Practice Questions on Count (Week 1)

The following questions are taken from "Rosen Ch 6 Counting.pdf", Chapters 6.1 and 6.3.

Tier-1 Questions

1. (Chp6.1, Q3) A MCQ test contains 10 questions. There are 4 possible answers for each question.
    a. In how many ways can a student answer the questions on the test if the student answers every question?
    b. In how many ways can a student answer the questions on the test if the student can leave answers blank?

2. (Chp6.1, Q23) How many positive integers between 100 and 999 inclusive:
    a. are divisible by 7?
    b. are odd?
    c. have the same three decimal digits?
    d. are not divisible by 4?
    e. are divisible by 3 or 4?
    f. are not divisible by either 3 or 4?
    g. are divisible by 3 but not by 4?
    h. are divisible by 3 and 4?

3. (Chp6.1, Q29) How many license plates can be made using either
    - 2 uppercase English letters followed by 4 digits or
    - 2 digits followed by 4 uppercase English letters?

4. (Chp6.1, Q45) How many ways are there to seat six people around a circular table where two seatings are considered the same when everyone has the same two neighbors without regard to whether they are right or left neighbors?

5. (Chp6.1, Q47) In how many ways can a photographer at a wedding arrange six people in a row, including the bride and groom, if:
    a. the bride must be next to the groom?
    b. the bride is not next to the groom?
    c. the bride is positioned somewhere to the left of the groom?

6. (Chp6.3, Q3) How many permutations of $\{a, b, c, d, e, f, g\}$ end with a?

7. (Chp6.3, Q13) A group contains n men and n women. How many ways are there to arrange these people in a row if the men and women alternate?

8. (Chp6.3, Q19) A coin is flipped 10 times where each flip comes up either heads or tails. How many possible outcomes
    a. are there in total?
    b. contain exactly two heads?
    c. contain at most three tails?
    d. contain the same number of heads and tails?

9. (Chp6.3, Q23) How many ways are there for 8 men and 5 women to stand in a line so that no two women stand next to each other? *Hint*: First position the men and then consider possible positions for the women.

10. (Chp6.3, Q27) A club has 25 members.
    a. How many ways are there to choose 4 members of the club to serve on an executive committee?
    b. How many ways are there to choose a president, vice president, secretary, and treasurer of the club, where no person can hold more than one office?

11. 19 SMU students are going to Cathay Cinema
    a. There are 22 seats in each row. The students can pick any seat in row A. How many ways are there for these 19 students to occupy the row?
    b. Andrew, Brandon, and Cheryl decide to join the group of 19 students. Brandon prefers to sit at either end of the row A. How many ways are there to arrange the students on the same row of 22 seats so that Brandon's preference is satisfied?

12. There are 10 men and 10 women.
    a. How many ways are there to arrange these people in a row?
    b. How many ways are there to seat these 20 people, if there are two different circular tables? One table is blue, and the other table is red. Each table has 5 men and 5 women alternating. Two seatings for a table are considered the same when everyone on that table has the same two neighbors without regard to whether they are right or left neighbors.

Tier-2 Questions

13. (Chp6.1, Q5) 6 different airlines fly from New York to Denver and 7 fly from Denver to San Francisco. How many different pairs of airlines can you choose on which to book a trip from New York to San Francisco via Denver, when you pick an airline for the flight to Denver and an airline for the continuation flight to San Francisco?

14. (Chp6.1, Q24) How many positive integers between 1000 and 9999 inclusive
    a. are divisible by 9?
    b. are even?
    c. have distinct digits? (e.g. 123 has distinct digits, but 112 does not have distinct digits)
    d. are not divisible by 3?
    e. are divisible by 5 or 7?
    f. are not divisible by either 5 or 7?
    g. are divisible by 5 but not by 7?
    h. are divisible by 5 and 7?

15. (Chp6.1, Q30) How many license plates can be made using either three uppercase English letters followed by three digits or four uppercase English letters followed by two digits?

16. (Chp6.1, Q44) How many ways are there to seat four of a group of ten people around a circular table where two seatings are considered the same when everyone has the same immediate left and immediate right neighbor?

17. (Chp6.1, Q46) In how many ways can a photographer at a wedding arrange 6 people in a row from a group of 10 people, where the bride and the groom are among these 10 people, if
    a. the bride must be in the picture?
    b. both the bride and groom must be in the picture?
    c. exactly one of the bride or the groom is in the picture? (both cannot be in together)

18. (Chp6.3, Q7) Find the number of 5-permutations of a set with nine elements.

19. (Chp6.3, Q15) In how many ways can a set of five letters be selected from the English alphabet?

20. (Chp6.3, Q18) A coin is flipped eight times where each flip comes up either heads or tails. How many possible outcomes
    a. are there in total?
    b. contain exactly three heads?
    c. contain at least three heads?
    d. contain the same number of heads and tails?

21. (Chp6.3, Q24) How many ways are there for 10 women and six men to stand in a line so that no two men stand next to each other?

22. (Chp6.3, 33) Suppose a department contains 10 men and 15 women. How many ways are there to form a committee with 6 members if it must have the same number of men and women?

23. In a class of Computational Thinking in year 2021, there are 20 SCIS students and 10 non-SCIS students.
    a. How many ways are there to choose a single group of three (3) members with at least one non-SCIS student?
    b. How many ways are there to form 10 groups of 3 members for doing projects with at least one non-SCIS student in each group?

24. Singapore's national football team is one of the 11 teams competing at the football tournament in the 2021 Southeast Asian Games.
    a. In the qualification round, the 11 teams are divided into two groups: Group A (5 teams) and Group B (6 teams) respectively. Every team will compete once with each of the other teams in the same group.
        i. How many ways are there to divide the 11 teams into the two groups?
        ii. How many matches are there in the qualification round?

    b. Singapore football team has 21 members, including 3 goalkeepers, 8 defenders, 6 midfielders and 4 forwards. The coach applies the 4-4-2 formation, which has:
        - 2 left defenders (LD),
        - 2 right defenders (RD),
        - 2 left midfielders (LM),
        - 2 right midfielders (RM),
        - 1 left forward (LF),
        - 1 right forward (RF) and
        - 1 goalkeeper,

        to form the playing team of 11 players. How many ways are there to form the playing team, if the left side and right sides are not considered the same position?

    c. Before its first match begins, the playing team of 11 players huddles in a circle. How many ways are there to create the circle, such that the goalkeeper will not be standing immediately next to any forward? Left and right neighbors are considered different.

~End

Tutorial Questions on Complexity (Week 2)

Tier-1 Questions

1. The table below shows the number of times an operation needs to be performed for the corresponding value of $\boldsymbol{n}$ (the input size) of a particular algorithm. What is the Big O complexity of the algorithm?

n	Number of times
1	10
2	11
3	12
4	13
5	14
6	15
7	16
…	
20	29
21	30

n	Number of times
22	31
23	32
…	
500	509
501	510
502	511
…	
1000	1009
1001	1010
1002	1011

2. The following expressions provide the number of steps taken by an algorithm to solve a problem of size **N**. For each expression, specify the Big O complexity.

