

CS 220 - Fall 2021
Instructors: Meenakshi Syamkumar, Andrew Kuemmel

Exam 2 — 10%

(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 11:00am (Meena)
 - 002 - MWF 1:20pm (Meena)
 - 003 - MWF 8:50am (Andy)
 - 004 - MWF 9:55am (Andy)
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 note sheet. 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 (including smart watches) 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

1. Which of the following is **not** a sequence?

A. tuple B. list C. string **D. dictionary**

2. What will be printed after this code runs?

```
s = "Nice day, isn't it?"
words = s.split(" ")
print(words.pop(-1))
```

A. ? **B. it?** C. isn't it? D. Nice E. Nice day, isn't

3. What will be in the file after this code runs?

```
f = open("file.txt", "w")
f.write("Hello")
f.close()
```

```
f = open("file.txt", "w")
f.write("World")
f.close()
```

```
f = open("file.txt")
data = f.read()
data = "!!!!!"
f.close()
```

A. !!!!! B. Hello **C. World** D. HelloWorld!!!! E. Hello World !!!!!

4. What is the output of the following code?

```
msg = "Hey220"
msg.upper()
for i in range(len(msg)):
    if msg[i].isdigit():
        print(msg[i - 2])
        break
```

A. 2 B. E **C. e** D. Hey220 E. HEY220

5. What will be printed after this code runs?

```
a = [1]
b = [2]
c = b
c.append(1)
c.sort()
a.append(b)
print(a, b)
```

- A. [1, [2]] [2]
- B. [1, 1, 2] [1, 2]
- C. [1] [1, 2]
- D. [1, [1, 2]] [1, 2]**
- E. [1, [2]] [2]

6. What is the output of the following code? Assume Python version is 3.6 or above.

```
d = {1: "one", 2: "two", 3: "three"}
d[0] = "zero"
d2 = dict()
for k in d:
    d2[d[k]] = k
print(len(d2), list(d2.keys())[0])
```

- A. 4 0 B. 4 1 C. 4 zero **D. 4 one**

7. Which of the following will access Jenny's math grade?

```
student = {}
student["name"] = "Jenny"
student["grades"] = {"math": 90, "biology": 95}
```

- A. grades["math"]
- B. grades["Jenny"]["math"]
- C. student["grades"]["math"]**
- D. student["Jenny"]["math"]

8. What is the output of the following code?

```
my_list = ["Alice", "Bob", "Evelyn"]
your_list = my_list[:]
my_list.append("Dennis")
your_list.pop(0)
print(your_list)
```

- A. ["Bob", "Evelyn"]
- B. ["Alice", "Bob", "Evelyn"]
- C. ["Bob", "Evelyn", "Dennis"]
- D. ["Alice", "Bob", "Evelyn", "Dennis"]

9. What is the output of the following code?

```
a = (1, 2, 3)
a[0] = a[1] + a[2]
print(a[1])
```

- A. 1 B. 2 C. 3 D. 5 **E. Nothing because the program crashes**

10. What is the value of x after the code is run?

```
import copy
orig = [1, (2, 3)]
x = orig
y = copy.copy(orig)
x[0] = 4
x[1] = (5, 6)
print(y)
```

- A. [1, (2, 3)] B. [1, (5, 6)] C. [4, (2, 3)] D. [4, (5, 6)]

11. What is the output?

```
def mystery(item_list):
    count = 0
    for item in item_list:
        if type(item) == list:
            count += mystery(item)
        else:
            count += 1
    return count

print(mystery([[1, 2], 3, [4, 5, 6], 7]))
```

A. 0 B. 4 **C. 7** D. 28

12. What is the output?

```
def foo(n):
    if n > 3:
        return n + foo(n - 1)
    return 0
print(foo(5))
```

A. 5 **B. 9** C. 12 D. 15

13. Which of the following will generate a list of words with a length longer than 7?

```
words = ["Psychology", "Bascom", "Noland", "Ingraham", "Soils"]
```

- A. `[len(w) for w in words if len(w) > 7]`
- B. `[len(words) for w in words if len(words) > 7]`
- C. `[words for w in words if len(words) > 7]`
- D. `[w for w in words if len(w) > 7]`**

14. Which of the following will sort words by the length of each word, with the longest word appearing first?

```
words = ["Psychology", "Bascom", "Noland", "Ingraham", "Soils"]
```

- A. `sorted(words, key = len, reverse = True)`**
- B. `sorted(words, key = len)`
- C. `sorted(words, key = lambda len, reverse = True)`
- D. `sorted(words, key = lambda len)`

15. Which of the following will generate a dictionary with name to age mapping?

```
players = [{"name": "Alia", "jersey": 17, "age": 20},  
           {"name": "Brianna", "jersey": 6, "age": 19},  
           {"name": "Carl", "jersey": 10, "age": 21} ]
```

