Review for Test #1

Please look over all class notes, handouts and lab questions to help you prepare for the test. Here are some sample questions for review. These questions show you the types of questions you may see on a test. However, they do not necessarily cover all topics that we have seen.

1. For each statement, determine if it is true or false.
	1. In Python, to obtain numerical input from the user, you should use the raw\_input( ) function instead of the input( ) function.
	2. In Python, forgetting to put a colon at the end of a line that introduces a loop is an example of a syntax error.
	3. The ENIAC was one of the world’s first commercially successful electronic computers.
	4. Code that is written in assembly language can be executed on any computer.
	5. The purpose of the CPU is to perform operations contained in computer instructions.
	6. Hollerith’s tabulator is an example of an analog machine.
2. Define the following terms:
	1. Iteration
	2. Algorithm
	3. Logical error
	4. Machine language
	5. Software
	6. Variable
3. According to a rule of thumb called \_\_\_\_\_\_\_\_\_\_ Law, the speed of the latest computer technology doubles every 18 months.
4. A simple program usually consists of three phases: \_\_\_\_\_\_\_\_\_, followed by \_\_\_\_\_\_\_\_\_\_, followed by \_\_\_\_\_\_\_\_\_\_\_.
5. Describe the steps necessary in the computer problem-solving procedure. For each step you specify, explain how we might fail to correctly perform that step.
6. Why is it preferable to write a computer program in a machine-independent language?
7. Write a Python statement that will subtract 5 from the variable count and put the result in a new variable called adjusted\_count.
8. Suppose w is a string variable containing a word.
	1. Show in pseudocode how you would determine whether the letter ‘z’ appears in w.
	2. How would your answer to the previous question change if we wanted to know the number of times the letter ‘z’ appears?
	3. Assuming that the word w contains at least one letter ‘z’, show in pseudocode how you would determine the location of the first ‘z’ in w.
9. Show in pseudocode how you would get the computer to print the numbers from 1 to 26, using a loop. How would you change your answer if we wanted to print the numbers in reverse order?
10. Consider the following Python code:

number = input(“Please enter a number”)

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

 valid = True

else:

 valid = False

Notice that the above code will set the variable valid to True if some condition (as yet unspecified) satisfies a certain condition. For each of the following cases, indicate what condition we should enter in the blank in order for the variable valid to be set to True. In all cases, assume that the number entered is an integer.

* 1. The number is a positive 3-digit number.
	2. The number is between 18 and 65 inclusive.
	3. The number is positive.
	4. The number is a multiple of 5, but not a multiple of 4.
	5. The number is an odd number.
	6. The number is 7, 11 or 13.
1. Show the steps necessary to sort [ 5, 4, 7, 3, 2, 6, 1 ] in ascending order using insertion sort. Include enough detail so that it is clear you understand how this sorting method works.
2. Given a number of cents (anywhere from 1 to 99), explain how we would determine the appropriate change, i.e. the number of quarters, dimes, nickels, and pennies.
3. In Python, what does it mean when a line begins with a #? What is the purpose?
4. What is the output of this Python code?

for i in range (2, 8):

 print i

1. The following Python code attempts to find the largest and smallest numbers in a list L. However, I forgot to indent some lines. Show how we should indent the statements.

max = L[0]

min = L[0]

for num in L:

if num > max:

max = num

if num < min:

min = num

print “The max is “ + str(max) + “ and the min is “ + str(min) + “.”

Answers:

1. False, true, false, false, true, true
2. Terminology:
	1. An iteration is one of the repetitions of the code inside a loop.
	2. An algorithm is a clearly spelled out, step-by-step, explanation in English that tells us how to solve a problem.
	3. A logical error is a mistake in a computer program that, although the program appears to run normally, causes the output to be incorrect.
	4. Machine language is the native language of a particular computer, unique to each manufacturer. In it, the instructions are coded using binary numbers.
	5. Software consists of the computer programs in your computer. Its purpose is to utilize the hardware and make the computer do useful work.
	6. A variable is a memory location used to store value in a computer program, whose value may change during the program execution.
3. Moore’s
4. Input, calculations, output
5. We can break down the problem-solving procedure into five steps.
	1. Read and understand the problem. One obstacle here is that we might not understand the problem because its specification is ambiguous.
	2. Explain in English how to solve a problem. The obstacle here is that we might not know how to solve the problem…
	3. Convert your English solution into code written in a programming language. An obstacle here is that you might not know enough about the programming language to express what you want in English.
	4. Compile the program, meaning we ask the computer to convert our code into machine language. The obstacle here is that we may have a syntax error and need to re-edit the program.
	5. Run and test the program. The obstacle here is that we may discover that the output is incorrect, and we have to go back to the way we designed the solution or how it was typed in.
6. With a machine-independent language, it doesn’t matter what kind of computer you use. Your program will work on many platforms, not just one. If you only had machine-dependent languages, you would have to retype your program on each type of computer you wish to support.
7. adjusted\_count = count – 5
8. Let’s assume that w is a word. The following solutions are English pseudocode.
	1. Create a variable called found and set it initially to False.