	Number of steps	Big O
a	$9 + 0.02N^2 + 0.1N$	
b	$N^2 + 2N^{-3}$	
c	$N! + 100N^{20}$	
d	$2^N + N!$	
e	$5N(\log_2 N) + N \times \text{sqrt}(N)$	
f	$N^2(\log_2 N) + N(\log_2 N)^2$	
g	$10N^2\log(N) + 5N^3 + N^{\log(N)}$	
h	$10^5 + 10^4(\log(N))^2 + 10^3\log(N^2)$	

3. Determine the Big O complexity of the following Python functions.

   a. Two loops in a row:

```
def func1(n, m):
  for i in range(n):
    # sequence of statements
  for j in range(m):
    # sequence of statements
  return
```

b. A nested loop followed by a non-nested loop:

```
def func2(n):
  for i in range(n):
    for j in range(n):
      # sequence of statements
  for k in range(n):
    # sequence of statements
  return
```

c. A nested loop in which the number of times the inner loop executes depends on the value of the outer loop index:

```
 def func3(n):
  for i in range(n):
    j = n
    while j > i:
      # sequence of statements
      j -= 1
  return
```

4. Provide the Big O complexity for each of the three functions below.

a.
```
def f(n):
  sum = 0
  for i in range(1, n):
    sum = sum + i
  return sum
```

b.
```
def g(n):
  sum = 0
  for i in range(1, n):
    sum = sum + i + f(i)
  return sum
```

c.
```
def h(n):
  return f(n) + g(n)
```

5. Given the function below:

```
 def func_a(n):
   for i in range(n):
     for j in range(n):
       f(j)
```

a. If the execution time of function **f(j)** is a constant **T** (independent of its parameter **j**), what is the complexity of **func_a**?
b. If the execution time of function **f(j)** is linear in its parameter **j** (for simplicity, assume that the execution time of **f(j)** is simply **j**), what is the complexity of **func_a**? (You should have both the count

of time units and the corresponding Big O notation)

6. Given that **sub_func** is a function that takes an integer n as input and has the complexity of $O(n)$.

   a. What is the Big O complexity of **func_6a**?

   Hint: $\sum_{i=1}^{n} i = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}$

```
01: def func_6a(n):
02:   for i in range(n):
03:     for j in range(i):
04:       sub_func(n)
05:   return
```

   b. What is the Big O complexity of **func_6b**?

   Hint: $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$

```
01: def func_6b(n):
02:   for i in range(1, n+1):
03:     for j in range(1, i*i + 1):
04:       sub_func(n)
05:   return
```

   c. What is the Big O complexity of **func_2c**?

   Hint: $\sum_{i=1}^{n} \frac{1}{i} = 1 + \frac{1}{2} + \cdots + \frac{1}{n} \approx O(\log n)$

```
01: def func_2c(n):
02:   for i in range(1, n+1):
03:     for j in range(1, n+1):
04:       if j % i == 0:
05:         for k in range(1, n+1):
06:           sub_func(n)
07:   return
```

Tier-2 Questions

7. The table below shows the number of times an operation needs to be performed for the corresponding value of n (the input size) of a particular algorithm. What is the Big O complexity of this algorithm?

n	Number of times
1	1
2	2
3	3
4	4
5	5
6	6
…	
20	50
21	50
22	50

n	Number of times
23	50
…	
500	50
501	50
502	50
…	
1000	50
1001	50
1002	50
Other numbers >1002	50

8. The following expressions provide the number of steps taken by an algorithm to solve a problem of size **N**. For each expression, specify the simplified Big O complexity.

	Number of steps	Big O
a	$2N^2 + 2N^3 + 3N^4$	
b	$N^2 \times 2N^{-3}$	
c	$N! \times 100N^{20}$	
d	$5 \times (2N)!$	
e	$N(\log_2 N) + N(\log_3 N) + N(\log_4 N)$	
f	$N(\log_2 (2N))$	
g	$1000 + 5\log(N) + 2N + N^2 + 2^{2N}$	
h	$\log(N^5) + \log_5(N) + 5N$	

9. Determine the Big O complexity of the following Python functions.

a.
```
def q1(n):
    for j in range(n):
      f(j) # f takes constant time
    return
```

b.
```
def q2(n):
    for j in range(n):
      g(j) # g takes time linear in the value of its parameter
    return
```

c.
```
def q3(n, k):
    for j in range(n):
      g(k) # g takes time linear in the value of its parameter
    return
```

10. Provide the complexity of **my_func** for the following algorithms.
    a. Given that **sub_func** is a function that takes an integer n as input and has a complexity of $O(n)$, what is the Big O complexity of **my_func**?

```
def my_func(n):
  for i in range(n):
    if i%2==0:
       sub_func(n)
  return
```

    b. Given that **sub_func** is a function that takes an integer n as input and has a complexity of $O(\log n)$, what is the Big O complexity of **my_func**?

```
def my_func(n):
  i = 2
  while i < n:
    i += 1
    sub_func(n)
  return
```

    c. Given that **sub_func** is a method that takes an integer n as input and has a complexity of $O(n)$, what is the Big O complexity of **my_func**?

```
def my_func(n):
  for i in range(n):
    sub_func(i)
  return
```

11. You work for a company that manufactures cups. To start cup manufacturing you need to set four switches [switch1, switch2, switch3, switch4] to a certain position. Every switch only has two possible positions, "ON" or "OFF". (For example, this is one of the valid positions: ["ON", "OFF", "ON", "OFF"]). Unfortunately, the lead engineer is on leave, and only he knows the right combination of switches to get the machines started.

    Suppose you have a function **start(combination_array)** that returns **True** if the specified input switch positions can get the machines started, and **False** otherwise.

    a. Provide an algorithm in pseudocode that prints out the right combination of switch positions.
    b. If only a call to the **start** function is considered as one operation, what is the worst-case complexity of an algorithm that takes as input the number of switches N, and prints out the right combination of N switches?
    c. After trying some positions of switches, you realize that two (or more) consecutive "ON", or two (or more) consecutive "OFF" would cause the system to overheat. For example, the following switch positions would cause overheating:

        - ["ON", "ON", "ON",..., "OFF"]
        - ["ON", "OFF", "OFF",..., "ON"]

        What is the complexity of an algorithm that calls the **start** function only for those combinations of N switches that would not cause overheating? Note that only a call to the **start** function is considered as one operation.

12. An array **a** has **n** distinct numbers (i.e., there are no duplicate numbers). The following algorithm finds the **k**th smallest number of an array (1≤k≤n). For example, given array **a** = [14, 12, 26, 19] and **k** = 3, the algorithm returns the 3rd smallest number which is 19.

```
01  def find_minimum(a,k):
02    min = a[0]
03    max = a[0]
04
05    # part (a)
06    for j in range(1, len(a)):
07      if min > a[j]:
08        min = a[j] # find minimum number
09      if max < a[j]:
10        max = a[j] # find maximum number
11
12    # part (b)
13    if k <= 1:
14      return min
15    if k >= len(a):
16      return max
17
18    # part (c)
19    for i in range(2, k+1):  # find ith minimum no. of a
20      ith_min = max
21
22      # part (c1)
23      for j in range(len(a)):
24        if a[j] < ith_min and a[j] > min:
25          ith_min = a[j]
26
27      min = ith_min
28      print(i)            # print the value of i
29      print(ith_min)      # print the value of ith_min
30      print("---")
31
32    return ith_min
```

a. What do lines 20-25 do? (i.e. what is the value of **ith_min** when the program reaches line 26?)
Hint:
Trace through these lines assuming that **a** = [1, 2, 3, 4, 5], **max** = 5 and **min** = 1.
Trace through these lines again for the same **a** and **max**, but when **min** = 3.

b. Given array **a**=[3, 5, 1, 10, 7, 22, 15, 19, 6, 8, 16] and **k**=6, what is the value of **ith_min** that will be printed by **line 29** when the value of **i** printed by **line 28** is 4?

c. What is the worst-case complexity of the **find_minimum** algorithm?

d. Describe a more efficient algorithm to find the kth smallest number of an array of **n** distinct numbers. Show that your proposed algorithm will have a lower (better) worst-case complexity than the algorithm **find_minimum**.

