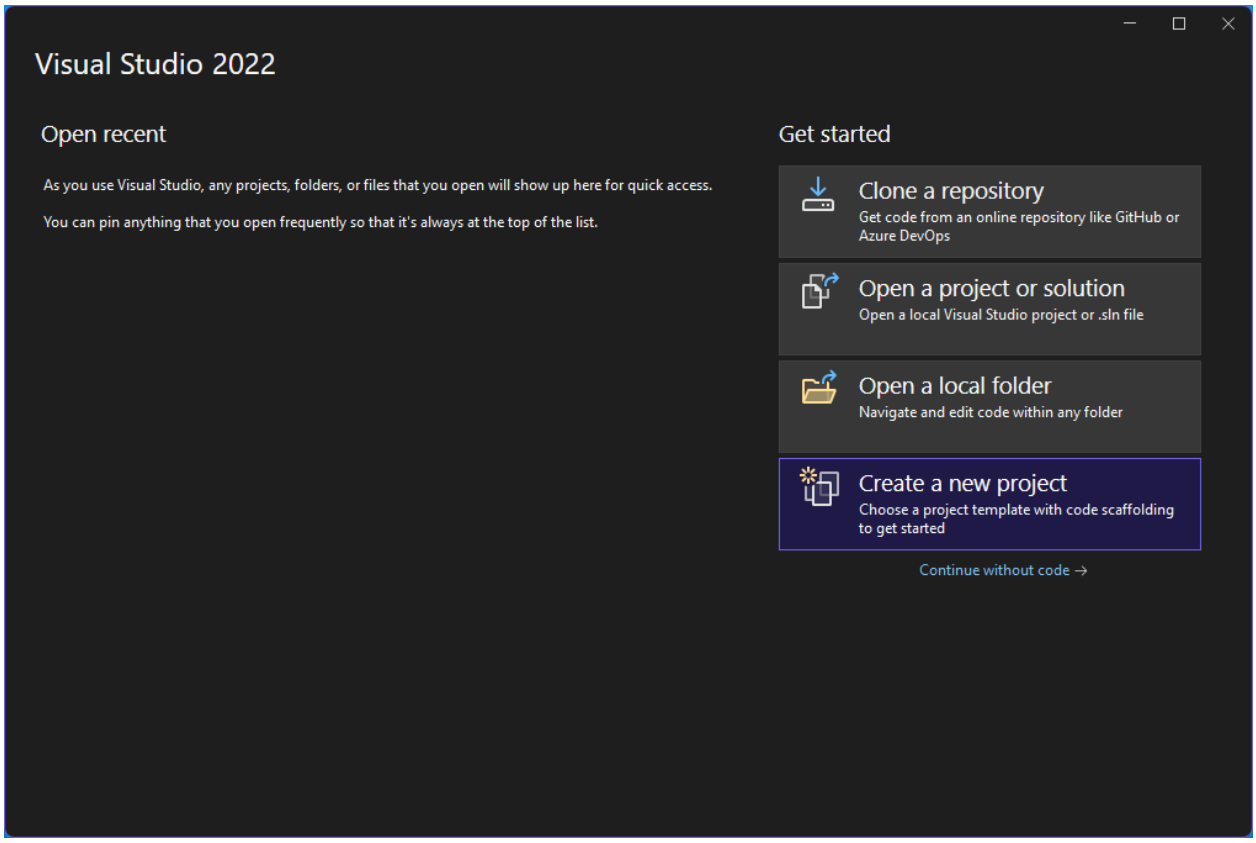
Lab 10

Introduction:

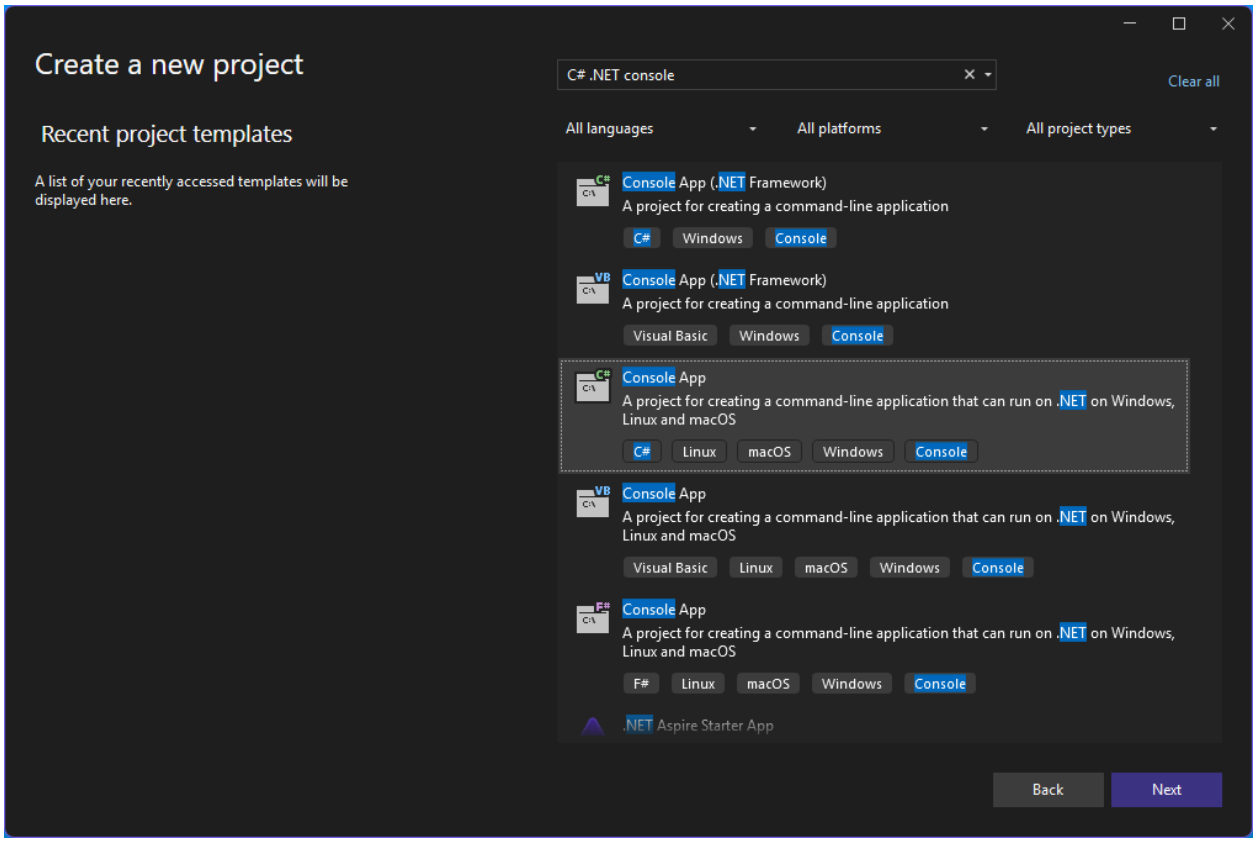
In this lab we will use **.NET** framework and **C#** language. The primary aim is to learn how to use **Visual Studio** and its **debugger**. By creating and executing console-based applications, we will learn about **loops, functions, object-oriented programming,** and **exception handling**.

Tools and Setup:

