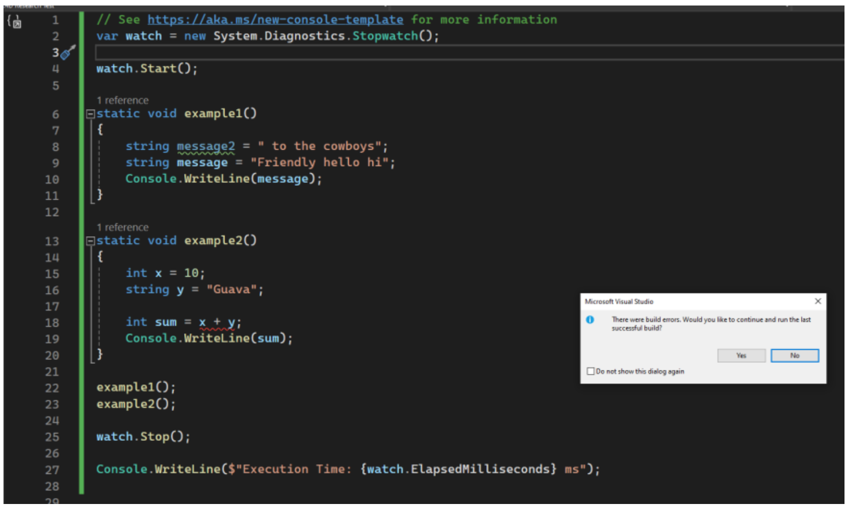
Research findings for topic: Comparing dynamic and static type systems.

Results

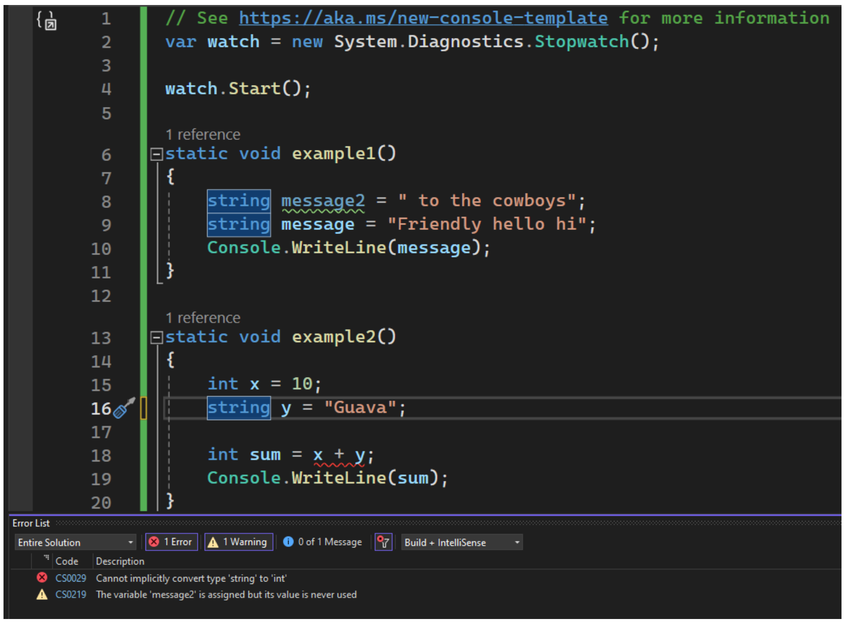
C# (static typed system)



```
// See https://aka.ms/new-console-template for more information
var watch = new System.Diagnostics.Stopwatch();

watch.Start();

1 reference
static void example1()
{
    string message2 = " to the cowboys";
    string message = "Friendly hello hi";
    Console.WriteLine(message);
}

1 reference
static void example2()
{
    int x = 10;
    string y = "Guava";

    int sum = x + y;
    Console.WriteLine(sum);
}

example1();
example2();

watch.Stop();

Console.WriteLine($"Execution Time: {watch.ElapsedMilliseconds} ms");
```

