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

NetID (email): _____ @wisc.edu

(Blank Page)

Review

1. What does the following print?

```
x = 1
y = 2
def f(x):
    x *= 2**3
    return x + y
w = f(x)
v = f(y)
print(str(3*x) + str(v) + "!")
```

   A. "318!"   B. "2418!"   C. "3010!   D. "218!"

   For the next few questions consider the following code that prints a grid displaying the areas where hurricanes commonly form in North-East Kansas. In particular, to take the border into account, the grid just contains the entries of the lower triangle of the whole grid. An "H" represents a hurricane usually forms in this area and a "." means hurricanes avoid this area. The only input is the size of the grid we want patterns for.

```
def radar(n=5):
    for i in range(n):
        for j in range(i):
            if (i + j) % 2 == 0:
                print("H", end = "")
            else:
                print(".", end = "")
        print() # default end is "\n"
```

2. The first line of output from `radar()` corresponds to which string?

   A. "\n"   B. ".\n"   C. "H\n"   D. ".H\n"

3. What characters appear on the top diagonal of the shape printed by `radar(5)`?
   (ignore lines that are just white space)

   A. 4 periods   B. 4 H's   C. 2 periods and 2 H's

4. What characters (ignoring whitespace) would be printed by `radar()` in the first column if the programmer had forgotton the parantheses in the condition in the radar function so that that line instead looked like this: `if i + j % 2 == 0:`

   A. 4 periods   B. 4 H's   C. 2 periods and 2 H's

"Fancy" Functions

```python
def curses(x):
    if x == 0:
        return 1
    else:
        return curses(x-1) * 2

def swap(lis, x, y):
    tmp = lis[y]
    lis[y] = lis[x]
    lis[x] = tmp

def mix(a_list, x):
    swap(a_list, x, x-1)
    if x > 0:
        mix(a_list, x-1)
```

5. Which statement creates a new reference to the **curses** function object?

   A. f := curses   **B. f = curses**   C. f = curses()   D. f = curses(x)   E. f = curses(5)

6. What does **curses(5)** return?

   A. 1   B. 10   C. 16   D. 8   **E. 32**

7. Which call causes a stack overflow (meaning we create too many frames)?

   A. curses(0)   B. curses(0.0)   C. curses(1/0)   **D. curses(-1)**   E. curses(None)

8. What does **my_list** look like after the following code?

   ```python
   my_list = ["A", "B", "C"]
   mix(my_list, 2)
   ```

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

9. The following code runs without an error/exception. Is **g** *iterable*? An *iterator*?

   ```python
   g = map(int, [1.1, 2.2, 3.3])
   first = next(g)
   it = iter(g)
   ```

   A. neither   B. iterable only   C. iterator only   **D. both**

Copying Movies

For the following, assume the initial code executed is as follows (if a question contains code that modifies the objects, those changes **should not** be considered in other questions):

```
import copy
genres = ["g1", "g2", "g3"]
movies = [
{"title": "A", "year": 17, "style": "short", "genres": ["g1"]},
{"title": "B", "year": 18, "style": "long", "genres": ["g2"]},
{"title": "C", "year": 19, "style": "short", "genres": ["g3"]},
{"title": "D", "year": 19, "style": "long", "genres": ["g1", "g2", "g3"]}
]
def first_n(movies, n):
    while len(movies) > n:
        movies.pop() # by default, removes last item
    return movies
```

10. What does the following print?

```
genres_new = genres
genres.remove("g3")
genres_new.remove("g1")
print(genres)
```

   A. ["g1", "g2"]   B. ["g1", "g2", "g3"]   **C.** ["g2"]   D. ["g2", "g3"]

11. What does the following print? Be careful!

```
movies1 = first_n(movies, 2)
movies2 = first_n(movies, 3)
print(len(movies1), len(movies2))
```

   A. 0 0   B. 2 0   **C.** 2 2   D. 2 3   E. 4 4

12. What does the following print?

```
cp = copy.copy(movies) # shallow copy
movies.append(0)
movies[0]["year"] = 16
print(cp[0]["year"], len(cp))
```

