Name: _____

This test has 16 questions across 11 pages worth a total of 118 points, and is to be completed in 120 minutes. Point totals are labeled next to each question. The exam is closed book, and no calculators or other electronic devices are permitted.

Write the statement out below in the blank provided and sign your name. You may do this before the exam begins.

"I have neither given nor received any assistance in the taking of this exam."

_____

_____

Signature: _____

Tips:

- There may be partial credit for incomplete answers. Write as much of the solution as you can, but bear in mind that we may deduct points if your answers are much more complicated than necessary.
- Use your time wisely. If you are having too much trouble on a question, skip it and return to it later. **Work through the ones you are comfortable with first. Do not get overly captivated by interesting design issues or complex corner cases you're not sure about.**
- If you are given skeleton code to fill in, we will not grade your answer if you alter the skeleton code or write outside the solution box.
- For questions with *square checkboxes*, you may select *more than one* choice.
- For questions with *circles*, you may *only select one* choice.
- Relax, and try to have fun with it! Think of this exam as a checkpoint along your JamCoders journey. You got this!

Staff use only.

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	
Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Total

1. ***(6 points)*** What will be printed after running the following code? Write your answers on the dashed lines.

```
a = 1
b = 2
c = 3
a = a - b + c
print(a)            # Printed: _____________________________________
c = a * c + 5
print(c)            # Printed: _____________________________________
a = c // a
print(a)            # Printed: _____________________________________
```

2. ***(4 points)*** What is $\log_8(64)$?

- ○ 1.73
- ○ 2
- ○ 3
- ○ 8
- ○ None of the above

3. ***(4 points)*** What will be printed after running the following code? Write your answers on the dashed lines.

```
A = [4, 0, 8]
B = [4, 1, 2]
L = A + B[:2]
print(L[4])         # Printed: _____________________________________
L = A[1:] + L
print(L[4])         # Printed: _____________________________________
```

4. ***(4 points)*** What will be printed after running the following code?

```
def mystery(L):
    total = 0
    for x in range(len(L)):
        total += x
    return total

print(mystery([3, 1, 0, 5]))
```

- ○ 0
- ○ 2
- ○ 6
- ○ 9

5. Adrianna wants to use binary search to find the number `3` in the list `[0, 4, 8, 7, 17, -9, -26]`.

Before running binary search, she must sort the array. Help her answer the following questions:

a. ***(4 points)*** What is the runtime of selection sort?

- ○ $O(1)$
- ○ $O(n)$
- ○ $O(n \log n)$
- ○ $O(n^2)$

b. ***(4 points)*** What is the runtime of merge sort?

- ○ $O(\log n)$
- ○ $O(n)$
- ○ $O(n \log n)$
- ○ $O(n^2)$

Now, she wants to run binary search on the ***sorted*** list.

c. ***(4 points)*** What is the ***index*** of the first number she guesses?

- ○ `-9`
- ○ `0`
- ○ `1`
- ○ `2`
- ○ `3`
- ○ `4`
- ○ `7`

d. ***(4 points)*** After guessing the previous number, what are the new left and right indices of the search range?

- ○ left = `0`, right = `6`
- ○ left = `0`, right = `2`
- ○ left = `1`, right = `3`
- ○ left = `4`, right = `6`

e. ***(4 points)*** Does Adrianna find the number `3` in the list?

- ○ Yes, at index `0`
- ○ Yes, at index `2`
- ○ Yes, at index `3`
- ○ No, the search returns `-1`

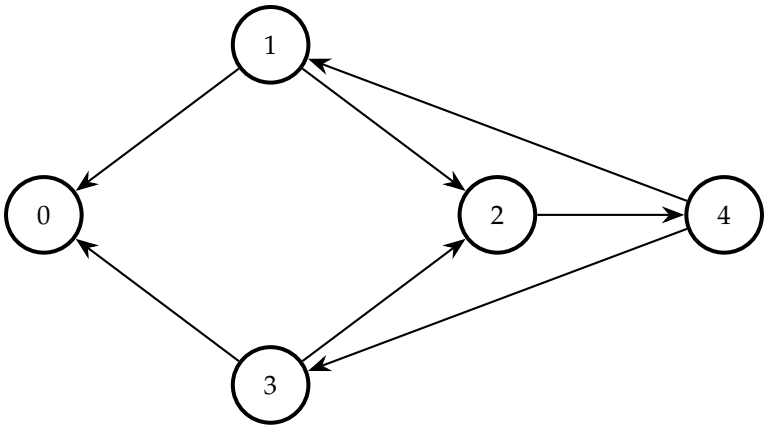
f. ***(4 points)*** What is the worst-case time complexity of binary search on a sorted list of size n?