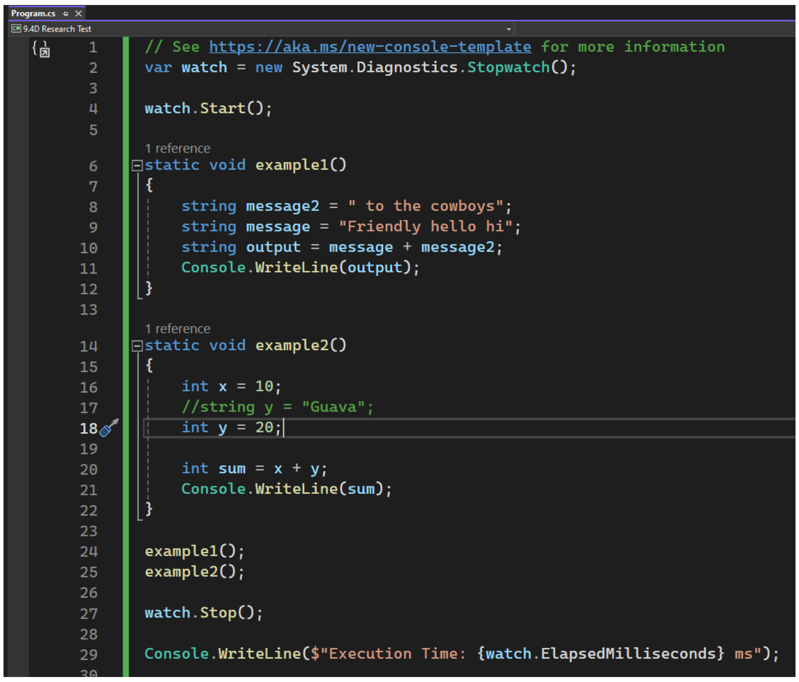
Microsoft Visual Studio

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

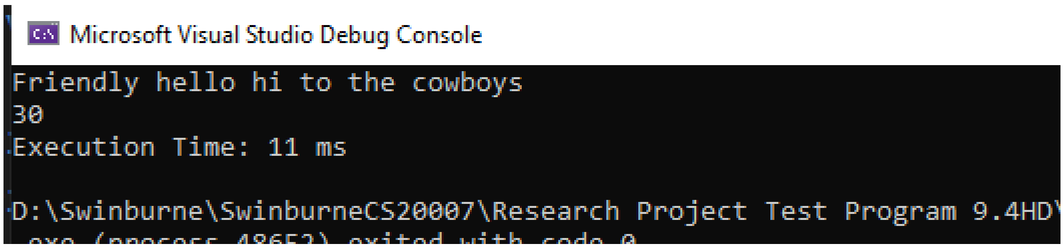


```
// See https://aka.ms/new-console-template for more information
var watch = new System.Diagnostics.Stopwatch();

watch.Start();

1 reference
static void example1()
{
    string message2 = " to the cowboys";
    string message = "Friendly hello hi";
    Console.WriteLine(message);
}

1 reference
static void example2()
{
    int x = 10;
    string y = "Guava";

    int sum = x + y;
    Console.WriteLine(sum);
}
```

```
Code    Description
CS0029  Cannot implicitly convert type 'string' to 'int'
CS0219  The variable 'message2' is assigned but its value is never used
```

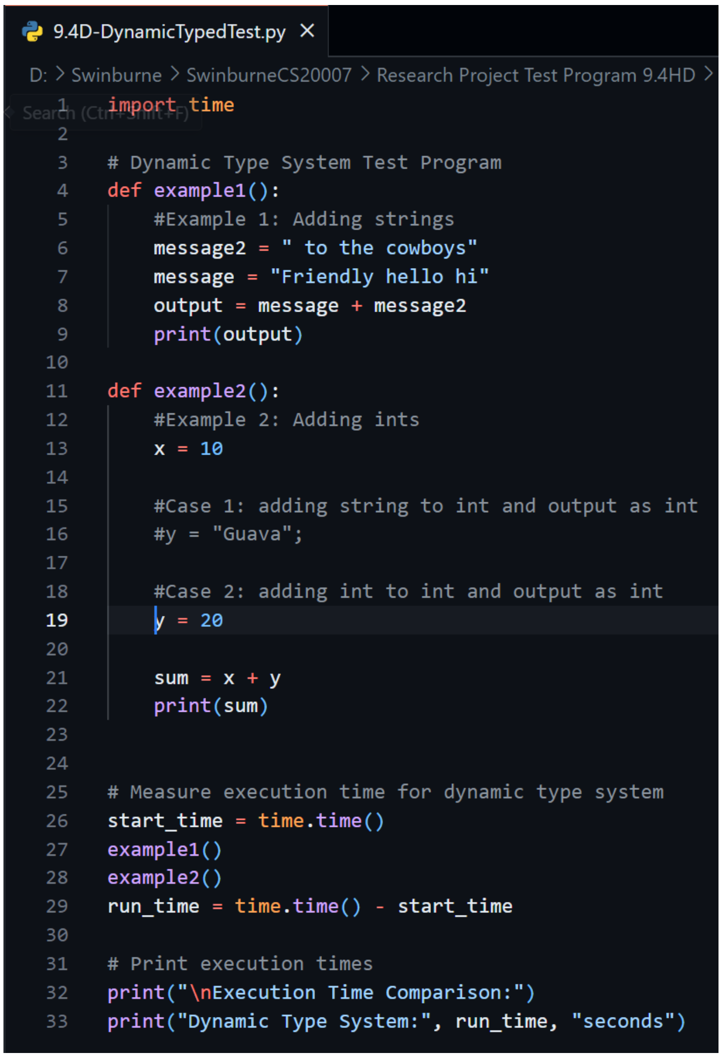


```
// See https://aka.ms/new-console-template for more information
var watch = new System.Diagnostics.Stopwatch();

watch.Start();

1 reference
static void example1()
{
    string message2 = " to the cowboys";
    string message = "Friendly hello hi";
    string output = message + message2;
    Console.WriteLine(output);
}

1 reference
static void example2()
{
    int x = 10;
    //string y = "Guava";
    int y = 20;

    int sum = x + y;
    Console.WriteLine(sum);
}

example1();
example2();

watch.Stop();

Console.WriteLine($"Execution Time: {watch.ElapsedMilliseconds} ms");
```