13. a. Is $2^{n+1} = O(2^n)$? b. Is $2^{2n} = O(2^n)$? (adopted from CLRS Ex 3.1-4)

Practice Questions on Iteration & Decomposition (Week 3)

Tier-1 Questions

1. The following sub-questions concern the array below:

```
x = [8, 9, 11, 16, 32, 37, 43, 45, 52, 55, 58, 61, 63, 66, 68, 82, 95, 99]
```

   a) How many elements will be checked by linear search and by binary search respectively to find the number 8?
   b) How many elements will be checked by linear search and by binary search respectively to find the number 16?
   c) How many elements will be checked by linear search and by binary search respectively to find the number 50?

2. The linear search algorithm (shown below) assumes that each value occurs only once in the array being searched.

```
01  def search(a, k):
02      i = 0
03      while i < len(a):
04          if a[i] == k:
05              return i
06          i = i + 1
07      return -1
```

   Suppose that the array **a** may contain multiple occurrences of the same value.
   a) Modify the above algorithm such that it returns the number of occurrences of the key **k** being searched.
   b) Modify the above algorithm such that it returns the index of the second occurrence of the key **k** if it exists, and **None** otherwise.
   c) Modify the above algorithm such that it returns the index of the last occurrence of the key **k** if it exists, and **None** otherwise.

3. The following algorithm implements insertion sort in ascending order.

```
01  def isort(list):
02    if list.size <= 1:
03      return list
04
05    for i in range(1, len(list)):
06      value = list[i]
07      position = i
08
09      while (position > 0) and value < list[position-1]:
10        position = position-1
```

```
11
12      while (i > position):
13        list[i] = list[i-1]
14        i = i - 1
15      list[position] = value
16
17    return list
```

a) Modify the above algorithm to sort in descending order.
b) What is the best-case complexity of this modified algorithm, and when does this happen?
c) What is the worst-case complexity of this modified algorithm, and when does this happen?

4. When merge sort is applied to an array **x**, what is the content of the array **x** just before the final merge step?
   a) When `x = [96, 80, 88, 34, 47, 65, 52, 73]`
   b) When `x = [62, 77, 18, 48, 64, 45, 34, 88, 82, 87, 17]`
   c) When `x = [27, 92, 93, 91, 35, 22, 31]`

5. Let us denote **A** to be a sorted array containing **N** elements, e.g.
   A `= [6, 19, 30, 31, 33, 47, 57, 60].`
   Let us further denote $\mathbf{A_k}$ to be a right-ward shift by `k` steps. For instance, we have
   $\mathbf{A_1}$ `= [60, 6, 19, 30, 31, 33, 47, 57]` and
   $\mathbf{A_3}$ `= [47, 57, 60, 6, 19, 30, 31, 33].`

   Design an algorithm (pseudo-code) to find the maximum element in $\mathbf{A_k}$ in the following conditions:
   a) When the value of **k** is known, and the algorithm should have O(1) complexity.
   b) When the value of **k** is unknown, and the algorithm should have O(N) complexity.
      Hint: linear search
   c) When the value of **k** is unknown, and the algorithm should have O(log N) complexity.
      Hint: binary search

6. You are given as input an unsorted array **B**, containing **n** distinct integers (no duplicates) in the range from 1 to (n+1), whereby exactly one integer in this range is missing from the array. The objective is to determine which integer is missing. For example, for the array **B** = [2, 3, 5, 6, 1, 7], we have **n** = 6, and the range is from 1 to 7. The missing integer is 4.
   Provide an algorithm (in pseudo code or Python code) to find the missing integer in an input array **B**. The lower the complexity of your algorithm is, the higher your marks will be.

Tier-2 Questions

7. For each of the array below, which sorting algorithm: insertion sort or merge sort, will make fewer comparisons?

   a) When `x = [2, 11, 26, 39, 58, 66, 74, 85]`

b) When `x = [27, 76, 43, 97, 55, 41, 2, 91]`
c) When `x = [91, 84, 77, 65, 64, 39, 22, 18]`

8. When merge sort is applied to an array of `N` elements:

   a) What is the best-case complexity?
   b) What is the worst-case complexity?
   c) Will merge sort make a smaller, the same, or a greater number of comparisons when sorting an array that is already sorted, as opposed to a random unsorted array?

9. When merge sort is applied to an array **x**, what is the content of the array **x** just before the final merge step?

   a) When `x = [48, 82, 51, 67, 65, 25, 45, 2, 36]`
   b) When `x = [8, 12, 55, 89, 54, 22, 95, 25, 75, 59, 29, 18, 79]`
   c) When `x = [69, 15, 21, 95, 43, 80, 42, 79, 73, 33, 81, 11, 67, 52, 60]`

10. Binary search systematically decomposes the array being searched into two sub-arrays. Ternary search systematically decomposes the array into three sub-arrays. N-ary search decomposes the array into N sub-arrays. If we keep increasing number of sub-arrays from two to three, four, and five and so on, will the resulting N-ary search algorithm be increasingly more and more efficient? Explain why.

11. Suppose you are given a series of **N** numbers as input, where **N** is even. Your objective is to derive pairs of numbers, such that the maximum sum among all pairs will be minimized.

    For example, suppose the numbers are 88, 48, 78, and 73.
    - One pairing solution is (88, 48) and (78, 73) with sums 88 + 48 = 136 and 78 + 73 = 151 respectively. The maximum sum is 151.
    - Another pairing solution is (88, 78) and (48, 73) with sums to 166 and 121 respectively. The maximum sum is 166.
    - The last pairing solution is (88, 73) and (48, 78) with sums to 161 and 126 respectively. The maximum sum is 161.

    The best pairing solution is the one with the smallest maximum sum, i.e., the first pairing solution. Design an algorithm (pseudo-code) to find the best pairing solution, and derive the time complexity of your algorithm. The lower the complexity of your algorithm is, the higher your marks will be.

12. You are given **k** arrays [A_1, A_2, A_3…A_k], where each array A_i has **n** sorted integers. For example, if **k** = 4 and **n** = 3, you have 4 sorted arrays of 3 integers each. Here is a possible set of arrays:
    A_1 = [1, 3, 5], A_2 = [4, 9, 10], A_3 = [2, 6, 7] and A_4 = [8, 12, 15]
    The objective is to combine these **k** input arrays into a single sorted array. The expected output is the combined sorted array, i.e. [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 15].

a) Provide an algorithm (in pseudo code or Python code) to perform this task.
b) What is the Big O time complexity of your proposed algorithm in terms of **k** and **n**?

Tier-3 Questions

Below are some coding challenges available as HackerRank links. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.