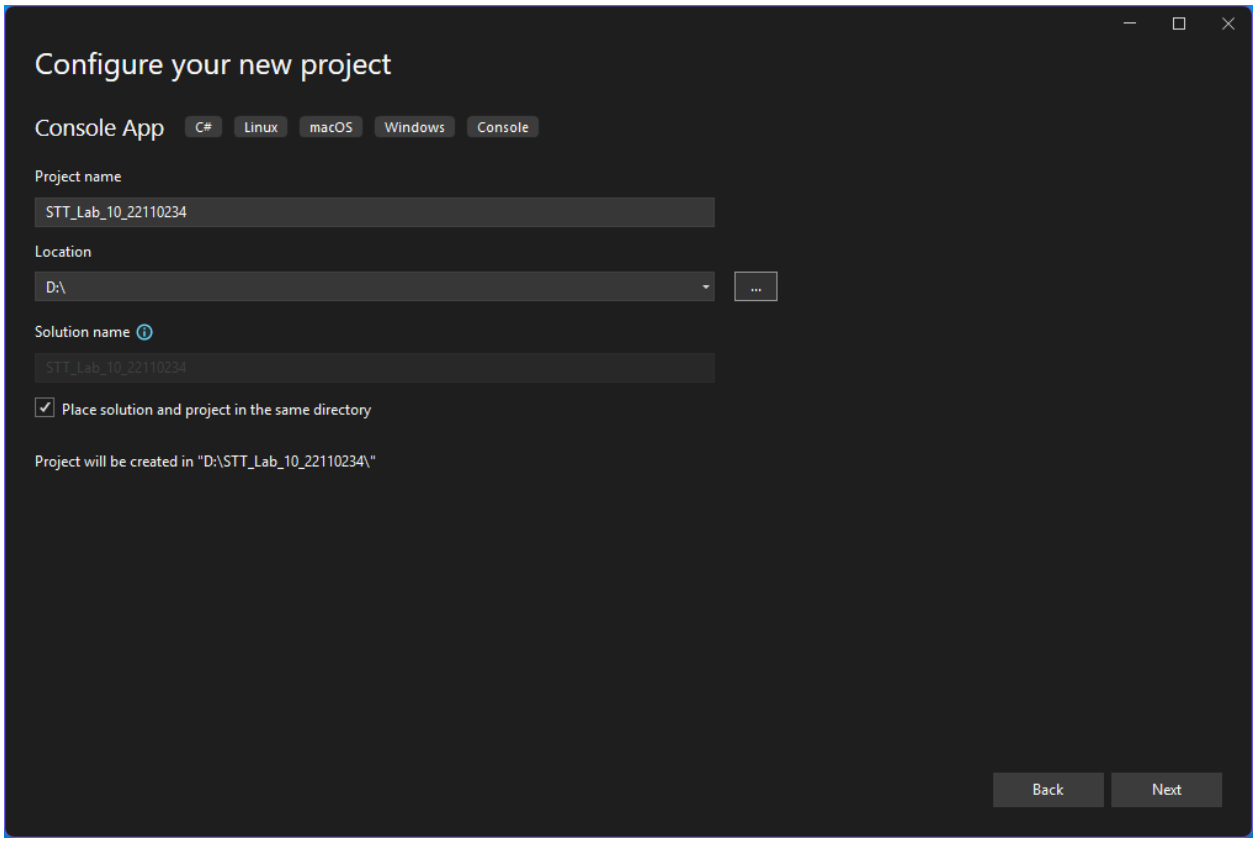
1. **Creating a Visual Studio Project**



2. Searching for .NET console app in C#

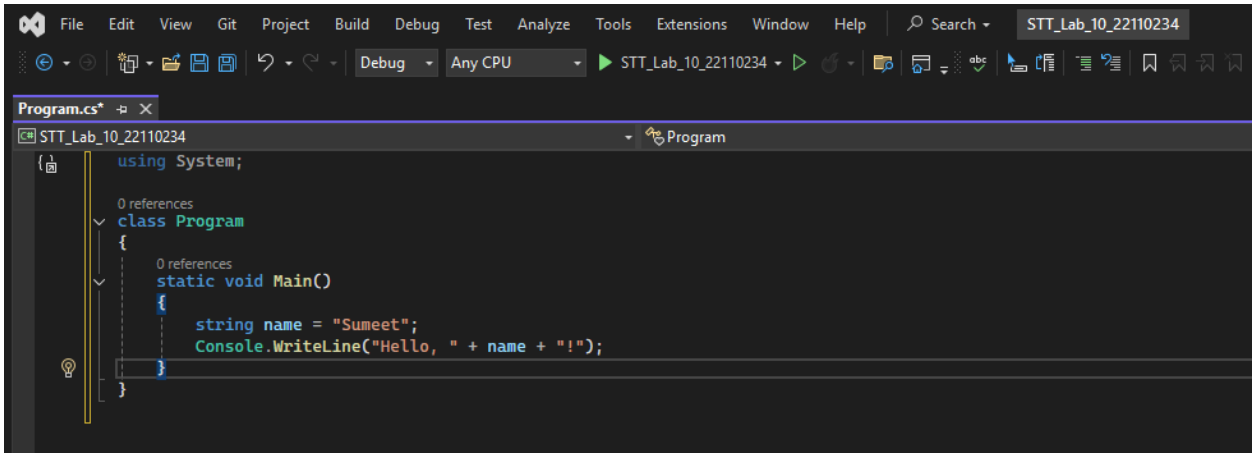


3. Creating the project

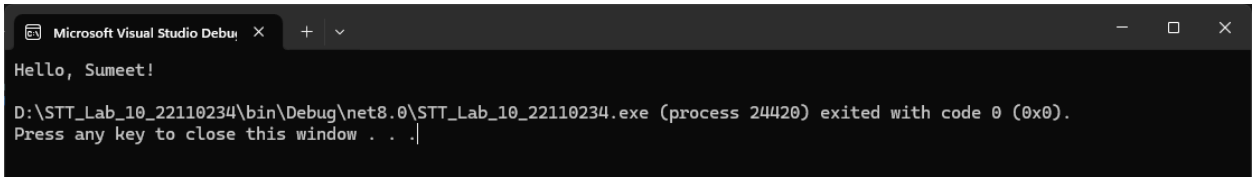


Methodology:

1. Writing a simple C# program



2. Executing the program



3. Writing C# program for basic calculator

```
using System;

0 references
class Program
{
    1 reference
    static void PerformOperations(double num1, double num2)
    {
        double sum = num1 + num2;
        double difference = num1 - num2;
        double product = num1 * num2;
        double quotient = num1 / num2;

        Console.WriteLine("Addition: " + sum);
        Console.WriteLine("Subtraction: " + difference);
        Console.WriteLine("Multiplication: " + product);
        Console.WriteLine("Division: " + quotient);

        if (sum % 2 == 0)
        {
            Console.WriteLine("The sum is even.");
        }
        else
        {
            Console.WriteLine("The sum is odd.");
        }
    }

    0 references
    static void Main()
    {
        Console.Write("Enter the first number: ");
        double number1 = Convert.ToDouble(Console.ReadLine());

        Console.Write("Enter the second number: ");
        double number2 = Convert.ToDouble(Console.ReadLine());

        PerformOperations(number1, number2);
        Console.WriteLine("Done");
    }
}
```

4. Executing the calculator program

```
Microsoft Visual Studio Debu
Enter the first number: 2
Enter the second number: 2
Addition: 4
Subtraction: 0
Multiplication: 4
Division: 1
The sum is even.
Done

D:\STT_Lab_10_22110234\bin\Debug\net8.0\STT_Lab_10_22110234.exe (process 10124) exited with code 0 (0x0).
Press any key to close this window . . .
```

5. Adding breakpoints

```
0 references
static void Main()
{
    Console.Write("Enter the first number: ");
    double number1 = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the second number: ");