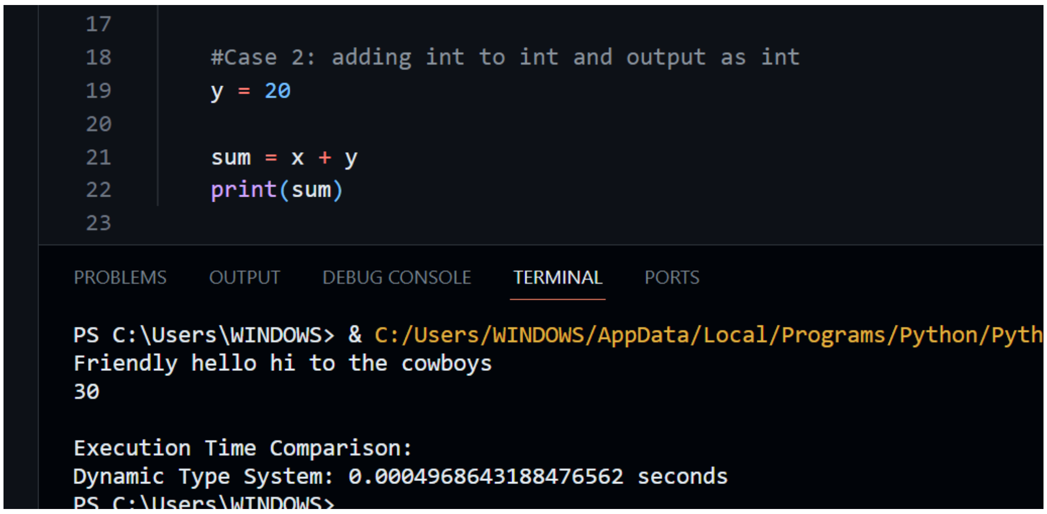


```
Friendly hello hi to the cowboys
30
Execution Time: 11 ms

D:\Swinburne\SwinburneCS20007\Research Project Test Program 9.4HD\
.exe (process 48652) exited with code 0
```

Python (dynamic type systems)