13. (**Missing numbers**) Given two arrays of integers, find which elements in the second array are missing from the first array.
Link: https://www.hackerrank.com/challenges/missing-numbers/problem?isFullScreen=true

14. (**Pairs)** Given an array of integers and a target value, determine the number of pairs of array elements that have a difference equal to the target value.
Link: https://www.hackerrank.com/challenges/pairs/problem?isFullScreen=true

15. (**Insertion Sort)** This challenge will cover part of Insertion Sort, a simple and intuitive sorting algorithm.
Link: https://www.hackerrank.com/challenges/insertionsort1/problem

~End

Practice Questions on Recursion (Week 4)

Tier-1 Questions

1. **Pascal's triangle** is made up of multiple levels of integers as shown below.

Level	List of Numbers
0	1
1	1 1
2	1 2 1
3	1 3 3 1
4	1 4 6 4 1

At any given level **n**, there are **n**+1 numbers. Let **pt(n, k)** represent the **k**th number at level **n** of the Pascal's triangle (range of **k** is from 1 to **n**+1).
The value **pt**(**n**, **k**) can be computed recursively as follows:

- For **k** = 1 or **n**+1, coefficient is 1;
- For any other value of **k**, its value is the sum of two numbers from the immediate previous level – the number to the left and the number to the right. Written formally:

  pt(**n**, **k**) = **pt**(**n**-1, **k**-1) + **pt**(**n**-1, **k**).

In the example above, the number 4 at level 4 is the sum of 1 and 3 from level 3, i.e.:
pt (4, 2) = **pt** (3, 1) + **pt** (3, 2)

a) Based on the above definition, complete the following recursive function design.

```
# Compute kth number at level n, n ≥ 0, k ≥ 1
def pt(n, k):
  # base case 1
  if _____________:
      return 1
  # base case 2
  if _____________:
      return 1
  # reduction step
```

b) Trace the sequence of recursive function calls for **pt**(4, 2). Hint: recall how Merge Sort's calling sequence was traced.

2. You are given the following algorithm to determine the value of **x** raised to the power of **n**.

```
01  def power3(x, n):
02      if n == 0:
03          return 1
04      elif n % 2 == 0:
05          temp = power3(x,
06  n//2)
07          return temp * temp
08      else:
09          temp = power3(x,
    n//2)
            return x * temp *
    temp
```

a) How many multiplications does **power3** perform for **x** = 3 and **n** = 16?
b) How many multiplications does **power3** perform for **x** = 3 and **n** = 19?
c) What is the complexity of **power3**?

3. Suppose that **sub_function** has linear complexity. What is the complexity of **my_function** below?

a)
```
def my_function(n):
    if (n > 0):
        sub_function(n)
        my_function(n//2)
```

Hint:

$$\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16}+\cdots=\sum_{n=1}^{\infty}\left(\frac{1}{2}\right)^n=\frac{\frac{1}{2}}{1-\frac{1}{2}}=1.$$

b)
```
def my_function(k, n):
    if (n > 0):
        sub_function(k)
        my_function(k, n//2)
```

c)
```
def my_function(n):
    if (n > 0):
        sub_function(n)
        my_function(n-1)
```

4. Write a recursive algorithm **max_array(a)** that takes in an array of positive integers and returns the biggest integer in the array. Your solution should not rely on sorting the array first.

5. A palindrome is a word that has the same spelling forwards and backwards, like “MADAM”. Write a recursive algorithm **is_palindrome(s)** to check if a string is a palindrome. For example, **is_palindrome("madam")** returns **True**, but **is_palindrome("madman")** returns **False**.

6. Given a recursive algorithm **f**(**n**) that takes a non-negative integer **n** as input.

```
def f(n):
    if n == 0 or n == 1:
        return 1
    return -f(n-1) - f(n-2)
```

a) Specify the output values of the following expressions: **f**(1), **f**(5), **f**(6), **f**(330).

b) What is the worst-case complexity of the algorithm **f**(n)? Show your working.

Tier-2 Questions

7. Rewrite the following Dijkstra's algorithm to calculate the greatest common divisor of two integers using recursion. Hint: base case will be when **a==b**

```
01  def dijkstra(a, b):
02      while a != b:
03          if a > b:
04              a = a - b
05          else:
06              b = b - a
07      return a
```

8. Rewrite the following Euclid's algorithm to calculate the greatest common divisor of two integers using recursion.

```
01  def euclid(a, b):
02      while b != 0:
03          t = b
04          b = a % b
05          a = t
06      return a
```

or

```
01  def euclid(a, b):
02      while b != 0:
03          a, b = b, a%b  # swap
04      return a
```

9. Write a recursive algorithm **repeat_string(s,n)** that returns a concatenation of n copies of the string **s**. For example, **repeat_string("apple", 3)** will return **"appleappleapple"**.

10. Write a recursive algorithm **sum**(**n**) that computes the sum of the first n positive integers. For example, **sum**(1) returns **1**, **sum**(2) returns **1+2**, **sum**(3) returns **1+2+3**.

11. Write a recursive algorithm **reverse(a)** that returns an array with the same elements as **a**, but in reverse order.

12. The function **f12**(**x**, **n**) is defined like this:

$$f(x,n) = \frac{1}{x^1} + \frac{1}{x^2} + \frac{1}{x^3} + \cdots + \frac{1}{x^n}$$

e.g.:

$$f(2,4) = \frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} = 0.9375$$

You are given this power function that returns $\mathbf{x^n}$. Use this in your solution:

```
def power(x, n):
    if n == 0:
        return 1
    if n == 1:
        return x
    return x * power(x, n-1)
```

Write the function **f12_rec**(**x**, **n**) that uses recursion to return the correct value.

13. The function **f13**(**x**, **y**, **n**) is defined like this:

$$f(x, y, n) = 1x + 2y + 3x + 4y + 5x \ldots (there\ are\ n\ terms\ in\ the\ series)$$

e.g.:

$$f(4, 3, 5) = 1(4) + 2(3) + 3(4) + 4(3) + 5(4) = 54$$

Write the function **f13_rec**(**x**, **y**, **n**) that uses recursion to return the correct value.

14. The function e^x is approximated by the following infinite series[1]: $e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$

The function **f14** takes two arguments **x** and **n** (where **n**>0) and returns the value of the series after **n** iterations, so that: $e^x \sim 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$

e.g.: $f(2,4) \sim 1 + \frac{2}{1!} + \frac{2^2}{2!} + \frac{2^3}{3!} + \frac{2^4}{4!} = 7.0$

```
01  def f14(x, n):
02      sum = 1
03      for j in range(1, n+1):
04          sum += (power(x, j) /
05  factorial(j))
06      return sum
07
08  def power(x, n):
09      if n == 0:
10          return 1
11      if n == 1:
12          return x
13      return x * power(x, n-1)
14
15  def factorial(n):
16      if n == 1:
17          return 1
        return n * factorial(n-1)
```

a) What is the complexity of the iterative version of **f14** given above?

b) Rewrite **f14** using recursion. Call your function **f14_rec**. You will be given more marks if your algorithm's time complexity is lower (better).

[1] For the Mathematically-inclined, see https://www.efunda.com/math/taylor_series/exponential.cfm

Tier-3 Questions

Below are some coding challenges available as HackerRank links. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.

15. (**Recursive Digit Sum**) This challenge involves writing a recursive algorithm to find the super digit of an integer number A.

Link: https://www.hackerrank.com/challenges/recursive-digit-sum/problem?isFullScreen=true

16. (**Password Cracker**) This challenge involves verifying whether a given string can be constructed by concatenating one or more passwords from a set of provided passwords in any order, with repetitions allowed. The task is to determine if the string is valid according to the site's password verification system.