- ○ $O(1)$
- ○ $O(\log n)$
- ○ $O(n)$
- ○ $O(n^2)$

6. ***(4 points)*** Beza has two sorted lists of integers A and B and does $A + B$ to get a new list of integers C. Beza now wants to run binary search to find if the number 22 exists in C. Is this a good idea, and why?

- ○ Yes, because binary search runs in $O(\log n)$ time.
- ○ Yes, because it is better to run binary search for very long lists.
- ○ No, because C is not necessarily sorted.
- ○ No, because C will be too long to run binary search.

7. Consider the following graph.



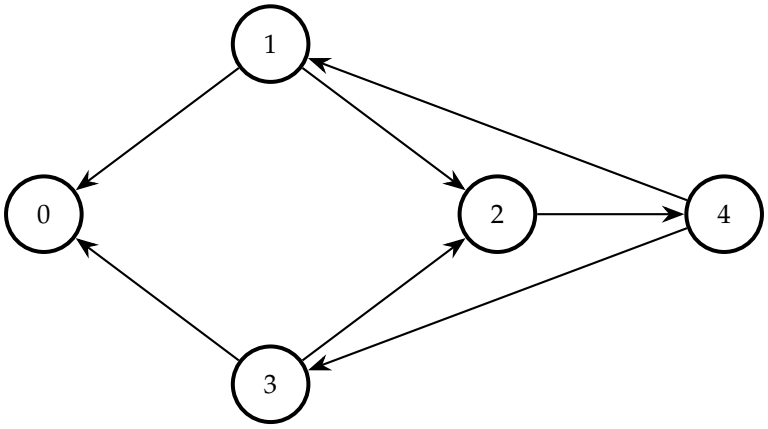
a. ***(4 points)*** Mark all the neighbors of the vertex 1.

- ☐ 0
- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

b. ***(4 points)*** True or False: `0` is connected to `2`.

- ○ True
- ○ False

The graph is shown again for ease of reference.



c. ***(4 points)*** Does this graph contain any cycles? If so, write out one valid cycle as a sequence of vertices.

d. ***(5 points)*** Complete the following code so that after it runs, the variable `G` stores the ***adjacency list*** of the graph.

```
G = _______________________________

    _______________________________

    _______________________________

    _______________________________

    _______________________________

    _______________________________

    _______________________________
```

8. ***(8 points)*** Fill out the recursive function `product_except_zero` which takes in a list of integers `L` and returns the product of all nonzero integers in `L`.

Example: `product_except_zero([1, 5, 0, 3, 2, 0])` should return `30`.

```
def product_except_zero(L):

    if _____________________________________________:
        return ________________________________________

    if _____________________________________________:
        return ________________________________________________

    else:
        return ________________________________________________
```

9. ***(6 points)*** Consider the following three functions:

$$f(n) = n^2$$
$$g(n) = n + 10000$$
$$h(n) = 2^{0.01n}$$

Rank these functions in order of asymptotic growth rate by filling in the blanks using f, g, and h. Use each exactly once:

As n gets large, _____(n) grows larger than _____(n), which grows larger than _____(n).

10. ***(5 points)*** What will be printed after running the following code?

```
def ackee(n):
    if n <= 1:
        return 1
    return ackee(n // 2) + ackee(n // 2)

print(ackee(8)) # Printed: ____________________________________
```

11. ***(4 points)*** What is the running time of the following code in terms of `n`?

```
x = 0
for i in range(0, n, 2):
    for j in range(i / 2):
        for k in range(10000000000):
            x += 1
```