   A. 16 4   B. 16 5   C. 17 0   D. 17 4   E. 17 5

13. If `copy.copy` were replaced with `copy.deepcopy` in the previous question, what would the code print?

   A. 16 4   B. 16 5   **C.** 17 4   D. 17 5   E. 17 6

Files

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

```
f = open("file.txt", "w")
f.write("hi")
f.close()
f = open("file.txt", "w")
f.write("I love")
f.write("Python")
f.close()
```

    A. `"Python"`   **B. `"I lovePython"`**   C. `"I love Python"`   D. `"hi\nI love\nPython"`

15. What is a good reason to serialize Python data structures to a JSON file instead of just using the data structures?

    A. Files remain after rebooting   B. Files are faster   C. JSON supports more types

16. A key in a Python dict may be an integer (T/F). **A. True**   B. False

17. A key in a JSON dict may be an integer (T/F). A. True   **B. False**

18. In which format is everything a string? **A. CSV**   B. JSON   C. Excel spreadsheet

19. Rows of a CSV are normally delimited by which of the following?

    A. commas   B. semicolons   C. quotes   **D. new lines**

20. Which of the following is a valid boolean **in JSON**?

    A. `true`   B. `True`   C. `TRUE`   D. `absolutely`

21. What is the **best** way to get the "C" value from `row`?
    ("best" means most readable by another programmer)

```
header = ["A", "B", "C"]
row = [9, 8, 7]
```

    A. `row[-1]`   B. `row["C"]`   C. `row[header["C"]`   **D. `row[header.index("C")]`**

Data Structures

```
from collections import namedtuple
Player = namedtuple("Player", ["name", "rating", "club", "country"])

hazard = Player(name="Eden Hazard", rating=91,
                club="Chelsea", country="Belgium")
pulisic = Player(name="Christian Pulisic", club="Chelsea",
                 rating=79, country="United States")
messi = Player(name="Lionel Messi", club="Barcelona",
               country="Argentina", rating=94)
players = ["Kevin De Bruyne", "Christiano Ronaldo",
           "Harry Kane", "Hugo Lloris", "Don Smart"]
salah = {}
salah["name"] = "Mo Salah"
salah["nationality"] = "Egypt"
salah["jersey"] = 11
salah["nationality"] = "Egypt"
```

22. What does `pulisic.club` evaluate to?

    A. "Chelsea"   B. Error   C. 79   D. club   E. rating

23. How can we switch Eden Hazard's club to Real Madrid?

    A. `hazard.club = "Real Madrid"`

    B. `hazard[1] = "Read Madrid"`

    C. `hazard["club"] = "Read Madrid"`

    D. `hazard = Player(name=hazard.name, rating=hazard.rating, club="Real Madrid", country=hazard.country)`

24. How can we get only the first 3 players in `players`?

    A. players[3]   **B. players[0:3]**   C. players[0:2]   D. players(3)   E. players[0, 1, 2]

25. What is `len(salah)`? Careful!

    A. 3   B. 4   C. 5   D. 6   E. 8

26. Which data type would you **not** be able to use if you wanted to store hazard, pulisic, and messi in a variable called data and access them using `data[0]`, `data[1]`, and `data[2]`?

    A. dict   B. list   **C. set**   D. tuple

Errors

For the following, assume the initial code executed is as follows (if a question contains code that modifies the objects, those changes **should not** be considered in other questions):

```
gens = ["g1", "g2", "g3"]
movs = [{"title": "A", "year": 17, "style": "short", "genres": ["g1"]}]

def chk(movies, year):
    assert type(year) == int
    assert type(movies[0]) == dict
    # TODO: finish this function
```

27. Which call will **NOT** cause an AssertionError?

    A. `chk(gens, "18")`   B. `chk(gens, 18)`   C. `chk(movs, "18")`   **D.** `chk(movs, 18)`

28. What does the following code print? **BAD QUESTION, DROPPED**

```
new_genres = ["g4", "g5"]
try:
    for i in range(4):
        genres.append(new_genres[i])
except:
    genres.append(0)
print(len(genres))
```