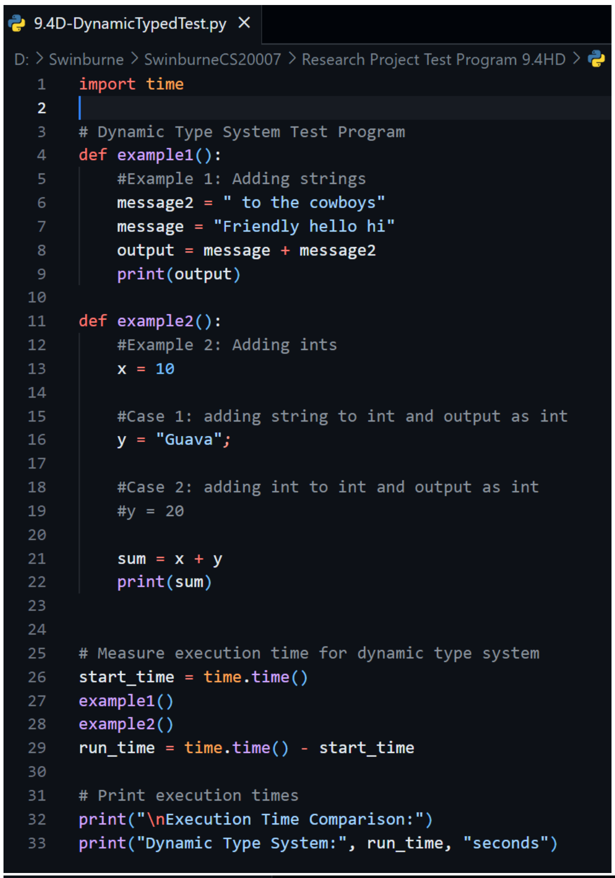

9.4D-DynamicTypedTest.py
D: > Swinburne > SwinburneCS20007 > Research Project Test Program 9.4HD >

```python
import time

# Dynamic Type System Test Program
def example1():
    #Example 1: Adding strings
    message2 = " to the cowboys"
    message = "Friendly hello hi"
    output = message + message2
    print(output)

def example2():
    #Example 2: Adding ints
    x = 10

    #Case 1: adding string to int and output as int
    #y = "Guava";

    #Case 2: adding int to int and output as int
    y = 20

    sum = x + y
    print(sum)


# Measure execution time for dynamic type system
start_time = time.time()
example1()
example2()
run_time = time.time() - start_time

# Print execution times
print("\nExecution Time Comparison:")
print("Dynamic Type System:", run_time, "seconds")
```



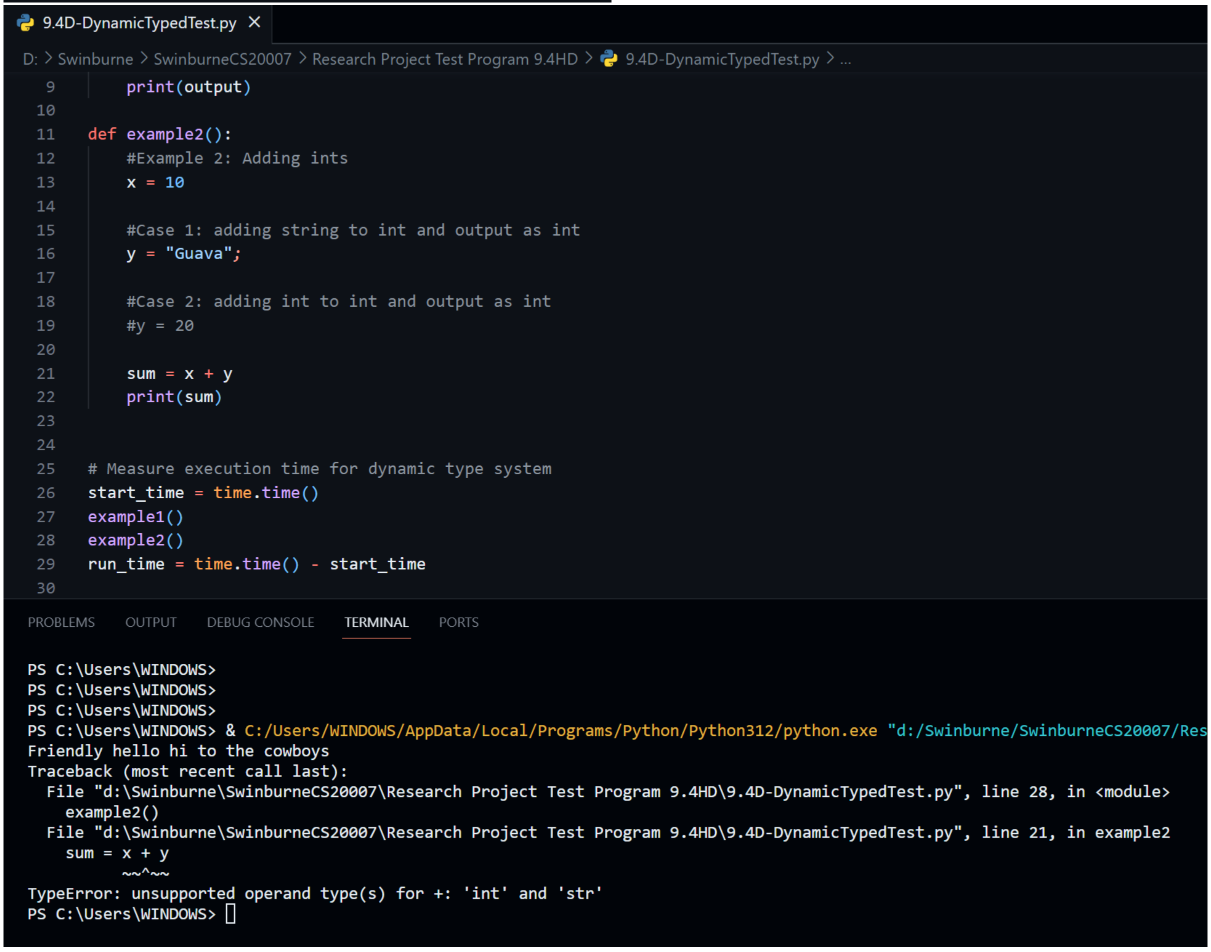
```python

    #Case 2: adding int to int and output as int
    y = 20

    sum = x + y
    print(sum)

