CS 301 - Spring 2019
Instructors: Tyler Caraza-Harter and Caroline Hardin

Exam 1 — 15%

(Last) Surname: _____ (First) Given name: _____

NetID (email): _____ @wisc.edu

Fill in these fields (left to right) on the scantron form (use #2 pencil):
1. LAST NAME (surname) and FIRST NAME (given name), fill in bubbles
2. IDENTIFICATION NUMBER is your Campus ID number, fill in bubbles
3. Under *ABC* of SPECIAL CODES, write your lecture number, fill in bubbles:
     001 - MWF 9:55pm (Caroline)
     002 - MWF 1:20pm (Tyler afternoon)
     003 - MWF 8:50am (Tyler morning)
4. Under *F* of SPECIAL CODES, write **A** and fill in bubble **6**

If you miss step 4 above (or do it wrong), the system may not grade you against the correct answer key, and your grade will be no better than if you were to randomly guess on each question. So don't forget!

Many of the problems in this exam are related to the course projects, but some questions assume the availability of slightly different functions (e.g., for accessing the data). We won't have any trick questions where we call a function that doesn't exist and you need to notice. Thus, if you see a call to a function we haven't explicitly defined in the problem, assume the function was properly implemented (perhaps immediately before the code snippet we DO show) and is available to you.

You may only reference your notesheet. You may not use books, your neighbors, calculators, or other electronic devices on this exam. Please place your student ID face up on your desk. Turn off and put away portable electronics now.

Use a #2 pencil to mark all answers. When you're done, please hand in these sheets in addition to your filled-in scantron.

(Blank Page)

General Concepts

1. True or False: if both keyword and default arguments are available to initialize a parameter, Python will use the default argument.

   A. True   B. False

2. Which of the following IS NOT of type int?

   A. 7 // 4   B. 8 / 4   C. 9 % 4   D. 10 + 4   E. 11 * 4

3. In what order does Python apply operators?

   A. math before logical before comparison

   B. logical before comparison before math

   C. comparison before logical before math

   D. math before comparison before logical

   E. logical before comparison before math

4. Suppose Python executes some code before crashing and providing an error message indicating the problematic line. What kind of error was this?

   A. syntax   B. runtime   C. semantic   D. infinite loop   E. infinite conditional

5. You're writing a loop inside a function. Which statement(s) would cause the loop to stop the current iteration and perform no more iterations (regardless of the loop's condition)?

   A. return   B. break   C. continue   D. both A and B   E. both B and C

6. Which of the following kinds of statements may execute at most once in a function during a single invocation?

   A. print   B. return   C. break   D. continue   E. pass

7. What does the following evaluate to? (3 * 3 - 3 / 3) ** 2 + 1

   A. 0   B. 1   C. 5   D. 17   E. 65

8. What is printed?

```
w = "x"
y = "z"
print(w + "w" + y)
```

   A. wwy   B. wwz   C. xwy   D. xwz   E. www

9. What is the **value in x** after the following code executes?

```python
def foo():
    print(1)

    return 2
x = foo()
```

A. None   B. 1   C. 2   D. both B and C   E. the program crashes before x is set

10. Consider the following code.

```python
x = 1

def f():
    print(x)
    x = 2
f()
print(x) # what does this print?
```

A. 0   B. 1   C. 2   D. 3   E. nothing (it crashes before)

11. What is printed?

```python
def h(x=1, y=2):
    print(x, y)

def g(x, y):
    h(y)

def f(x, y):
    g(x=x, y=y+1)

x = 10
y = 20
f(y, x)
```

A. 1 2   B. 10 20   C. 10 21   D. 11 2   E. 20 11

City Spending Project

12. Which of the following is a valid variable name?

A. 2018_dollars   B. spend$18   C. __spend_18   D. if   E. "year18"

13. If we want to simplify the body of the `rising` function, what is the shortest alternative that produces the same results as the original code? Assume all parameters are initialized with integer values.

```
def rising(spend1, spend2, spend3, spend4):
    if spend1 < spend2:
        if spend2 < spend3:
            if spend3 < spend4:
                return True
    return False
```

    A. return spend4 > spend1

    B. spend1 <= spend2 <= spend3 <= spend4

    C. spend3 < spend4 and spend2 < spend3 and spend1 < spend2

    D. spend1 < spend2 or spend2 < spend3 or spend3 < spend4

    E. none of the above implements equivalent logic

14. What is the output?

```
def func(y1, y2):
    if y2 > y1:
        return y2 - y1
    return y1 - y2
print(func(2018, 2015))
```

