Other Language report:

Name: S M Ragib Rezwan

ID: 103172423

Object Oriented Program is a paradigm used by several languages including C#, C++, Java, Rust, Python, etc. But even though they use the same concepts to fulfill the OOP paradigms in their code, they end up doing so in slightly different ways. Thus, in this report, I will look into the different ways C# and Python fulfill the OOP paradigms.

Comparison between the languages:

	C#	Python
Paradigm it follows	Object Oriented only	Object Oriented and Procedural
Unwanted data removal from memory	Automatic garbage collector	Automatic garbage collector
Syntax comparison	Need to declare variables before using them Must end every line with semicolon	No need to declare variables before using them No need to end every line with semicolon
Language type	Statically-typed, compiled (build)	Dynamically interpreted
Code reading aspect	Consistent syntax	Human readable, indented (ie the whitespaces)
Time taken to learn	Slower	Faster

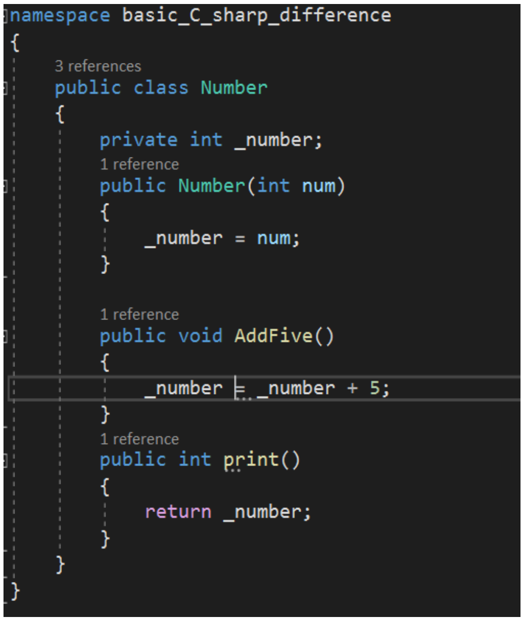
Key difference between them:

After considering all the possible differences, the core difference that stands out between them is the type of language they are!

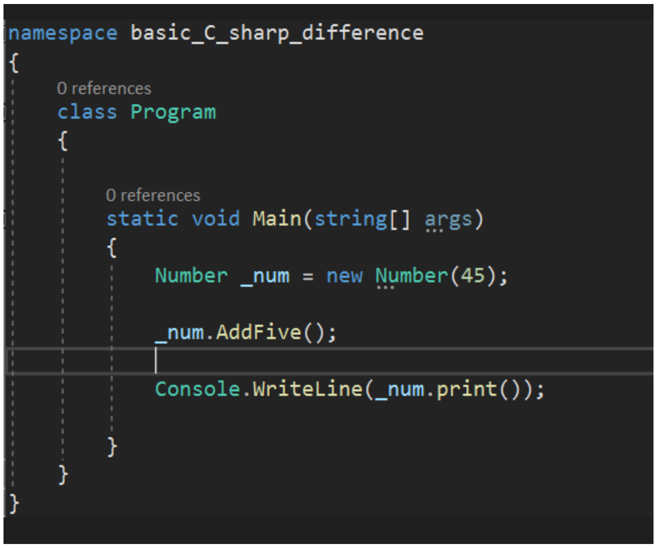
C#:

C# is a statically typed language. This means in order for us to use a variable, we must first declare what type of data it can accept. This variable can be in the form of parameter in the field, a parameter passed to a method or class constructor, a return variable, etc.

Otherwise, the program will return an error at compile time and stop the code from compiling!

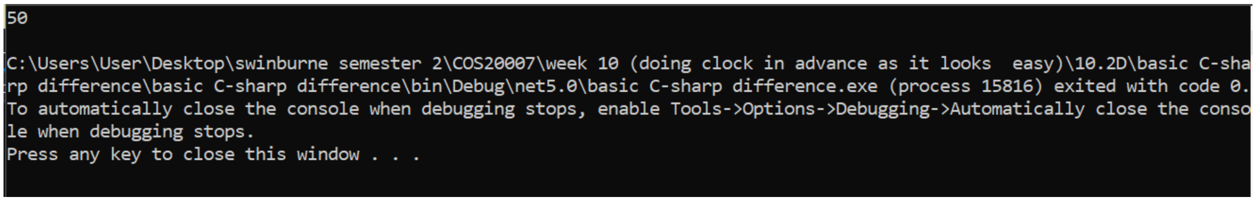


```
namespace basic_C_sharp_difference
{
    3 references
    public class Number
    {
        private int _number;
        1 reference
        public Number(int num)
        {
            _number = num;
        }

        1 reference
        public void AddFive()
        {
            _number = _number + 5;
        }
        1 reference
        public int print()
        {
            return _number;
        }
    }
}
```