```

```
PS C:\Users\WINDOWS> & C:/Users/WINDOWS/AppData/Local/Programs/Python/Pyth
Friendly hello hi to the cowboys
30

Execution Time Comparison:
Dynamic Type System: 0.0004968643188476562 seconds
PS C:\Users\WINDOWS>
```


9.4D-DynamicTypedTest.py

D: > Swinburne > SwinburneCS20007 > Research Project Test Program 9.4HD >

```python
import time

# Dynamic Type System Test Program
def example1():
    #Example 1: Adding strings
    message2 = " to the cowboys"
    message = "Friendly hello hi"
    output = message + message2
    print(output)

def example2():
    #Example 2: Adding ints
    x = 10

    #Case 1: adding string to int and output as int
    y = "Guava";

    #Case 2: adding int to int and output as int
    #y = 20

    sum = x + y
    print(sum)


# Measure execution time for dynamic type system
start_time = time.time()
example1()
example2()
run_time = time.time() - start_time

# Print execution times
print("\nExecution Time Comparison:")
print("Dynamic Type System:", run_time, "seconds")
```


9.4D-DynamicTypedTest.py

D: > Swinburne > SwinburneCS20007 > Research Project Test Program 9.4HD > 9.4D-DynamicTypedTest.py > ...

```python
        print(output)

def example2():
    #Example 2: Adding ints
    x = 10

    #Case 1: adding string to int and output as int
    y = "Guava";

    #Case 2: adding int to int and output as int
    #y = 20

    sum = x + y
    print(sum)


# Measure execution time for dynamic type system
start_time = time.time()
example1()
example2()
run_time = time.time() - start_time

```

```
PS C:\Users\WINDOWS>
PS C:\Users\WINDOWS>
PS C:\Users\WINDOWS>
PS C:\Users\WINDOWS> & C:/Users/WINDOWS/AppData/Local/Programs/Python/Python312/python.exe "d:/Swinburne/SwinburneCS20007/Res
Friendly hello hi to the cowboys
Traceback (most recent call last):
  File "d:\Swinburne\SwinburneCS20007\Research Project Test Program 9.4HD\9.4D-DynamicTypedTest.py", line 28, in <module>
    example2()
  File "d:\Swinburne\SwinburneCS20007\Research Project Test Program 9.4HD\9.4D-DynamicTypedTest.py", line 21, in example2
    sum = x + y
          ~~^~~
TypeError: unsupported operand type(s) for +: 'int' and 'str'
PS C:\Users\WINDOWS> 
```

Discussion

It is evident that the dynamic type systems have an incredibly fast running speed at 0.5 milliseconds, comparing to the static type systems' running speed at 11 milliseconds. The C# program that is used to test static type system has prevented the program from being executed, as it detects incompatible data types during the compile time. On contrast, the Python program that is used to test dynamic type system can assign and concatenate variables without any problem, until the error of incompatible data type was detected during the run time. Dynamic type systems do not enforce type annotations and semi colons in the syntax. This can make the Python program harder to read and debug as the errors are not detected until execution time which can be frustrating for programmers when they are working on projects with high complexity. However, the C# program code enforces type annotations and semi colons in its syntax. The quicker and simpler debugging can compensate for the program's slower running time.

Conclusion

To summarise, the dynamic type systems may offer a lot simpler syntax, flexibility, and much lesser regulation for writing code, but it can cause potential problems that will require an extensive amount of time and effort to debug in a more complicated program. The static type of system will require more work in writing code, but a program/software that are less complex, easier to read, and quicker to debug, will most definitely make up for effort that was put into adhere static type systems' coding regulations and logic structure. In my opinion, both types of type systems have their own benefits and can be used in many projects. What matters is that the programmer must first understand the project requirements before deciding which type of type systems will be implemented for the said project.