Link: https://www.hackerrank.com/challenges/password-cracker/problem?isFullScreen=true

17. (**Power Sum**) This challenge involves finding the number of ways an integer **X** can be expressed as the sum of the n-th powers of unique natural numbers. The task is to implement the *powerSum* function, which returns the total number of such combinations.

Link: https://www.hackerrank.com/challenges/the-power-sum/problem?isFullScreen=true

~End

Practice Questions on Linear Data Structures (Week 5)

Tier-1 Questions

15. Suppose that you are given the following sequence of letters.

A R D G * O * C A R E L D O * * C F U L * * C L O C * K * * *

a. If a letter means push and an asterisk means pop, give the sequence of letters returned by the pop operations when this sequence of operations is performed on an initially empty **stack**.
b. If a letter means enqueue and an asterisk means dequeue, give the sequence of letters returned by the dequeue operations when this sequence of operations are performed on an initially empty **queue**.

16. Draw the queue **q** returned by the function below if the input is **n** = 6. Clearly indicate the head of the queue in your drawing.

```
def do_weird_stuff(n):
    q = Queue()
    for i in range(0, n):
```

```
        if i%2 == 1:
            x = q.dequeue()
            q.enqueue(x+i)
        else:
            q.enqueue(i)
    return q
```

17. Suppose that you can only push numbers from 1 to n onto a stack in increasing order (i.e. you can only push 3, after you have pushed 2. You can only push 2 only after you have pushed 1). At any point in time, you can call the pop operation and print the popped value.

    For example, the following operations:

```
s = Stack()
s.push(1)
print(s.pop())
s.push(2)
s.push(3)
print(s.pop())
s.push(4)
print(s.pop())
print(s.pop())
s.push(5)
print(s.pop())
```

    will print out the sequence 1 3 4 2 5.

    a. Give a list of operations (like above) that would result in the following sequence: 1 3 5 4 2
    b. Is there any sequence that cannot occur? If so, give an example and explain why.

18. Suppose you are given a stack **s** and a queue **q**.

    a. If initially **s** is empty and **q** is not empty, describe how to invert (reverse the ordering) the queue using the stack.
    e.g. **q** initially contains [(head)"a", "b", "c"(tail)]. After inverting **q**, **q** will contain: [(head)"c", "b", "a"(tail)]

    b. If initially **s** is not empty and **q** is empty, describe how to invert the stack using the queue.
    e.g. **s** initially contains [(top)"a", "b", "c"]. After inverting **s**, **s** will contain: [(top)"c", "b", "a"]

    c. If initially **s** is not empty and **q** is empty, describe how to delete an element in the stack using the queue. Other than the item being removed, the remaining content of the stack should remain in the same order after the removal.
    e.g. **s** initially contains [(top)"a", "b", "c", "d", "e"]. After deleting "c" from **s**, **s** will contain: [(top)"a", "b", "d", "e"]

19. The following is a recursive implementation of the factorial function. Convert it into a non-recursive implementation using a stack.

```
def factorial(n):
    if n == 1:
        return 1
    return n * factorial(n-1)
```

Tier-2 Questions

20. Draw the queue returned by the function below if the input **n** is 6. Clearly indicate the head of the queue in your drawing.

```
def do_strange_stuff(n):
    q = Queue()
    q.enqueue(0)
    for i in range(0, n):
        if i%2 == 0:
            x = q.dequeue()
            q.enqueue(i-x)
        else:
            q.enqueue(i)
    return q
```

21. Consider the following function:

```
01  def mystery(x):
02    s = Stack()
03    out = 0
04
05    while x > 0:
06      s.push(x % 2)
07      x = x // 2
08
09    while s.count() > 0:
10      out = out*10 + s.pop()
11    return out
```

a) What will be returned by **mystery**(3)?
b) What will be returned by **mystery**(5)?
c) What does the **mystery** function do?

If you are able to answer (c),

d) **mystery**() above pushes **x%2** to the stack. Rewrite **mystery** so that **x//2** is pushed to the stack instead. What the function does remains the same.

22. Suppose that you can only enqueue numbers from 1 to n onto a queue in increasing order. At any point in time, you can call the dequeue operation and print the dequeued value.

    a) Give a list of queue operations that will produce the sequence: 1 2 3 4 5
    b) Is there any sequence that cannot occur? If so, give an example, and explain why.

23. You were given an example of how to write **is_palindrome(word)** using a queue and a stack in your slides. Now, design alternative implementations of **is_palindrome** using:

    a) two queues only
    b) one stack only

24. The following is a recursive implementation of the Dijkstra's algorithm to find the greatest common divisor of two numbers **a** and **b**. Convert it into a non-recursive implementation using stack.

```
def dijkstra(a, b):
  if a == b:
    return a
  if a > b:
    return dijkstra(a - b, b)
  else:
    return dijkstra(a, b - a)
```

Tier-3 Questions

Below are some coding challenges available as HackerRank links. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.

11. (**Queue using two stacks**): This challenge involves implementing a queue using *two stacks*.
Link: https://www.hackerrank.com/challenges/queue-using-two-stacks/problem?isFullScreen=true 12. (**Balanced brackets**): Given a string of brackets, determine whether the sequence of brackets is balanced.

Link: https://www.hackerrank.com/challenges/balanced-brackets/problem?isFullScreen=true

13. (**Largest rectangle**): Find an integer representing the largest rectangle that can be formed within the bounds of consecutive buildings.

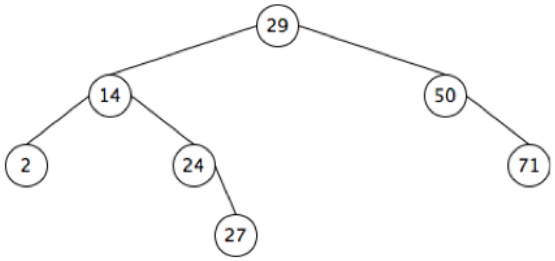
Link: https://www.hackerrank.com/challenges/largest-rectangle/problem?isFullScreen=true

~End

Practice Questions on Binary Trees (Week 7)

Tier-1 Questions

25. This question concerns the following binary tree:



a) Which is the root node?
b) Which are the leaf nodes?
c) Which nodes are the parents of 14?
d) Which nodes are the children of 14?
e) Which nodes are the descendants of 14?
f) Which nodes are the ancestors of 14?
g) Which nodes are the siblings of 14?

26. Build a binary search tree (BST) based on alphabetical ordering for the words **Le**xus, **Me**rcedes, **B**MW, **A**udi, **La**mborghini, **C**hevrolet, **P**orsche, **Ma**serati, **L**otus.

a. Draw the BST obtained by inserting the words in the order they are listed above. How many comparisons are needed to locate the following words respectively:
   i. **A**udi,
   ii. **La**mborghini,
   iii. **Ma**serati,
   iv. **P**orsche?

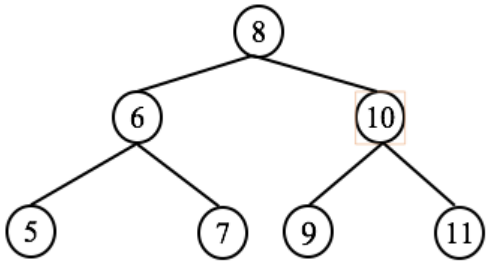
b. Draw the BST obtained by inserting the words in alphabetical order (i.e. we will first insert **A**udi, followed by **B**MW, **C**hevrolet, and so on in alphabetical order). How many comparisons are needed to locate the following words respectively:
   i. **A**udi,
   ii. **La**mborghini,
   iii. **Ma**serati,
   iv. **P**orsche?

