Name: _____ **Section:** **1** **2** **3** **4**

1 = Mutchler, 1st-2nd periods. **2 =** Mutchler, 3rd-4th periods. **3 =** Anderson, 7th-8th periods. **4 =** Anderson, 9th-10th periods.

Use this quiz to help make sure you understand the videos/reading. ***Answer all questions.*** Make additional notes as desired. ***Not sure of an answer?*** Ask your instructor to explain in class and revise as needed then. **Please print two-sided.**

Throughout, where you are asked to "circle your choice", you can underline or circle it (whichever you prefer).

Video: ***The Wait-Until-Event Pattern*** [7:34 minutes]

1. Write (to the right) a ***definite loop (using a `for` statement)*** that prints the numbers 1 through 1000, inclusive.

2. One of the following is the ***Definite Loop*** pattern and one is the ***Wait-Until-Event*** pattern. Draw arrows from the phrases to the patterns to indicate which is which.

***Wait-Until-Event* pattern** ***Definite Loop* pattern**

```
Run n times:
    ...
    ...
```

```
Repeatedly:
    ...
    Has the event occurred?
    If so, break out of the loop.
    ...
```

3. Explain the role of the "sentinel" value in the *wait-until-event* pattern, when getting input from a user.

4. Write (to the right) an ***indefinite loop (using a `while` statement)*** that prints the integers 1 through 1000, inclusive. (This problem would be better solved with a ***`for`*** statement, but I am asking you to use a ***`while`*** statement here so that you can practice ***`while`*** statements.)

5. Write (to the right) an ***indefinite loop (using a `while` statement)*** that prints integers starting at 100,000 and stopping when it encounters an integer whose cosine is less than **`-0.999`**. Do NOT print the integer whose cosine is less than **`-0.999`**.

6. How would you need to modify the previous problem if you were supposed to print the integer whose cosine is less than -0.999 (but still stop the loop after doing so)?

Textbook Reading: ***Section 4.1 – The WHILE Loop*** (pages 156 – 159) and ***Section 4.2 – Problem Solving: Hand-Tracing*** (pages 163 – 165) and ***Section 4.3 Application – Processing Sentinel Values*** (pages 166 – 169)

7. First, convince yourself that the two code snippets to the right are equivalent, that is, they print the same thing, doing essentially the same computation. Then, show what they print. (To save space, write your answer in a line instead of a column.)

   Hint: they each print fewer than 10 numbers!

```
n = 1
while n < 100:
    n = 2 * n
    print(n)
```

```
n = 1
while True:
    if n >= 100:
        break
    n = 2 * n
    print(n)
```

Question to think about: Which of the above two "feels" better to you, for this problem?

8. What is the ***sentinel*** value in the code snippet shown to the right?

```
# The following prints the sum
# of the numbers that are input.

total = 0
while True:
    x = input('Enter a number, or Q to quit.')
    if x == 'Q':
        break
    number = int(x)
    total = total + number
print(total)
```

9. In the code snippet to the right, suppose we tried to change the ***input*** line to:

```
x = int(input('Enter
a number, or Q to quit.'))
```

   and then deleted the subsequent call to ***int***. We have done similar things in previous sessions. But it won't work here – what goes wrong?

10. Again continuing to refer to the code snippet above, suppose that the human user enters a lower-case `'q'` in an attempt to stop the loop. What will happen?

Video: ***The Create Robot Library*** [9:10 minutes]

11. Write a complete ***`main`*** function that does the following (in the order listed):
    - Constructs a ***`Create`*** object (assume that the COM port number is 4) and puts the Create in full mode.
    - Prints the distance the robot has traveled (which should be zero at this point).
    - Makes that robot go backward at 30 cm/second for 2.5 seconds, then stop.
    - Prints the distance the robot traveled.
    - Shuts down the robot.

12. Write a function called ***`go_distance`*** that takes as a parameter a ***`Create`*** object and makes that robot go forward at 20 cm/second until the robot has gone a distance of 50 cm, ***using the following algorithm***:

    Start the robot moving at 20 cm/second.

    Repeatedly:

        Using the distance sensor, determine whether the robot has gone 50 cm or more yet. If so, break out of the loop.

    Stop the robot.