    double number2 = Convert.ToDouble(Console.ReadLine());

    PerformOperations(number1, number2);
    Console.WriteLine("Done");
}
}
```

6. Running in debug mode

```
D:\STT_Lab_10_22110234\bin\I
Enter the first number: 4
Enter the second number: 2
```

Program

Use the dropdown to view and navigate to other items in this file.

7. Implementing loops and functions

```
using System;

0 references
class Program
{
    1 reference
    public static void PrintNumbers()
    {
        for (int i = 1; i <= 10; i++)
        {
            Console.WriteLine(i);
        }
    }
    1 reference
    public static long CalculateFactorial(int number)
    {
        long factorial = 1;
        for (int i = 1; i <= number; i++)
        {
            factorial *= i;
        }
        return factorial;
    }
    0 references
    static void Main()
    {
        Console.WriteLine("Numbers from 1 to 10:");
        PrintNumbers();

        string userInput;
        do
        {
            Console.WriteLine("\nEnter a number to calculate its factorial or type 'exit' to quit:");
            userInput = Console.ReadLine().Trim();

            if (userInput.ToLower() != "exit")
            {
                if (int.TryParse(userInput, out int number))
                {
```

```
                    if (number >= 0)
                    {
                        long result = CalculateFactorial(number);
                        Console.WriteLine($"Factorial of {number} is {result}");
                    }
                    else
                    {
                        Console.WriteLine("Please enter a non-negative integer.");
                    }
                }
                else
                {
                    Console.WriteLine("Invalid input. Please enter a valid number.");
                }
            }
        }
        while (userInput.ToLower() != "exit");

        Console.WriteLine("Goodbye!");
    }
}
```