A. -3   B. 3   C. both -3 and 3   D. 2015   E. 2018

15. Assume `get_budget(agency)` returns the current budget of the given agency. For which agency does the following code print the budget?

```
police = "library"
parks = get_budget(police)
police = "streets"
print(parks)
```

A. police   B. library   C. streets   D. parks   E. fire

16. What is the type of x?

```
streets_spending = "26.655"
x = len(streets_spending)
```

A. str   B. float   C. int   D. bool   E. NoneType

17. Assume `get_spending(2015)` returns 5 and `get_spending(2016)` returns 4. What is printed?

```python
def change(y1, y2):
    spend1 = get_spending(y1)
    spend2 = get_spending(y2)
    if y2 > y1:
        if spend2 > spend1:
            return "up-1"
        else:
            return "down-2"
    else:
        if spend2 > spend1:
            return "down-3"
        else:
            return "up-4"
print(change(2016, 2015))
```

A. up-1   B. down-2   C. down-3   D. up-4   E. None

18. What should replace `????` if only years 2000 through 2018 (inclusive) are allowed? Remember that `and` is higher precedence than `or`.

```python
def compute_increase(year1, year2):
    if ????:
        print("WARNING!  Bad year provided")
        return None
    # code that assumse year1 and year2 are in range...
```

  A. year1 >= 2000 and year1 <= 2018 and year2 >= 2000 and year2 <= 2018

  B. year1 >= 2000 or year1 <= 2018 or year2 >= 2000 or year2 <= 2018

  C. year1 < 2000 or year1 > 2018 or year2 < 2000 or year2 > 2018

  D. year1 < 2000 or year1 > 2018 and year2 < 2000 or year2 > 2018

  E. (year1 < 2000 or year1 > 2018) and (year2 < 2000 or year2 > 2018)

19. Which of the following is a possible value for `budget` after running the following? Although the code is strangly spaced, assume there are no bugs and y17 and y18 were defined earier in the code.

```
budget = 0
if y18 >= 1000:
    budget += 30

elif y18 >= 500:
    budget += 20
else:
    budget += 10
if y17 >= 1000:
    budget += 2

else:
    budget += 1
```

A. 10   B. 22   C. 33   D. 51   E. 63

Quiz Project

20. Your quiz program never crashes, but makes the quiz taker keep retrying the same question until all retries are exhausted, even after a correct guess! This is an example of a _____ error:

A. syntax   B. runtime   C. semantic   D. diabolical   E. all of the above

21. What is printed? Assume the users always answers 3, making the if's condition True.

```
score = 0

def quiz():
    score = 0
    for i in range(3):
        if input("what is 1+2? ") == "3": # assume always True
            score += 1
quiz()
quiz()
print(score)
```

A. 0   B. 3   C. 4   D. 6   E. 8

22. What is the **type** of `num`? Assume the user types the following (with no quotes): 3

```
num = input("What is 1+1?")
```

A. True   B. False   C. bool   D. str   E. int

23. How could we start a for loop to ask the same number of questions as asked by this while loop?

```
i = 0
while i <= 5:
    askRandQuestion()
    i += 1
```

    A. for 5:

    B. for i in range(5):

    C. for i in range(6):

    D. for i in 5:

    E. for i in range(len(6)):

24. Assume there is a function `is_correct(guess, actual)` that returns True or False, depending on whether or not `guess` is close enough to `actual`. How can the code below be shortened without changing what it does? Choose the answer that minimizes the keystrokes necessary to type the code but that doesn't affect what will be printed under any scenario.

```
guess = "A"
actual = input("your guess: ")
if is_correct(guess, actual) == True:      # line 1
    print("rock on!")                      # line 2
elif is_correct(guess, actual) == False:  # line 3
    print("better luck next time")         # line 4
```

    A. the code cannot be shortened further

    B. delete `== True` on line 1

    C. delete `== True` on line 1 and replace all of line 3 with `else:`

    D. delete `== True` on line 1 and delete line 3 completely

    E. delete `== True` on line 1, delete line 3 completely, and remove the indent on line 4

25. What value of raw would NOT be counted correct by this code? In the answers, assume the user typed the text inside the quotes (the quotes themselves were not typed by the user).

```
raw = input("guess: ")
cleaned = raw.lower().rstrip().replace(",", "")
print("correct?", cleaned=="1000")
```