- A. {k:v for (k, players["age"]) in players}
- B. {d["name"]:d["age"] for d in players}
- C. {d[0]:d[2] for d in players}
- D. {k:v for (k, v) in players.items()}

16. Which of the following will correctly sort players by jersey number?

```
players = [{"name": "Alia", "jersey": 17, "age": 20},  
           {"name": "Brianna", "jersey": 6, "age": 19},  
           {"name": "Carl", "jersey": 10, "age": 21} ]
```

- A. sorted(players, key = jersey)
- B. sorted(players.items(), key = jersey)
- C. sorted(players, key = lambda p:p["jersey"])
- D. sorted(players.items(), key = lambda p:p["jersey"])

Airbnb

For each question in this section, assume that the initial code executed is as follows:

```
airbnb = [  
    {"name": "A", "room_id": 837, "start_year": 2021, "reviews": [""]},  
    {"name": "B", "room_id": 389, "start_year": 2021, "reviews": ["*", "", "***"]},  
    {"name": "C", "room_id": 108, "start_year": 2021, "reviews": ["***", "***"]},  
    {"name": "D", "room_id": 237, "start_year": 2020, "reviews": ["*"]},  
    {"name": "E", "room_id": 278, "start_year": 2020, "reviews": ["***", "*", "***"]},  
]
```

17. What does the following evaluate to? `airbnb[-1]["reviews"][1]`

- A. "*" B. "***" C. ["***"] D. ["***", "*", "***"] E. "reviews"

18. What does the following evaluate to?

```
airbnb[2]["reviews"].index(airbnb[1]["reviews"][2])
```

- A. 0 B. 1 C. 2 D. "*" E. ""

19. What does the following evaluate to?

```
airbnb[1]["reviews"][:1] + airbnb[2]["reviews"][1:]
```

- A. ["*", "", "***"]
B. ["*", "***"]
C. ["*"]
D. {"reviews": ["*", "***']}

20. Which of the following would generate a list of airbnb listing names, which have room_id greater than 250?

- A. `{key: val for key, val in airbnb.items() if item["room_id"] > 250}`
B. `[item if item["room_id"] > 250 for item in airbnb]`
C. `[item["name"] for item in airbnb if item["room_id"] > 250]`
D. `[item for item in airbnb if item["room_id"] > 250]`

21. Which of the following will **result** possibly refer to? Assume Python version is 3.6 or above.

```
result = []
for x in airbnb[1]:
    result.append(x)
result
```

- A. ["r", "o", "o", "m", "_", "i", "d"]
- B. ["reviews", "start_year", "room_id", "name"]
- C. ["name", "room_id", "start_year", "reviews"]
- D. ["A", 837, 2021, [""]]
- E. ["A", 837, 2021, ""]

22. What will **counts** contain? Read the code carefully! **Note:** This question has an intentional semantic error.

```
counts = {}                # Line 1
for item in airbnb:        # Line 2
    reviews = item["reviews"] # Line 3
    for review in reviews:  # Line 4
        if review not in counts: # Line 5
            counts[review] = 0    # Line 6 semantic error
        else:                  # Line 7
            counts[review] += 1   # Line 8
```

- A. {"": 0, "*": 0, "**": 0, "***": 0}
- B. {"": 1, "*": 1, "**": 2, "***": 2}
- C. {"***": 2, "****": 2, "": 1, "*": 1}
- D. {"": 1, "*": 2, "**": 3, "***": 4}
- E. {"": 2, "*": 2, "**": 3, "***": 3}

23. How should line 6 be modified to fix the semantic error?

- A. counts[review] += 0
- B. counts[review] += 1
- C. counts[review] = []
- D. counts.append(review)
- E. counts[review] = 1

Assume that the initial code executed is as follows:

```
airbnb = [  
    {"name": "A", "room_id": 837, "start_year": 2021, "reviews": [""]},  
    {"name": "B", "room_id": 389, "start_year": 2021, "reviews": ["*", "", "**"]},  
    {"name": "C", "room_id": 108, "start_year": 2021, "reviews": ["***", "**"]},  
    {"name": "D", "room_id": 237, "start_year": 2020, "reviews": ["*"]},  
    {"name": "E", "room_id": 278, "start_year": 2020, "reviews": ["***", "**", "***"]},  
]
```

24. Which of the following will sort `airbnb` based on descending order of length of reviews?

- A. `sorted(airbnb, key = airbnb:len(d["reviews"]), reverse = True)`
- B. `sorted(airbnb, key = lambda d:len(d))`
- C. `sorted(airbnb, key = lambda d:len(d["reviews"]))`
- D. `sorted(airbnb, key = lambda d:len(d["reviews"]), reverse = True)`
- E. `sorted(airbnb, key = lambda airbnb:len(item) for item in airbnb)`