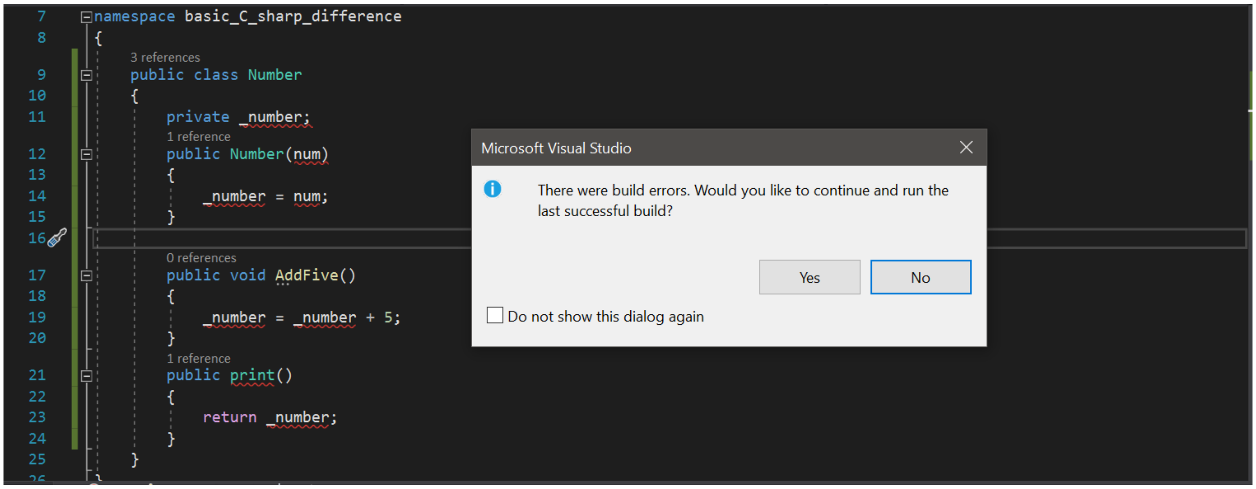
```
namespace basic_C_sharp_difference
{
    0 references
    class Program
    {

        0 references
        static void Main(string[] args)
        {
            Number _num = new Number(45);

            _num.AddFive();

            Console.WriteLine(_num.print());

        }
    }
}
```

(When we declare the datatype of the variable declared in field, datatype of the variable given as the parameter to the method and constructor, datatype of return variable, etc before using it, it compiles and runs properly)



```
50

C:\Users\User\Desktop\swinburne semester 2\COS20007\week 10 (doing clock in advance as it looks  easy)\10.2D\basic C-sha
rp difference\basic C-sharp difference\bin\Debug\net5.0\basic C-sharp difference.exe (process 15816) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

(When we don't declare the type, the code doesn't even compile)



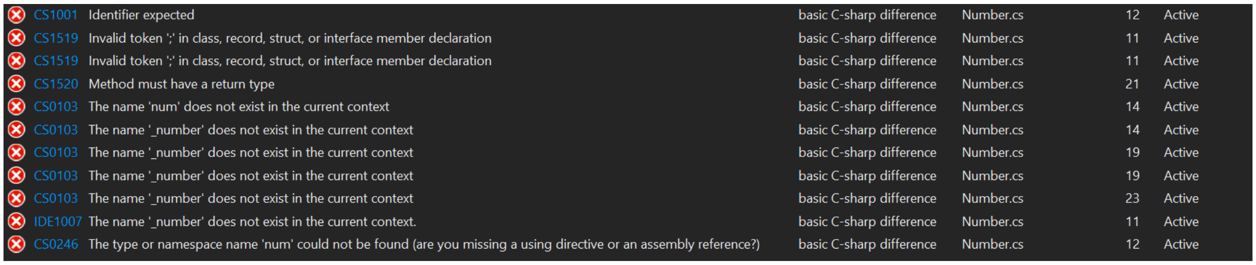
```
namespace basic_C_sharp_difference
{
    3 references
    public class Number
    {
        private _number;
        1 reference
        public Number(num)
        {
            _number = num;
        }

