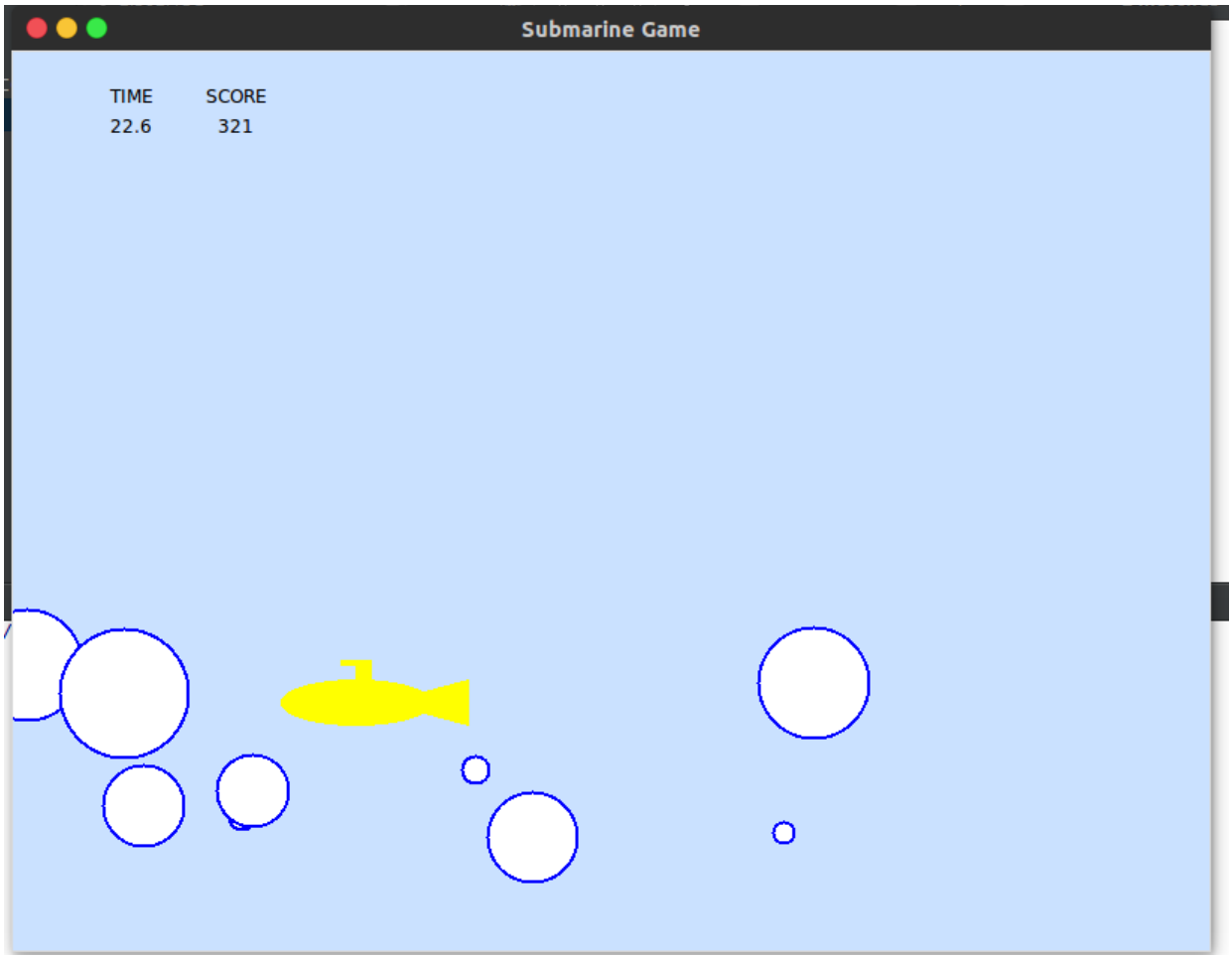
Matthew Koken
COEN 171
Assignment 2

1. Is Python an interpreted language? What does this mean?
   For the most part, Python is an Interpreted language. This means it does not strictly go through a compilation process in order to be run. Instead, it is interpreted by an interpreter so that it can be run. This can sort of be seen through IDLE – this allows scripts, or single lines of code to be interpreted and run on demand, like a command line interface. Here, the instructions are executed directly by the interpreter and are not compiled compiled beforehand. It is important to note, however, that some implementations of Python differ. Some may see a *.pyc file, which is byte code file for the Python virtual machine. Others may use a form of just-in-time compilation that compiles this byte code (created by the interpreter) into machine code.
2. Will you consider Python an imperative Programming Language? Why?
   I consider Python to be an Imperative Programming Language. Python follows the basic structure of an Imperative PL – expressions in Python consist of variables, assignments, condition statements, etc. Like a recipe, you read and follow the instructions. The statements change the state of the program, and describe how the program operates. The primary control flow consists of loops, conditionals, and function/method calls. Again, all of which fit the imperative PL paradigm.
3. Can you declare global variables?
   Yes, you can declare global variables in Python. They must be declared before any instance attempts to use them. To do so, you would use the keyword “global.” For example, you would declare “global count” towards the top of the file so that it can be used globally. Once it has been declared, you can perform operations and assignment on it. However, if it is to be used in a function, you must declare that the global variable is used, and not a local. Again, before the variable is used, you would declare “global count” within the function.
4. How are parameters passed to functions?
   In Python, parameters are passed by reference to a function. Given a function definition like “def func(parameter),” you would pass a reference to the function when you call it such as in “func(arg1)”
5. How are for loops created?
   In Python, anything with an iterable method can be used to implement a for loop – essentially the loops run using lists. So given “for x in range(0, 10):” the loop will iterate through the range of 0 to 10, covering x=0,1,2,3,4,5,6,7,8,9. Anything that can give a list or be iterated through can be used, so from my code the example “for mine in mine_list:” would iterate through all of the mine arrays within the mine_list array.
6. Flexibility (of for loops)
   Again, Python for loops are quite flexible. They can utilize any variable that is iterable or a list. So anything that can provide a sequence can be used. However, traditional C-style for loops are not valid. In that case, your only option would be to use a while loop.
7. Execution Speed
   Because Python is interpreted, it will execute slower than a compiled language such as C++. Every variable in Python is an object, so it cannot find performance improvements in using primitive types that may be hardware supported. Because of this, and requiring a Python virtual machine to run, Python uses much more memory. Altogether, this drastically reduces the execution speed of Python statements.


Submarine Game
TIME
22.6
SCORE
321