25. What will be printed? Read the code carefully!

```
buckets = {}  
bucket = []  
  
for item in airbnb:  
    start_year = item["start_year"]  
    if start_year not in buckets:  
        buckets[start_year] = []  
    buckets[start_year].append(item)  
print(len(buckets[2020]), len(buckets[2021]))
```

- A. 0 0 B. **2 3** C. 3 2 D. 5 0 E. 5 5

Hurricanes

For each question in this section, assume that the initial code executed is as follows:

```
from collections import namedtuple
Hurricane = namedtuple("Hurricane", ["name", "damage", "deaths", "mph"])
katrina = Hurricane(name = "Katrina", damage = "3.98B",
                    deaths = "1836", mph = "175")
baker = {"name": "Baker", "damage": "2.55M", "deaths": "38",
        "mph": "105"}
```

26. What does `katrina.deaths` evaluate to?
- A. "1836" B. 1836 C. "deaths" D. KeyError
27. Which of the two lines of code marked A and B will **not** execute successfully? Assume that if line A fails to execute, we comment it out and try to execute line B.

```
some_dict = {}
some_dict[katrina] = "Katrina - Category 5 hurricane"      # line A
some_dict[baker] = "Baker - Category 2 hurricane"          # line B
```

- A. line A **B. line B** C. Both line A and line B D. Neither A nor B
28. What will be printed after this code runs?

```
import copy

some_list = [baker, katrina]
new_list1 = copy.copy(some_list)
new_list1[0]["name"] = new_list1[0]["name"].upper()
new_list2 = copy.deepcopy(some_list)
new_list2[0]["deaths"] = int(new_list1[0]["deaths"])
print(some_list[0]["name"], type(some_list[0]["deaths"]))
```

- A. Baker <class 'str'>
B. Baker <class 'int'>
C. BAKER <class 'str'>
D. BAKER <class 'int'>

29. Which options sorts the `hurricanes` list in decreasing values of `mph`?

```
katrina = Hurricane("Katrina", "3.98B", "1836", "175")
sandy = Hurricane("Sandy", "2.34M", "546", "240")
baker = Hurricane("Baker", "2.55M", "38", "105")
hurricanes = [katrina, sandy, baker]
```

```
def extract(hurricane):
    return -int(hurricane.mph)
```

- A. `sorted(hurricanes, key = extract())`
- B. `sorted(hurricanes, key = extract(hurricane))`
- C. `sorted(hurricanes, key = extract)`
- D. `sorted(hurricanes, key = extract(hurricanes))`

Files

30. Which of the following are the delimiters of the rows in a CSV file?

- A. space **B. newline** C. comma D. semicolon

31. Why is the following JSON file formatted incorrectly?

```
{
  "Messi": {
    "Stats": [91, 93, 85],
    "Country": null,
    "Club": 'PSG'
  }
}
```

A. Strings must be enclosed within double quotations

B. null used instead of None

C. List items must be strings

D. Dictionary keys must be integers

32. What is the output of the following piece of code? Assume that the code does not crash due to exceptions. Also, assume that `some_file.txt` contains the following content.

```
I promise
to always
close my files

f = open('some_file.txt')
data = f.read()
lines = data.split('\n')
print(type(data), type(lines))
```

A. <class 'list'> <class 'list'>

B. class 'str'> <class 'str'>

C. <class 'list'> <class 'str'>

D. <class 'str'> <class 'list'>

Errors

33. What call to `foo` will **not** cause an `AssertionError`?

```
def foo(a, b):  
    assert type(a) == int and type(b) == int  
    assert a % 2 == 1 and a < b  
    return a + b
```

- A. `foo(12.0, 7.0)`
- B. `foo(17, 11)`
- C. `foo(12, 17)`
- D. `foo(11, 17)`**
- E. `foo(12, 17.0)`

34. What is the output of the following piece of code?

```
input_data = [12, 3, 0, 2]  
output_data = []  
for number in input_data:  
    try:  
        output_data.append(12 // number)  
    except:  
        output_data.append(-1)  
print(output_data)
```

- A. `[1, 4]`
- B. `[1, 4, -1]`
- C. `[1, 4, -1, 6]`**
- D. `[-1, 1, -1, -1]`
- E. `ZeroDivisionError`

35. What line is printed **last** when the following code is run?

```
my_list = [1.1, 0.3, 0.6]

for element in my_list:
    try:
        print("The element is:", element)
        reciprocal = 1 // int(element)
        print(reciprocal)
    except ZeroDivisionError as e:
        print("Oops!!!")
```

- A. The element is: 0.6
- B. The element is: 0.3
- C. The element is: 1.1
- D. ZeroDivisionError
- E. Oops!!!**

(Blank Page)

(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)