        0 references
        public void AddFive()
        {
            _number = _number + 5;
        }
        1 reference
        public print()
        {
            return _number;
        }
    }
```

Microsoft Visual Studio

There were build errors. Would you like to continue and run the last successful build?

□ Do not show this dialog again

(and instead, it shows the following errors where it doesn't know what number is and thus can't do anything using it)

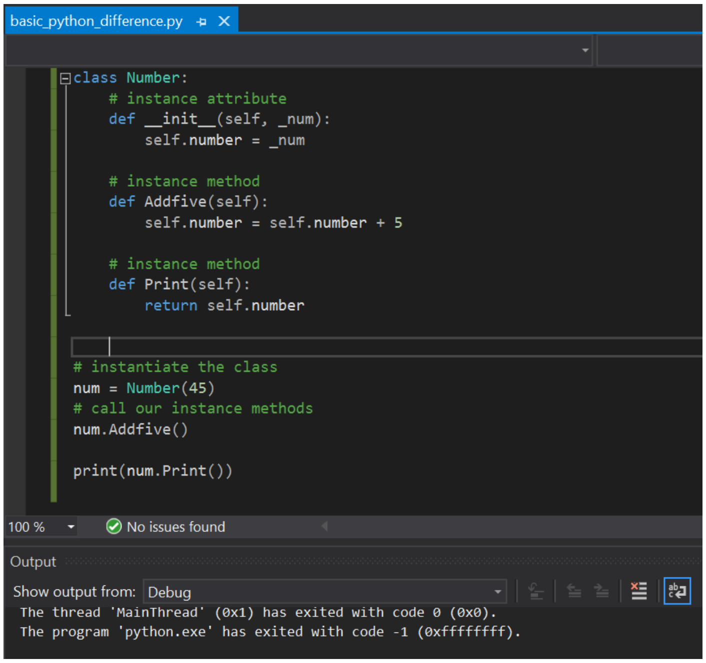


CS1001	Identifier expected	basic C-sharp difference	Number.cs	12	Active
CS1519	Invalid token ';' in class, record, struct, or interface member declaration	basic C-sharp difference	Number.cs	11	Active
CS1519	Invalid token ';' in class, record, struct, or interface member declaration	basic C-sharp difference	Number.cs	11	Active
CS1520	Method must have a return type	basic C-sharp difference	Number.cs	21	Active
CS0103	The name 'num' does not exist in the current context	basic C-sharp difference	Number.cs	14	Active
CS0103	The name '_number' does not exist in the current context	basic C-sharp difference	Number.cs	14	Active
CS0103	The name '_number' does not exist in the current context	basic C-sharp difference	Number.cs	19	Active
CS0103	The name '_number' does not exist in the current context	basic C-sharp difference	Number.cs	19	Active
CS0103	The name '_number' does not exist in the current context	basic C-sharp difference	Number.cs	23	Active
IDE1007	The name '_number' does not exist in the current context.	basic C-sharp difference	Number.cs	11	Active
CS0246	The type or namespace name 'num' could not be found (are you missing a using directive or an assembly reference?)	basic C-sharp difference	Number.cs	12	Active

PYTHON:

But python is a dynamically typed language. This means we don't have to declare a variable's type before using it. Instead we can directly use the variables for our own purposes!

Furthermore, even if there is an error in type, it will still compile the code with the error (as it relies on inbuilt interpreter) and will only detect it during run time!



```
class Number:
    # instance attribute
    def __init__(self, _num):
        self.number = _num

    # instance method
    def Addfive(self):
        self.number = self.number + 5

    # instance method
    def Print(self):
        return self.number


# instantiate the class
num = Number(45)
# call our instance methods
num.Addfive()

print(num.Print())
```

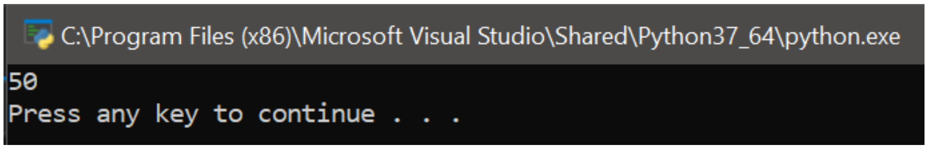
100 % No issues found

Output

Show output from: Debug