8. Adding breakpoints

```
0 references
static void Main()
{
    Console.WriteLine("Numbers from 1 to 10:");
    PrintNumbers();

    string userInput;
    do
    {
        Console.WriteLine("\nEnter a number to calculate its factorial or type 'exit' to quit:");
        userInput = Console.ReadLine().Trim();

        if (userInput.ToLower() != "exit")
        {
            if (int.TryParse(userInput, out int number))
            {
                if (number >= 0)
                {
                    long result = CalculateFactorial(number);
                    Console.WriteLine($"Factorial of {number} is {result}");
                }
```

9. Object-oriented programming

```
using System;

6 references
class Student
{
    4 references
    public string Name { get; set; }
    4 references
    public int ID { get; set; }
    8 references
    public double Marks { get; set; }

    2 references
    public Student(string name, int id, double marks)
    {
        Name = name;
        ID = id;
        Marks = marks;
    }

    2 references
    public string GetGrade()
    {
        if (Marks >= 90)
        {
            return "A";
        }
        else if (Marks >= 80)
        {
            return "B";
        }
        else if (Marks >= 70)
        {
            return "C";
        }
        else if (Marks >= 60)
        {
            return "D";
        }
        else
        {
            return "F";
        }
    }
```

```
        1 reference
        public void DisplayDetails()
    {
        Console.WriteLine("Student Name: " + Name);
        Console.WriteLine("Student ID: " + ID);
        Console.WriteLine("Student Marks: " + Marks);
        Console.WriteLine("Student Grade: " + GetGrade());
    }
}
```

```
3 references
class StudentIITGN : Student
{
    2 references
    public string Hostel_Name_IITGN { get; set; }

    1 reference
    public StudentIITGN(Student student, string hostelName)
        : base(student.Name, student.ID, student.Marks)
    {
        Hostel_Name_IITGN = hostelName;
    }

    1 reference
    public void DisplayIITGNDetails()
    {
        Console.WriteLine("Student Name: " + Name);
        Console.WriteLine("Student ID: " + ID);
        Console.WriteLine("Student Marks: " + Marks);
        Console.WriteLine("Student Grade: " + GetGrade());
        Console.WriteLine("Hostel Name: " + Hostel_Name_IITGN);
    }
}
```

```
0 references
class Program
{
    0 references
    static void Main()
    {
        Student student = new Student("Sumeet", 22110234, 95);

        StudentIITGN studentIITGN = new StudentIITGN(student, "Emiet Hostel");

        Console.WriteLine("Displaying student details");
        student.DisplayDetails();

        Console.WriteLine("\nDisplaying IITGN student details");
        studentIITGN.DisplayIITGNDetails();
    }
}
```

10. Executing the program

```
Microsoft Visual Studio Debu

Displaying student details
Student Name: Sumeet
Student ID: 22110234
Student Marks: 95
Student Grade: A

Displaying IITGN student details
Student Name: Sumeet
Student ID: 22110234
Student Marks: 95
Student Grade: A
Hostel Name: Emiet Hostel

D:\STT_Lab_10_22110234\bin\Debug\net8.0\STT_Lab_10_22110234.exe (process 23084) exited with code 0 (0x0).
Press any key to close this window . . .
```

11. Adding breakpoints

```
0 references
class Program
{
    0 references
    static void Main()
    {
        Student student = new Student("Sumeet", 22110234, 95);

        StudentIITGN studentIITGN = new StudentIITGN(student, "Emiet Hostel");

        Console.WriteLine("Displaying student details");
        student.DisplayDetails();

        Console.WriteLine("\nDisplaying IITGN student details");
        studentIITGN.DisplayIITGNDetails();
    }
}
```

