CS 301 - Fall 2019

Instructor: Tyler Caraza-Harter

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 9:55am (Tyler morning)
   002 - MWF 4:35pm (Tyler afternoon)
4. Under *F* of SPECIAL CODES, write **C** and fill in bubble **8**

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)

City

1. If we want to simplify the body of the `decrease` function, which of the following has the same logic as the original code? Assume all parameters are initialized with integer values.

```
def decrease(yA, spendA, yB, spendB):
    if yA < yB:
        if spendA > spendB:
            return True
    if yA > yB:
        if spendA < spendB:
            return True
    return False
```

   A. return spendB > spendA

   B. return (yA < yB and spendA > spendB) and (yA > yB and spendA < spendB)

   C. return (yA < yB or spendA > spendB) and (yA > yB or spendA < spendB)

   D. return (yA < yB and spendA > spendB) or (yA > yB and spendA < spendB)

   E. return (yA < yB or spendA > spendB) or (yA > yB or spendA < spendB)

2. Which kind of import at `????` **would not** allow us to call the `log` function as we do on the next line?

```
????
x = log(10)
```

   A. `from math import *` B. `from math import log` **C.** `import math`

3. Assume `get_budget(agency)` returns the current budget of the given agency. If the following code prints the budget of the library agency, what must `????` be?

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

   A. `library` B. `streets` **C.** `parks` D. `fire`

4. What is printed?

```
def foo(y1, y2):
    n = 0
    if y2 > y1:
        n += 1
    elif y2 >= y1:
        n += 1
    else:
        n += 1
    if not(y2 < y1):
        n += 1
    if y2 != y1:
        n += 1
    return n
print(foo(2018, 2019))
```

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

5. Assume `get_spending(2018)` returns 5 and `get_spending(2019)` returns 4. What is printed?

```
def change(y1, y2):
    spend1 = get_spending(y1)
    spend2 = get_spending(y2)
    if y2 > y1:
        if spend2 > spend1:
            return "1"
        else:
            return "2"
    else:
        if spend2 > spend1:
            return "3"
        else:
            return "4"
    return "5"
print(change(2019, 2018))
```

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

6. Which function call **cannot** be used to answer the question "how much has spending increased per year (on average) for police from 2015 to 2018"?

```
def change_per_year(agency, start_year=2015, end_year=2018):
    agency_id = project.get_id(agency)
    y1 = project.get_spending(agency_id, start_year)
    y2 = project.get_spending(agency_id, end_year)
    return (y2-y1)/(end_year - start_year)
```

A. change_per_year()

B. change_per_year("police")

C. change_per_year("police", 2015)

D. change_per_year("police", 2015, 2018)

7. What should replace `????` to compute the average police spending over the period from 2015 to 2018? Note: there are two answers that produce the correct value; choose the better one.

```
total = 0
start = 2015
end = 2018
year = start
while year <= end:
    # assume returns spending by agency in given year
    total += get_budget("police", year)
    year += 1
avg = total / ????
```

A. 4

B. 2018

C. `end`

D. `(end - start)`

E. `(end - start + 1)`

8. What will be the type of `avg` in the previous question?

A. int **B. float** C. str D. bool E. sequence

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

A. `2019_year` **B.** `_year_2019_` C. `year.2019` D. `"year2019"` E. `2019`

General

10. If a train is heading towards a city at 50 mph, what does the following code print?

```
x = 5
y = 10

def z(x, y):
    x = y
    y = x
    y = y + 5
    x = x - y
    print(x, y)

z(x, y)
```