27. Andy is organizing a badminton tournament. Each game is played between two participants. The winner goes to the next round. The loser is eliminated. There are no ties. If there are an odd number of participants in a particular round, one unpaired participant will advance directly to the next round.

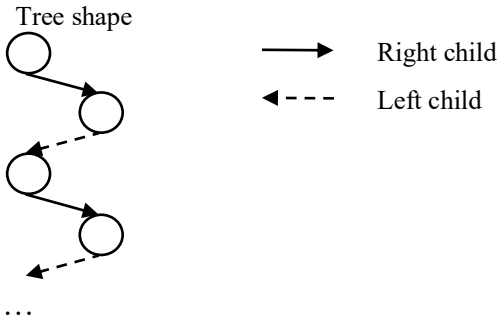
a. Use a binary tree to determine how many games need to be played to determine the champion if there are 14 participants.
b. We can view the tournament as an algorithm, which takes as input the number of participants **n**, and returns as output the champion. If each game is a step in this algorithm, what is the complexity of the algorithm?

28. We would like to design a sorting algorithm called TreeSort. It works by inserting each data element into a BST. To print the data elements in a sorted order, we traverse the tree in a certain order.
    a. Which traversal method do we use for this purpose?
    b. What is the complexity of the above algorithm?

29. Given an array of distinct numbers, we seek to determine whether the input array could conceivably be the postorder traversal of some BST containing elements in that array. For example, if the input is `a = [5, 7, 6, 9, 11, 10, 8]`, there exists a BST (shown below) for which `a` is the postorder traversal. For another example, if the input is `b = [3, 1, 2]`, there is no BST containing these integers could possibly have `b` as its postorder traversal.
    a. Design an algorithm (in pseudocode or Python code) that will return **True** if the input array could conceivably be the postorder traversal of some BST, and return **False** otherwise.
    b. What is the worst-case complexity of your algorithm?



30. Given an array **A** = [2, 10, 4, 7, 19, 5, 8, 11, 3, 15]
    a) The binary search tree (BST) of an array is constructed by inserting elements in the array according to their index position from left to right. Draw the BST of the given array **A** above.
    b) For the BST built above, produce the sequence of nodes visited by preorder and postorder traversals respectively.
    c) The sequence of numbers in an array affects the shape of the constructed BST. One of the shapes resulting in the highest number of comparisons in the worst case when searching is below:



Reorder the elements in the given array **A** to produce another array **B**, such that the BST of **B** would have the above shape.

Tier-2 Questions

31. Suppose we run the following operations (do not actually run it by code, instead do them by hand).

```
>> t = BinaryTree()
>> t.setRoot(Node("A"))
>> t.root.setLeft(Node("B"))
>> t.root.setRight(Node("C"))
>> t.root.left.setLeft(Node("D"))
>> t.root.right.setRight(Node("E"))
>> t.root.right.right.setLeft(Node("F"))
>> t.root.left.setRight(Node("G"))
```

    a) What is the output of calling `t.root.right`?
    b) What is the output of calling `t.root.left.right`?
    c) What is the height of the tree?
    d) What is the parent of "E"?
    e) What is the order of nodes visited for preorder traversal?
    f) What is the order of nodes visited for inorder traversal?
    g) What is the order of nodes visited for postorder traversal?

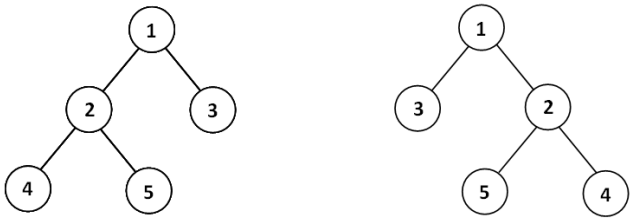
32. If a BST with **n** nodes is not full,
    a. What is the worst case complexity of **searchbst** in terms of **n**?
    b. What is the shape of the tree for this worst case?

33. Bob has a contact list of his colleagues. He finds it tiresome to have to flip through his notebook to find the names. Bob has not taken the Computational Thinking course yet, and is seeking your advice on how to organize his contact list into a BST (based on alphabetical ordering).

    a. What is the best way, i.e., in what sequence, should Bob insert the names of his colleagues into a BST? Explain why.
    b. What is the worst way? Explain why.

34. Andy starts a chain email. He sends an email to two of his friends, with an instruction to either forward it to two other friends (who have never received it), or not to forward it at all. The chain of email forwarding can be represented by a binary tree. If there are 100 other people who end up sending emails:
    a) What is the maximum possible height of this binary tree?
    b) What is the minimum possible height of this binary tree?

35. Consider the problem of getting the mirror of a given binary tree. The mirror of a binary tree is another binary tree with the left and right children of all non-leaf nodes interchanged. For example, the following figure shows two binary trees that are mirrors of each other.
    a. Design an algorithm to derive the mirror of a binary tree using recursion.
    b. What is the Big O complexity of this recursive function?



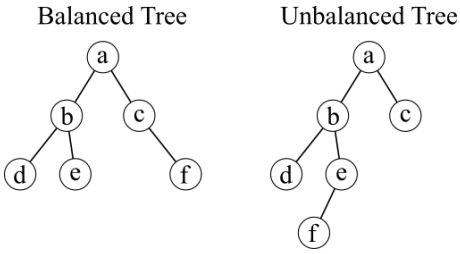
Mirror Binary Trees

36. The following algorithm determines the height of a tree rooted at the input root.

```
01: def depth(root):
02:   if root == None:
03:     return 0
04:   else:
05:     return 1 + max(depth(root.left), depth(root.right))
```

a) What is the worst-case complexity of the **depth** algorithm for a binary tree of **N** nodes?
b) Write a recursive algorithm **is_balanced**, which takes in as input the root of a binary tree, and returns **True** if that binary tree is balanced and **False** otherwise. You may assume that the algorithm **depth** above is given, and you are allowed to use it within your proposed recursive algorithm **is_balanced**. State the worst-case complexity of the algorithm **is_balanced** for a binary tree of **N** nodes and clearly show your working.

A binary tree is balanced if for every node, the depth of the node's left subtree is different from the depth of the node's right subtree by at most one. Refer to the trees at the right: the tree on the left is balanced. The tree on the right is not balanced because for node a, its left subtree has a height of depth(b) = 3 and its right subtree has a height of depth(c) = 1.



Here is another example to explain how to determine if a tree is balanced or not:

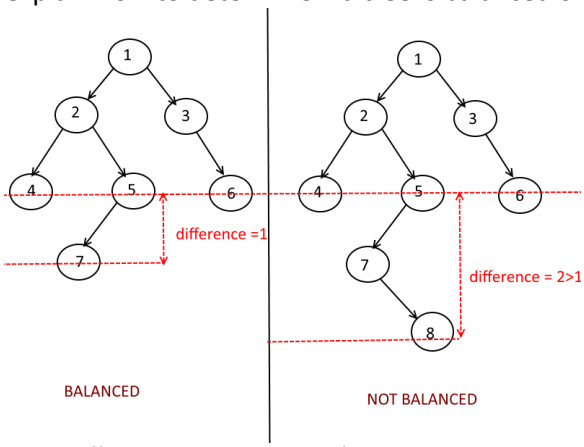


Diagram taken from https://algorithms.tutorialhorizon.com/find-whether-if-a-given-binary-tree-is-balanced

Tier-3 Questions

Below are some coding challenges available as HackerRank links. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.