12. Exception handling

```
using System;

0 references
class Program
{
    1 reference
    static void PerformOperations(double num1, double num2)
    {
        try
        {
            double sum = num1 + num2;
            double difference = num1 - num2;
            double product = num1 * num2;
            double quotient = num1 / num2;

            Console.WriteLine("Addition: " + sum);
            Console.WriteLine("Subtraction: " + difference);
            Console.WriteLine("Multiplication: " + product);
            Console.WriteLine("Division: " + quotient);

            if (sum % 2 == 0)
            {
                Console.WriteLine("The sum is even.");
            }
            else
            {
                Console.WriteLine("The sum is odd.");
            }
        }

        catch (DivideByZeroException)
        {
            Console.WriteLine("Cannot divide by zero");
        }
    }
```

```
    0 references
    static void Main()
    {
        Console.Write("Enter the first number: ");
        double number1 = Convert.ToDouble(Console.ReadLine());

        Console.Write("Enter the second number: ");
        double number2 = Convert.ToDouble(Console.ReadLine());

        PerformOperations(number1, number2);
    }
}
```

13. Executing the program

Results and Analysis:

Basic calculator program

1. Step-over takes us to next line

```
PerformOperations(number1, number2);
Console.WriteLine("Done");    ≤ 3ms elapsed
}
```

2. Step-into takes us to the definition of the function

```
static void PerformOperations(double num1, double num2)
{    ≤ 1ms elapsed
    double sum = num1 + num2;
    double difference = num1 - num2;
    double product = num1 * num2;
    double quotient = num1 / num2;

    Console.WriteLine("Addition: " + sum);
    Console.WriteLine("Subtraction: " + difference);
    Console.WriteLine("Multiplication: " + product);
    Console.WriteLine("Division: " + quotient);

    if (sum % 2 == 0)
    {
        Console.WriteLine("The sum is even.");
    }
    else
    {
        Console.WriteLine("The sum is odd.");
    }
}

0 references
static void Main()
{
    Console.Write("Enter the first number: ");
    double number1 = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the second number: ");
    double number2 = Convert.ToDouble(Console.ReadLine());

    PerformOperations(number1, number2);
    Console.WriteLine("Done");
}
```

3. Step-out takes us out of the function execution and to the next line

```
            PerformOperations(number1, number2);    ≤ 2ms elapsed
            Console.WriteLine("Done");
        }
    }
```

Loops and functions program

1. Step-into takes us to the start of function

```
    public static void PrintNumbers()
    {    ≤ 1ms elapsed
        for (int i = 1; i <= 10; i++)
        {
            Console.WriteLine(i);
        }
    }
    1 reference
    public static long CalculateFactorial(int number)
    {
        long factorial = 1;
        for (int i = 1; i <= number; i++)
        {
            factorial *= i;
        }
        return factorial;
    }
    0 references
    static void Main()
    {
        Console.WriteLine("Numbers from 1 to 10:");
        PrintNumbers();

        string userInput;
        do
        {
            Console.WriteLine("\nEnter a number to calculate its factorial or type 'exit' to quit:");
            userInput = Console.ReadLine().Trim();

            if (userInput.ToLower() != "exit")
            {
                if (int.TryParse(userInput, out int number))
                {
                    if (number >= 0)
                    {
                        long result = CalculateFactorial(number);
                        Console.WriteLine($"Factorial of {number} is {result}");
```

2. Step-out takes us out of the function call

```
Console.WriteLine("Numbers from 1 to 10:");
PrintNumbers();    ≤ 2ms elapsed

string userInput;
do
{
    Console.WriteLine("\nEnter a number to calculate its factorial or type 'exit' to quit:");
    userInput = Console.ReadLine().Trim();

    if (userInput.ToLower() != "exit")
    {
        if (int.TryParse(userInput, out int number))
        {
            if (number >= 0)
            {
                long result = CalculateFactorial(number);
                Console.WriteLine($"Factorial of {number} is {result}");
```