A. `0 10` B. `15 -5` C. `5 15` D. `-5 10` **E.** `-5 15`

11. Which of the following is of type int?

A. `1.0 + 3.0` B. `6 / 4` **C.** `6 // 4` D. `"35"` E. `5.0`

12. Which expression evaluates to 24?

A. (23) * 3** B. (3**2) * 2 C. 4 * (3 + 12) D. 4 + 4 * 2 + 8 E. 49 / 2

13. What type will be printed?

```
x = 10
print(type("w" + x))
```

A. int B. string C. float D. bool **E. none of the above due to an error**

14. A function without a return statement will automatically return the last variable used.

A. True **B. False**

15. What part causes a syntax error in the following code?

```
x = 0
while x < 10:
    x += 1
    if (x % 3) == 0:
        print(x)
    elif:
        print(10-x)

    else:
        continue
```

A. `x = 0`

B. `x += 1`

C. `elif:`

D. the extra blank line before the `else:`

E. `continue`

16. What is the correct order of output for the following:

```
def fe():
    fi(1)
    fo(2)
    fum(3)

def fi(x):
    print(x)

def fo(y):
    fi(y+2)
    print(y)

def fum(z):
    fo(z+2)

fe()
```

A. 1 2 4 5 7 **B. 1 4 2 7 5** C. 1 4 4 7 7 D. 4 2 1 7 5 E. 2 4 7 5 1

17. What is the output?

```
a = 1

def func(a):
    a = a + 1
    return a

b = func(a)
c = func(3)
print(a*100 + b*10 + c)
```

A. 124 B. 224 C. 324 D. 334 E. 444

18. Which operators have the **lowest** precedence?

A. logic B. math C. comparison

19. Which statement is **false**?

A. some functions take zero parameters

B. some functions take more than one parameter

C. multiple return statements may be written in a single function definition

D. multiple return statements may execute in a single function invocation

E. the same function may be invoked multiple times in a program

20. If you want to stop the current iteration of a loop and start a new iteration, you should use:

A. `break` **B.** `continue` C. `return` D. `pass`

21. What is printed?

```
total = 0
text = "12 34 5"
for c in text:
    if c.isdigit():
        total += int(c)
print(total)
```

A. `0` B. `5` **C.** `15` D. `12 34 5` E. `51`

Decisions

Consider these functions for the next four questions (assume any variables passed as arguments to these functions contain booleans):

```
def f(x, y):
    if x:
        if y:
            return True
        else:
            return False
    else:
        return False

def g(x, y):
    if x:
        return True
    else:
        return y
```

22. What does `f(True, False)` evaluate to?

    A. False B. True

23. A call to `f(b1, b2)` will return a result that is equal to which expression?

    A. b1 == b2 **B. b1 and b2** C. b1 or b2 D. b1 or not b2 E. not b1 or b2

24. A call to `g(b1, b2)` will return a result that is equal to which expression?

    A. b1 == b2 B. b1 and b2 **C. b1 or b2** D. b1 or not b2 E. not b1 or b2

25. What does the following evaluate to?

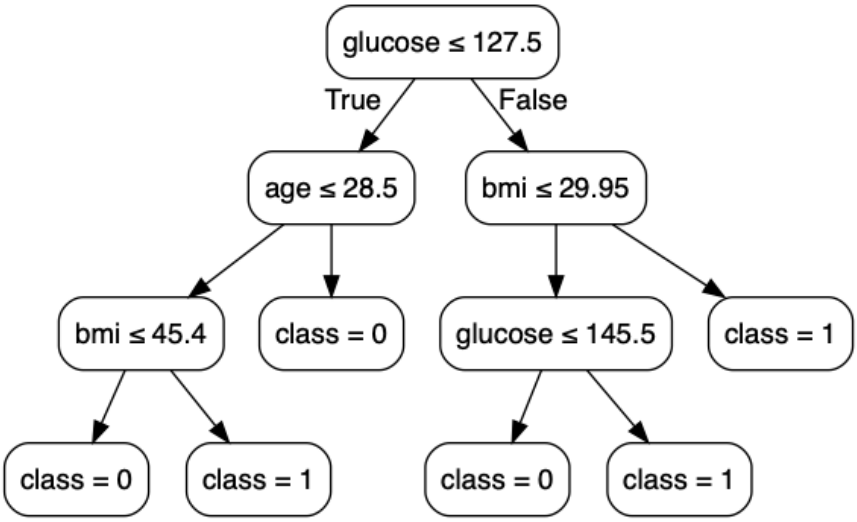
    ```
    g(f(1>2, 2>1), f(4**0.5==2, False))
    ```