13. (**Height of tree**): The height of a binary tree is defined as the number of edges between the tree's root and its furthest leaf. Complete a function to return the height of a given binary tree
    Link: https://www.hackerrank.com/challenges/tree-height-of-a-binary-tree/problem?isFullScreen=true

14. (**Tree insertion**): You are given a pointer to the root of a binary search tree and values to be inserted into the tree. Insert the values into their appropriate position in the binary search tree and return the root of the updated binary tree.
    Link: https://www.hackerrank.com/challenges/binary-search-tree-insertion/problem?isFullScreen=true

15. (**Binary search tree verification**): Given a binary tree, return a *boolean* denoting whether or not the binary tree is a binary search tree.
    Link: https://www.hackerrank.com/challenges/is-binary-search-tree/problem?isFullScreen=true

~End

Practice Questions on Graph (Week 9)

Tier-1 Questions

37. Scout Airways flies from Singapore to all the capitals in ASEAN countries. However, there is a return flight to Singapore only if the capital's population is larger than Singapore's. Otherwise, to return to Singapore, one needs to first fly to another capital with a return flight to Singapore. There is always a flight from any capital city to another capital city (other than Singapore) with the next largest population size (e.g., there is a flight from Bandar Seri Begawan (pop: 0.1M) to Naypyidaw (pop: 0.9M)). The following table lists the population sizes of capitals in South East Asia.

Country	*Capital*	*Population*
Singapore	Singapore	5.2M
Indonesia	Jakarta	10.2M
Malaysia	Kuala Lumpur	1.6M
Thailand	Bangkok	8.3M
Philippines	Manila	1.7M
Brunei	Bandar Seri Begawan	0.1M
Vietnam	Hanoi	6.6M
Cambodia	Phnom Penh	1.5M
Laos	Vientiane	6.5M
Myanmar	Naypyidaw	0.9M

    Draw a directed graph, where the vertices are the capital cities, and an edge exists from one city to another city only if there is a flight from the first to the second city.

38. A graph has 10 vertices, *A* to *J*.
    - *A* connects to *B* and *I*.
    - *B* connects to *D* and *H*.
    - *C* connects to *D*.
    - *D* connects to *H*.
    - *E* connects to *F* and *G*.
    - *F* connects to *A*.
    - *G* connects to *J*.
    - *H* connects to *E*.
    - *I* connects to *B* and *J*.
    - *J* connects to *A*.

    a. Represent this graph in adjacency list and adjacency matrix respectively.
    b. Which vertices have the highest in-degree?
    c. Which vertices have the highest out-degree?
    d. What is the sequence of vertices visited by breadth-first search beginning from *A*?
    e. What is the sequence of vertices visited by depth-first search beginning from *A*?

39. Here are the names of 8 courses in a particular degree program, and their pre-requisites:
    - IS200, PMSB and DM have no pre-requisites
    - To take OOAD, the student must have passed IS200
    - To take SE, the student must have passed OOAD
    - To take EI, student must have passed OOAD, DM and PMSB
    - To take AA, the student must have passed IS200 and EI
    - To take EWS, the student must have passed SE and EI

    a. Draw a directed graph to represent these courses and their pre-requisites
    b. Represent the graph you have drawn in the form of an adjacency matrix:

From \ To	IS200	DM	OOAD	SE	EI	AA	EWS	PMSB
IS200								
DM								
OOAD								
SE								
EI								
AA								
EWS								
PMSB								

    c. What will be the order in which the courses are visited if a Depth First Search is performed starting from IS200 given that the order of the vertices in the graph goes like this: IS200, DM, OOAD, SE, EI, AA, EWS, PMSB (as shown in the table above).
    d. Assuming that a student can only take 1 course per term, suggest ONE valid topological order in which all these courses can be taken over 8 terms.

40. G is a graph with the following adjacency list:

```
A | B, E, F
```

```
B | C, D, F
C | D
D | F
E | D, F
F | C
```

a. Which node is visited right after visiting node F if we perform depth-first-search on G, starting from A, based on the adjacency list above?
b. Invert the direction of one edge in G so that we get a DAG. Find a topological ordering in the resulting DAG.

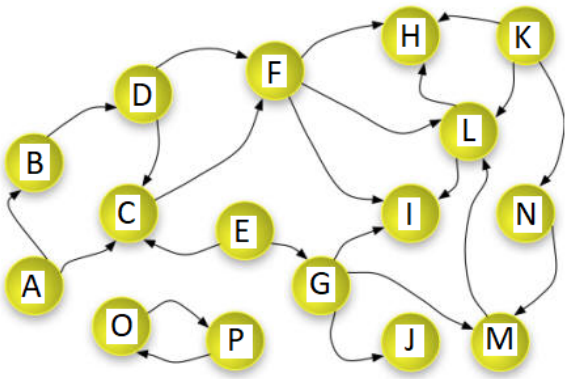
41. A graph has six vertices, labeled A to F. There are exactly two paths from A to F. The first path is (A, B, C, F). The second path is (A, D, E, F). All the edges in this graph can be found in either of this path.

    a. If you run topological sorting algorithm, what is the output topological ordering?
    b. Could there be more than one possible outputs? What affects which output you get?

Tier-2 Questions

42. You are attending a friend's birthday party. Suppose that you want to track who knows whom in the birthday party, what kind of graph: directed or undirected, will you use? Can it have cycles?

43. Refer to the graph below:



(Diagram adapted from www.openarchives.org/ore/0.2/datamodel-images/WebGraphBase.jpg)

a) Represent this graph in adjacency list and adjacency matrix respectively.
b) Which vertices have the highest in-degree?
c) Which vertices have the highest out-degree?
d) What is the sequence of vertices visited by **BFS** beginning from *A*?
e) What is the sequence of vertices visited by **DFS** beginning from *A*?

44. G is a weighted graph with the following adjacency list.

A	(B, 2)	(E, 1)	(F, 5)
B	(A, 2)	(D, 3)	(F, 3)
C	(D, 4)		
D	(B, 3)	(C, 4)	(F, 3)

E	(A, 1)
F	(A, 5) (B, 3) (D, 3)

A simple cycle is a cycle that goes through each of its nodes exactly once. Add one edge in to G so that the resulting graph contains at least one simple cycle going through all the nodes.

45. You are considering taking a certificate program in marketing. There are three levels of courses.
    - The first-level courses are 1A, 1B, and 1C. There are no prerequisites for first-level courses.
    - The second-level courses are 2A and 2B. The prerequisites for 2A are 1A and 1B. The prerequisites for 2B are 1B and 1C.
    - The third-level courses are 3A, 3B, and 3C. 2A is the prerequisite for all third-level courses. In addition, to take 3B, you first need to take 2B. To take 3C, you first need to take 1C.

    You need to complete the program in eight terms, taking one course per term. You have to propose the sequence of courses that you will follow over the eight terms. Figure out one valid sequence of courses.

46. A graph has **n** vertices, with labels 1, 2, ..., **n**. There is an edge from every vertex with label **i** to another vertex with label **j** if **i** < **j**.
    a) How many edges exist in this graph?
    b) Is there any cycle in this graph?
    c) How many topological orderings exist in this graph?