   A. 3   B. 4   C. 5   D. 6   E. 7

29. What should replace `????` to trigger the most informative exceptions when year isn't an int?

```
def filter_by_year(movies, year):
    if type(year) != int:
        ????
    # TODO: finish writing this function
```

    A. `return TypeError("year shoud be int")`

    B. `return ValueError("year shoud be int")`

    C. `raise TypeError("year shoud be int")`

    D. `raise ValueError("year shoud be int")`

Complicated Lookups

For the following, consult the following diagram of objects and references. Assume any code in the questions can access variables in both frames.
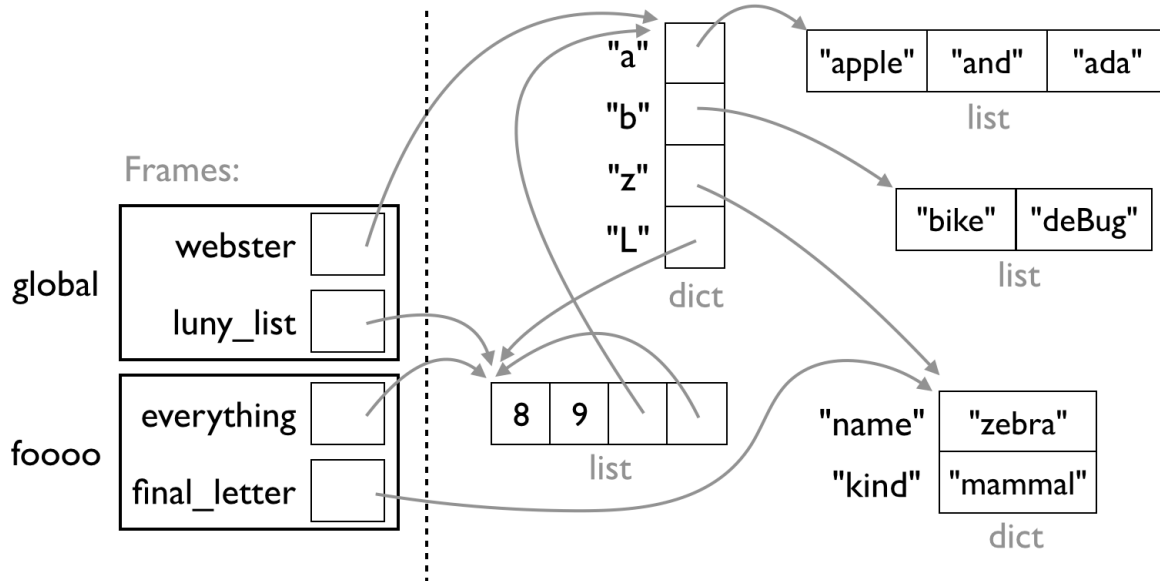
30. What does the following evaluate to? `"apple" in webster`

    A. True   **B. False**

31. `luny_list` refers to the same object as which of the following?

    A. `luny_list[0]`   B. `luny_list[0][0]`   C. `luny_list[3][0]`   **D. `luny_list[3][3]`**

32. Which of the following is True?

    A. `everything[0] is everything[1]`
    B. `luny_list is webster`
    C. `webster["b"] is ["bike", "deBug"]`
    D. `luny_list is everything[3][2]["L"]`

33. What is the type of `webster["L"][2]["z"]["name"]`?

    A. int   **B. str**   C. list   D. dict

34. What is the value of `luny_list[2]["a"][1][:3]`?

    A. `"and"`   B. `"app"`   C. `"bike"`   D. `"mam"`

35. What is `str(final_letter["kind"]) + str(webster["L"][1])`?

    A. `"appleand"`   B. `"bike9"`   **C. `"mammal9"`**   D. `"zebramammal"`

(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)