    A. False B. True

Answer the next several questions about the following code, which attempts to implement the decision tree below (a 1 in the tree should respond to `predict3` returning `True`). Note that **the code contains 2 bugs**. Assume the `column_avg` function (not shown) is correctly implemented and that it returns the average of the given variable when it is called below.

```
def predict3(glucose=None, bmi=None, age=None):
    if glucose == None:
        glucose = column_avg("glucose")
    if bmi == None:
        bmi = column_avg("bmi")
    if age == None:
        age = column_avg("age")

    if glucose <= 127.5:
        if age <= 28.5:
            if bmi <= 45.4:
                return False
            else:
                return True
        else:
            return True
    else:
        if bmi <= 29.95:
            if glucose <= 145.5:
                return False
            else:
                return True
        else:
            pass
```



26. What will the following return? `predict3(100, 30, 20)`

A. False B. True

27. What will the following return (assume `column_avg("glucose")` returns 150)? `predict3(bmi=25)`

A. False **B. True**

28. How many default arguments will be used to initialize our parameters for this call? `predict3(glucose=120)`

A. 0 B. 1 **C. 2** D. 3

29. What test case would cause the buggy function to incorrectly return True, in contradiction with the decision tree at the bottom?)

A. `predict3(glucose=999, bmi=999, age=999)`

B. `predict3(glucose=-1, bmi=-1, age=-1)`

C. `predict3(glucose=120, bmi=30, age=20)`

D. `predict3(glucose=120, bmi=30, age=40)`

30. What is the technical term for the kind of error mentioned in the previous question?

A. unforgivable B. syntax C. runtime **D. semantic**

31. What test case is guaranteed to cause the buggy function to incorrectly return `None` (assuming we don't know what `column_avg returns for various inputs`)?

A. `predict3(glucose=999, bmi=999, age=999)`

B. `predict3(glucose=-1, bmi=-1, age=-1)`

C. `predict3(glucose=120, bmi=30, age=20)`

D. `predict3(glucose=120, bmi=30, age=40)`

E. `predict3()`

Battleship

Reference the following code for the next few questions. The code correctly draws a map, but does not draw an X where the user guesses due to a bug.

```
def draw(x, y, M=8, N=6):
    i = 0
    while i < M:
        j = 0
        while j < N:
            if y == i and x == j:
                print("X", end="")
            else:
                print(".", end="")
            j += 1
        print() # just a newline because end="\n" by default
        i += 1

x = input("enter x: ")
y = input("enter y: ")
draw(x, y)
```

32. Assume the user types `3` and `4.0` as input; what will the types of the values in global variables `x` and `y` be, respectively?

    A. int, int B. int, float C. float, float **D. str, str**

33. What call would print a map with an `X` in the top-left corner?

    A. `draw(0, 0)` B. `draw(1, 1)` C. `draw(M=1, N=1)` D. `draw("top", "left")`

34. Calling `draw(-1, -1)` prints the equvalent of what?

    A. `print("."*(8*6))`

    B. `print(("."*6 + "\n") * 8)`

    C. `print(("."*8 + "\n") * 6)`

    D. `print("X" + ("."*8 + "\n") * 6)`

    E. `print(("."*8 + "\n") * 6 + "X")`

35. Which parameter to `draw` represents the width of the map?

    A. `x` B. `y` C. `M` **D.** `N`

(Blank Page: you may tear this off for scratch work, but hand it in with your exam)

(Blank Page: you may tear this off for scratch work, but hand it in with your exam)