Computer Science 1	Exercises 09.07-11	Date:
Name:		Period:

1. Are arguments used by all subroutines, or just the majority of them?

2. All subroutine declarations have an identifier followed by _____.

3. In terms of Python programming, what are *arguments*?

4. In terms of Python programming, what are *parameters*?

5. A copy of the information is passed from the _____ to the _____.

6. An argument can be several different things. List 4 examples.

7. Does argument sequence matter?

8. Compare programs **ParameterProcedures03.py** and **ParameterProcedures04.py**.
Both programs call a procedure twice, the first time with the arguments in the correct sequence and the second time where the arguments are reversed. As expected, this causes 2 different outputs in **ParameterProcedures04.py**; however, even with the arguments in the wrong order **ParameterProcedures03.py** manages to display the same output twice. How is this?

9. Look at program **ParameterProcedures05.py**. Why does this program not execute?

10. When calling a subroutine, do all of the arguments all need to be of the same type?

11. Can the variable names of the arguments be the same as the variable names of the parameters?

12. Can the variable names of the arguments be different from the variable names of the parameters?

13-16. List 4 important rules for using procedures with parameters. (This counts as 4 questions.)

17. What is the difference between a *function* and a *procedure*?

18. All functions require a _____ statement, and it will usually be the _____ statement in the function.

19. The real benefit of Boolean variables is they help to make your programs more _____.

20. Look at program **Functions01.py**. What is returned by the **getNextNumber** function?

21. Look at program **Functions02.py**. What is accomplished by the **checkPIN** function?

22. Can functions have multiple arguments and parameters?

23. Look at the **add1** procedure and **add2** function in program **Functions03.py**. Both of these receive 2 parameters and compute their sum. List 2 differences between these 2 subroutines.

24. Explain how calling a *function* is different from calling a *procedure*.

25. When a value is returned from a function, you should do something with it.
Give 3 examples of something you can do with the returned value.

26. Look at program **Functions05.py**. What is wrong with the **add** function call on line 28?

27. Can one procedure be used to create another procedure?

28. Look at program **SubFromSub03.py**.
Give 2 examples of how this program confirms your answer to the previous question.

29. Can you still use a subroutine even if you do not know how it works?

30. What do you need to know in order to use a subroutine?

31. Can one function be used to create another function?

32. Look at program **SubFromSub05.py**.
Explain how this program confirms your answer to the previous question.