For each letter in the word w:

If this letter is ‘z’, then:

 Set the variable found to True

 After the loop, check to see whether found is True or False.

* 1. Instead of a found variable that is simply True or False, we replace it with a variable called count that will represent the number of times ‘z’ appears in the word. The solution becomes:

Create a variable called count and set it initially to zero.
For each letter in the word w:
 if this letter is ‘z’, then:
 add 1 to count

After the loop, the value of count is how many z’s were in the word.

* 1. To find the location of the first z, we need to keep track of locations (apartment numbers) in our word. And once we find the z, we need to immediately exit the loop.

Create a variable called location and set it initially to -1.
For i = 0 to the length of w minus 1, inclusive:
 if this letter is ‘z’, then:
 set location equal to i
 break from the loop

After the loop, the value of location is the index of the first z in w. If location is still -1, then we can conclude that there was no z in w. Note that the first step was optional since we were given the fact that the word did contain a z.

1. For each number from 1 to 26 inclusive:

Print the number

 To count backwards:

 For each number from 1 to 26 inclusive:
 Print 27 minus the number.

1. We need to specify a condition for a variety of scenarios:
	1. The number is a positive 3-digit number.

if number >= 100 and number < 1000:

* 1. The number is between 18 and 65 inclusive.

if number >= 18 and number <= 65:

* 1. The number is positive.

if number > 0:

* 1. The number is a multiple of 5, but not a multiple of 4.

if number % 5 == 0 and number % 4 != 0:

* 1. The number is an odd number.

if number % 2 == 1:

* 1. The number is 7, 11 or 13.

if number == 7 or number == 11 or number == 13:

1. Let’s do insertion sort on [ 5, 4, 7, 3, 2, 6, 1 ] to make this list ascending.

First, we let 5 sit at the beginning of the list.

Next, we consider the second number, 4. We observe that 4 < 5, so need to shift 5 to the right to make room for the 4. Now, we have [ 4, 5, 7, 3, 2, 6, 1 ].

Next, we consider the third number, 7. We observe that 7 > 5, so 7 is exactly where it needs to be. No change to the list.

Next, we consider the fourth number, 3. We observe that 3 < 7, 3 < 5, and 3 < 4. So, 3 must go to the front of the list. The numbers 4, 5, and 7 must shift to the right to make room. The list becomes [ 3, 4, 5, 7, 2, 6, 1 ].

Next, we consider the fifth number, 2. We observe that 2 is less than all the numbers before it in the list. So, they must shift to the right to make room for the 2. The list becomes

[ 2, 3, 4, 5, 7, 6, 1 ].

Next, we consider the sixth number, 6. We observe that 6 < 7 but 6 > 5. So 6 must be inserted between the 5 and 7. To do so, we must shift the 7 to the right to make room. The list beomes [2, 3, 4, 5, 6, 7, 1 ].

Finally, we consider the last number, 1. We observe that 1 is less than all the numbers before it. We must shift these numbers to the right to make room. Then our list is [ 1, 2, 3, 4, 5, 6, 7 ].

1. We will make use of the following variables:

cents – the input value that is somewhere between 1 and 99, inclusive.

quarters, dimes, nickels, pennies – the values of each type of coin

quarters = cents / 25

cents = cents % 25

dimes = cents / 10

cents = cents % 10

nickels = cents / 5

pennies = cents % 5

1. Lines that begin with a # are called comments. Comments are ignored by the computer, and they are for the human reader. We often write our English explanations in pseudocode, and these can go into a comment. For example, if a computer program contains a formula that is not obvious, or that took a long time to figure out, then it is a good idea to write a comment before this formula to remind ourselves how we derived the formula. That way, later we may forget the formula, and can read the comment to understand what our thought process was.
2. 2
3
4
5
6
7
3. It should be indented like this:

max = L[0]

min = L[0]

for num in L:

 if num > max:

 max = num

 if num < min:

 min = num

print “The max is “ + str(max) + “ and the min is “ + str(min) + “.”