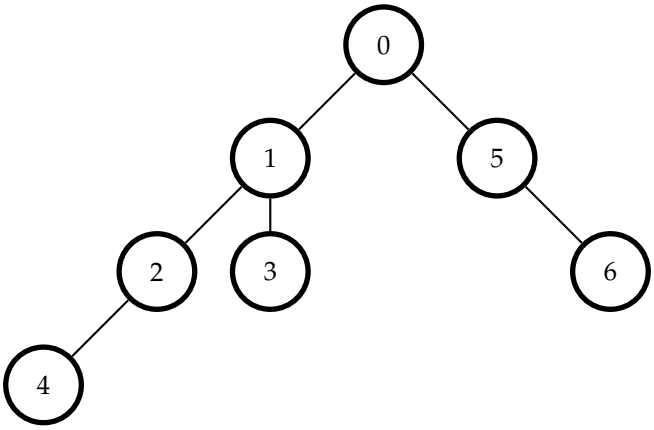
☐ $O(n \log n)$

☐ $O(n^2)$

☐ $O(n^3)$

☐ $O(10000000000)$

12. After cheerleading practice, Anushka runs search on the following graph, starting from node `0`. Assume we break ties in numerical order.



a. ***(5 points)*** If Anushka runs ***depth-first search***, which node will be visited the last?

- ○ `0`
- ○ `1`
- ○ `2`
- ○ `3`
- ○ `4`
- ○ `5`
- ○ `6`

b. ***(5 points)*** If Anushka runs ***breadth-first search***, which node will be visited the last?

- ○ 0
- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ○ 5
- ○ 6

13. Write a function `remove_character(s, c)` that takes in a string `s` and returns a new string removing all the characters in `s` that match `c`.

Examples:

- `remove_character("banana", "a")` returns `"bnn"`
- `remove_character("Mississippi", "s")` returns `"Miiippi"`
- `remove_character("computer", "x")` returns `"computer"`
- `remove_character("", "a")` returns `""`

a. ***(6 points)*** First, solve the problem *iteratively*.

```
def remove_character(s, c):
        result = ______________________
        for idx in range(_________________________):
            if ___________________ != ___________________:
                ________________ += ____________________________
        ______________________________
```

b. ***(8 points)*** Next, solve the problem *recursively*.

```
def remove_character(s, c):
        if ___________________________________:
            return ____________________________
        if ___________________________________:
            return ____________________________
        return ______________________________
```

14. ***(8 points)*** Tom the cat is climbing the stairs of Rex Hall. On each step, there are a certain number of pieces of fried chicken. Tom may choose to eat all the chicken on a step, but if he does, he won't be able to eat any chicken on the next step. He may eat chicken again starting two steps later (or any later step).

Write a recursive function `chicken` that takes a list of integers `stairs`, where each integer represents the number of pieces of fried chicken on a stair, and returns the maximum number of pieces of chicken Tom can eat.

Hint: `max(a, b)` returns the larger of the two values `a` and `b`.

```
def chicken(stairs):
    if _______________________________:
        return _______________________________

    eats = _______________________________
    does_not_eat = _______________________________

    return ____________________________________________________________
```

15. ***(10 points) Optional*** **Challenge Problem:** *Finish other problems before attempting.*

Write a function `tournament` that takes in a list of teams called `team_lst`, prints out every game that happens if we form a tournament with the teams in `team_lst`, and returns the winner of the tournament. Assume that the number of teams is a power of two and is at least two.

We have access to a function called `winner` that takes in the names of two teams and returns the winner of a game between the two teams.

Example:

Suppose the following outcomes are returned by `winner`:

- `winner("Andrey", "Anushka")` returns `"Anushka"`
- `winner("Beza", "Daniel")` returns `"Beza"`
- `winner("Mallika", "Robert")` returns `"Robert"`
- `winner("Frank", "Sarika")` returns `"Frank"`
- `winner("Anushka", "Beza")` returns `"Beza"`
- `winner("Robert", "Frank")` returns `"Frank"`
- `winner("Beza", "Frank")` returns `"Frank"`

Now, suppose we also have the following code:

```
print("The winner is", tournament(["Andrey", "Anushka", "Beza", "Daniel",
                                   "Mallika", "Robert", "Frank", "Sarika"]))
```

Executing the line above produces the output:

```
Andrey plays Anushka
Beza plays Daniel
Mallika plays Robert
Frank plays Sarika
Anushka plays Beza
Robert plays Frank
Beza plays Frank
The winner is Frank
```

Write your implementation of the `tournament` function on the next page.

16. ***(10 points) Optional*** **Challenge Problem:** *Finish other problems before attempting.*

Write a function `callaloop(x)` that takes a positive integer x and returns the largest positive integer y such that

$$y^y \leq x.$$

Use binary search to find the answer.