3. Step-over skips the function execution and take us to the next line

```
{
    Console.WriteLine("Numbers from 1 to 10:");
    PrintNumbers();

    string userInput;
    do
    {
        Console.WriteLine("\nEnter a number to calculate its factorial or type 'exit' to quit:");
        userInput = Console.ReadLine().Trim();

        if (userInput.ToLower() != "exit")
        {
            if (int.TryParse(userInput, out int number))
            {
                if (number >= 0)
                {
                    long result = CalculateFactorial(number);
                    Console.WriteLine($"Factorial of {number} is {result}");    ≤ 1ms elapsed
                }
                else
                {
                    Console.WriteLine("Please enter a non-negative integer.");
                }
```

Object-oriented program

1. Step-into takes us to the start of the function

```
1 reference
public void DisplayDetails()
{   ≤1ms elapsed
    Console.WriteLine("Student Name: " + Name);
    Console.WriteLine("Student ID: " + ID);
    Console.WriteLine("Student Marks: " + Marks);
    Console.WriteLine("Student Grade: " + GetGrade());
}
}

3 references
class StudentIITGN : Student
{
    2 references
    public string Hostel_Name_IITGN { get; set; }

    1 reference
    public StudentIITGN(Student student, string hostelName)
        : base(student.Name, student.ID, student.Marks)
    {
        Hostel_Name_IITGN = hostelName;
    }

    1 reference
```

2. Step-out takes us out of the function execution

```
0 references
class Program
{
    0 references
    static void Main()
    {
        Student student = new Student("Sumeet", 22110234, 95);

        StudentIITGN studentIITGN = new StudentIITGN(student, "Emiet Hostel");

        Console.WriteLine("Displaying student details");
        student.DisplayDetails();   ≤ 2ms elapsed

        Console.WriteLine("\nDisplaying IITGN student details");
        studentIITGN.DisplayIITGNDetails();
    }
}
```

3. Step-over takes us to the next breakpoint

```
0 references
class Program
{
    0 references
    static void Main()
    {
        Student student = new Student("Sumeet", 22110234, 95);

        StudentIITGN studentIITGN = new StudentIITGN(student, "Emiet Hostel");

        Console.WriteLine("Displaying student details");
        student.DisplayDetails();

        Console.WriteLine("\nDisplaying IITGN student details");
        studentIITGN.DisplayIITGNDetails();    ≤ 1ms elapsed
    }
}
```

Exception handling program

1. Step-into takes us to the function definition

```
static void PerformOperations(double num1, double num2)
{   ≤ 1ms elapsed
    try
    {
        double sum = num1 + num2;
        double difference = num1 - num2;
        double product = num1 * num2;
        double quotient = num1 / num2;

        Console.WriteLine("Addition: " + sum);
        Console.WriteLine("Subtraction: " + difference);
        Console.WriteLine("Multiplication: " + product);
        Console.WriteLine("Division: " + quotient);

        if (sum % 2 == 0)
        {
            Console.WriteLine("The sum is even.");
        }
        else
        {
            Console.WriteLine("The sum is odd.");
        }
    }

    catch (DivideByZeroException)
    {
        Console.WriteLine("Cannot divide by zero");
    }
}

0 references
static void Main()
{
    Console.Write("Enter the first number: ");
    double number1 = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the second number: ");
    double number2 = Convert.ToDouble(Console.ReadLine());

    PerformOperations(number1, number2);
}
}
```

2. Step-out takes us out of the function execution

```
0 references
static void Main()
{
    Console.Write("Enter the first number: ");
    double number1 = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the second number: ");
    double number2 = Convert.ToDouble(Console.ReadLine());

    PerformOperations(number1, number2);    ≤ 2ms elapsed
}
}
```

3. Step-over takes us to the next line

```
0 references
static void Main()
{
    Console.Write("Enter the first number: ");
    double number1 = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the second number: ");
    double number2 = Convert.ToDouble(Console.ReadLine());

    PerformOperations(number1, number2);
}    ≤ 2ms elapsed
}
```

Conclusion:

In this lab, we were first introduced to the **C#** language and **Visual Studio debugger**. We explored the syntax of **C#** and learned about **object-oriented** design. We also did **exception-handling** such as **invalid input** and **division by zero** error**.** We used **Visual Studio's debugging tools** for **step-by-step** code analysis. We looked at **step-in, step-out** and **step-over.**