A. "1000"   B. "1,000"   C. "1,0,0,0"   D. "   1000"   E. "1000   "

26. How many questions does the following code print?

```
x = 4
while x != 0:
    print("what is " + str(x) + " squared?")
    x = (x + 3) % 5
```

A. 1   B. 2   C. 3   D. 4   E. 5

27. What hint will be printed?

```
hint = "trust your instincts"

def show_hint(hint="drink some coffee"):
    print(hint) # what is printed

def ask(q, a, hint="eliminate the worst option"):
    show_hint(hint=hint)
    # more code here...

ask("where do penguins keep their money?", "in a snow bank!")
```

     A. trust your instincts

     B. drink some coffee

     C. eliminate the worst option

     D. where do penguins keep their money?

     E. in a snow bank!

Hurricane Project

28. The following function tries to compute the average damage per hurricane since the year 2000. Identify the bug in the code. This question is about the logical correctness of the average computation (assume the code runs without crashing).

```
def avg():
    total = 0
    for i in range(project.count()):
        if project.get_year(i) >= 2000:
            total += project.get_damage(i)
    return total / project.count() # numerator / denominator
```

    A. the numerator for the average is too large for some datasets

    B. the numerator for the average is too small for some datasets

    C. the denominator for the average is too large for some datasets

    D. the denominator for the average is too small for some datasets

29. Suppose we want to print 345000000. What should replace ???? with?

```
val = "345M"
print(????)
```

    A. `val.replace("M", "") * 1000000`

    B. `int(val[:-1]) * 1000000`

    C. `int(val[1:4]) * 1000000`

    D. `int(val - "M") * 1000000`

    E. `int(val - "M") + "000000"`

30. What should replace ???? so that the code prints "Flossy" (no spaces)? Hint: a call to `s.find(t)` returns the index of the string `t` within the string `s`.

```
name = "Hurricane Flossy"
print(????)
```

    A. `name[name.find(" ") :  ]`

    B. `name[name.find(" ")+1 :  ]`

    C. `name[ :  name.find("F")]`

    D. `name[name.find("Flossy")]`

    E. `name[1]`

31. What should replace `????` so that the code counts every "Floyd" in any case (upper, lower, mixed, etc.)?

```
count = 0
needle = "FLOYD"
for i in range(project.count()):
    name = project.get_name(i)
    if ????:
        count += 1
```

    A. `name == needle`

    B. `name == needle.lower()`

    C. `name.lower() == needle`

    D. `name == needle.upper()`

    E. `name.upper() == needle`

32. How many sub-blocks of code are there? Don't count the code that is not indented as a block for the purposes of this question.

```
flag = False
for i in range(project.count()):
    for j in range(project.count()):
        if i == j:
            continue
        if project.get_name(i).lower() == project.get_name(j).lower():
            flag = True
            break
    if flag:
        break
```

A. 3   B. 4   C. 5   D. 7   E. 9

33. What does `flag` represent in the previous code?

    A. Nothing, because it will always be False

    B. Nothing, because it will always be True when there is at least one Hurricane

    C. Whether all the hurricanes have the same name

    D. Whether all the hurricanes have lower-case names

    E. Whether there are at least two different hurricanes with the same name

34. What should replace ???? so that century is computed properly? For example, century should be a number like 1800, 1900, 2000, etc.

```
year = project.get_year(0) # will be a 4-digit int representing the year
century = ????
```

       A. `year / 100`

       B. `year // 100`

       C. `year - year % 100`

       D. `(year / 100) * 100`

       E. `int(year / 10) * 10`

35. What does the value in `total` represent at the end?

```
total = 0
for i in range(project.count()):
    mph = project.get_mph(i) # speed in miles-per-hour
    if mph > 100:
        continue
    deaths = project.get_deaths(i)
    total += deaths
```

       A. total deaths by all hurricanes

       B. deaths caused by hurricanes that are 100 MPH or slower

       C. deaths caused by hurricanes that are faster than 100 MPH

       D. deaths caused by hurricanes at the beginning of the dataset, up until the first one faster than 100 MPG

       E. average MPH of the hurricanes