```
The thread 'MainThread' (0x1) has exited with code 0 (0x0).
The program 'python.exe' has exited with code -1 (0xffffffff).
```

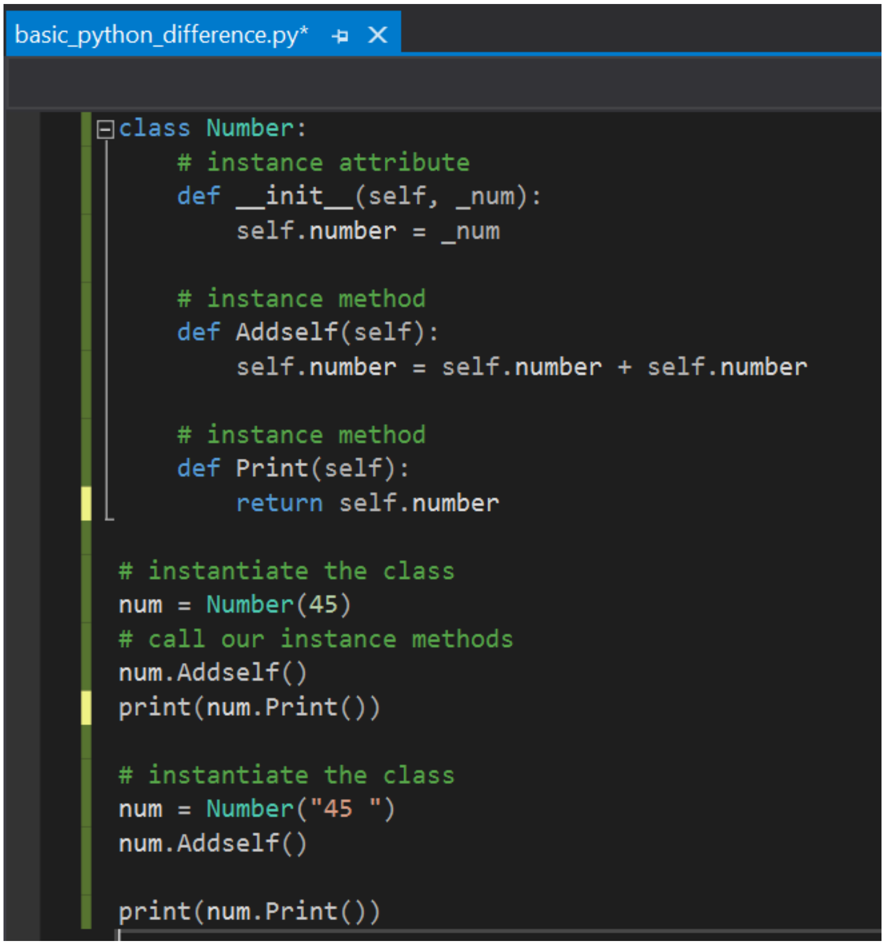
(Although we didn't mention the datatype anywhere, the code still ran as intended and 45 has been assigned to the number, 5 can be added to it and result can be printed out)



C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\python.exe

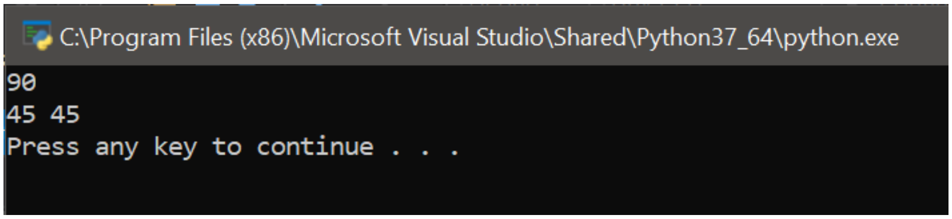
```
50
Press any key to continue . . .
```

Although this can lead to logical errors in code (as the same variable can be used to store different datatypes), it can also be exploited to create a dynamic class that does different things, depending on the type of data it receives, providing more flexibility to the code!



```
class Number:
    # instance attribute
    def __init__(self, _num):
        self.number = _num

    # instance method
    def Addself(self):
        self.number = self.number + self.number

    # instance method
    def Print(self):
        return self.number

# instantiate the class
num = Number(45)
# call our instance methods
num.Addself()
print(num.Print())

# instantiate the class
num = Number("45 ")
num.Addself()

print(num.Print())
```

(here I have altered the code to make it add itself instead of adding 5.
Now in 1st case, it is taking in a number and adding itself making 45 + 45 = 90.
In the 2nd case, it is taking in a string and concatenating itself to it, forming "45 " + "45 " which gives "45 45 " as output in print!)


C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\python.exe

```
90
45 45
Press any key to continue . . .
```

Conclusion:

Although there are several other differences between them, this point is the main one. That is because no matter how big or small, complex or simple the code is; it will have to have at least one variable to store data. Thus, knowing whether or not type needs to be declared for a variable, before assigning data to it, is absolutely crucial!

So, I end the report with a slight modification on Shakespeare's famous quote:

To declare or not to declare, that is the question

Reference:

https://hackr.io/blog/c-sharp-vs-python

https://stackoverflow.com/questions/41259203/does-python-garbage-collection-automatic-in-python

https://www.poetryfoundation.org/poems/56965/speech-to-be-or-not-to-be-that-is-the-question

https://www.netguru.com/blog/python-vs-c-sharp