Tier-3 Questions

*Below are some coding challenges available as **Leetcode** links. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.*

11. (**Course Schedule**): You need to complete certain courses in a specific order. You are given an array that specifies prerequisite relationships, meaning you must take one course before enrolling in another.

    Link: https://leetcode.com/problems/course-schedule/description/

12. (**Number of Islands**): Given an *m*n* 2D binary grid representing a map where '1' denotes land and '0' represents water, return the total number of islands. *An island consists of connected land cells ('1's) that are adjacent either horizontally or vertically and is surrounded by water. You can assume that the grid's boundaries are entirely water.*

    Link: https://leetcode.com/problems/number-of-islands/description/

13. (**Find If a Path Exists in a Graph**): Given an undirected graph with *n* vertices labeled from 0 to n-1, the edges are represented as a list where each pair $[u_i, v_i]$ indicates a connection between two vertices. There are no self-loops or duplicate edges. Determine if there is a path from the `source` vertex to the `destination` vertex. Return `true` if a path exists, otherwise return `false`.

    Link: https://leetcode.com/problems/find-if-path-exists-in-graph/description/?envType=problem-list-v2&envId=graph

~End

Practice Questions on Greedy Heuristics (Week 11)

Tier-1 Questions

47. Suppose we are solving the Traveling Salesman Problem (TSP) for 20 cities. How many possible routes need to be examined if:

    a. the distance between two cities are symmetric, e.g., A to B is the same as B to A
    b. the distances between two cities are different in different directions, e.g., A to B is shorter or longer than B to A.

48. The following table shows the distances between 7 cities. Work out a proposed route and the distance of the proposed round trip.

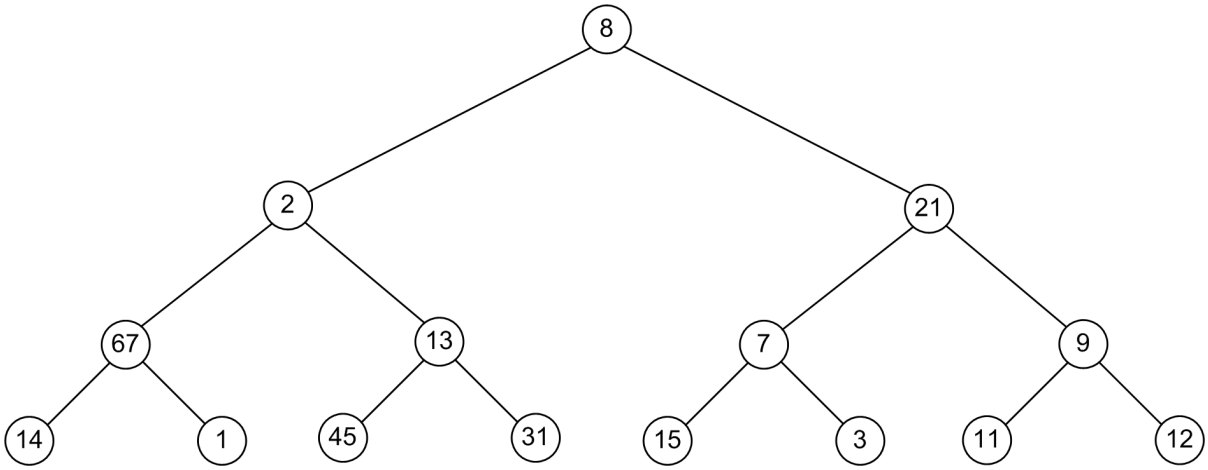
    a) Using the first greedy algorithm in the slide with A with the starting city.
    b) Using the second greedy algorithm in the slide starting with the closest pair of cities.

	A	B	C	D	E	F	G
A	-	205	86	76	259	278	144
B	205	-	246	250	112	81	191
C	86	246	-	17	325	326	230
D	76	250	17	-	323	329	220
E	259	112	325	323	-	79	173
F	278	81	326	329	79	-	230
G	144	191	230	220	173	230	-

49. Following is a greedy algorithm for finding the path from the root to a leaf in a binary tree with the highest sum of values of its nodes.

    "Starting from the root, go to the child of the current node with the higher value and record the edge; repeat until we reach a leaf"

    a. Perform the above algorithm on the tree below.
    b. Provide another example tree to show that the above algorithm may not find the path with the highest sum of values for that tree.



50. You work as a cashier in a convenience store. Suppose the coin denominations are 20 cents, 10 cents, 5 cents, and 1 cent. Your job is to give change to customers, in minimal number of coins. For example, if the change is 35 cents, the minimal number is 3 coins, consisting of 20-cent, 10-cent, and 5-cent coins.

    a) Design a brute force algorithm to work out the change in as few coins as possible.
    b) Design a greedy algorithm to work out the change.
    c) For the above coin denominations, is there any change amount for which greedy fails to find the optimal solution?

51. You have a bag that can carry no more than 15 kg. You have the following items to put into the bag:
    - item A has weight of 12 kg and value of $10
    - item B has weight of 5 kg and value of $8
    - item C has weight of 4 kg and value of $2
    - item D has weight of 3 kg and value of $5
    - item E has weight of 2 kg and value of $6

    You want to optimize the sum of values of items to carry in the bag, without going over the 15 kg limit.

    a. Design a brute force algorithm to determine which items to be put into the bag.
    b. Design a greedy algorithm to determine the items.
    c. For the above five items, is there any weight limit for which greedy fails to find the optimal solution?

Tier-2 Questions

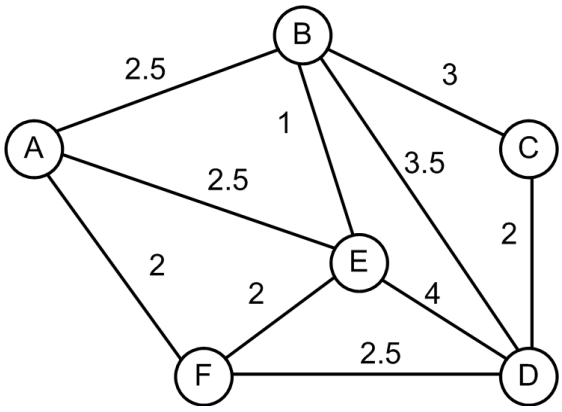
52. What would be the answer to the Q1a and Q1b in the Tutorial Questions above, if the tour must cover all cities but there is no need to return to the starting point?

53. What are the complexities of both the greedy algorithms (greedy 1 and greedy 2) for the Travelling Salesman Problem covered in your lecture slides?

54. Following is a greedy algorithm for finding the shortest path between the pair of nodes (**s**, **t**) in a graph.

    "Starting from **s**, go to the node closest to the current node that has not yet been visited and record the edge; repeat until we reach **t**".

    a) Perform the above algorithm on the following graph with the pair of nodes (A, D). (i.e. starting from A, ending at D.)
    b) Provide another example pair of nodes (other than A, D) to show that the above algorithm may not find the shortest path for this pair.



55. Suppose that in the Kingdom of Shrek, the coin denominations are 1-dollar, 60-cent, 35-cent, 5-cent, and 1-cent. You are a shopkeeper. Each time a customer pays for purchase, your objective is to return the change (if any) with as few coins as possible.

    You adopt the greedy strategy to always return the largest coin denomination possible. For example, if the change is $2.65, then:

    - first you return 1-dollar coin, remainder: $1.65
    - then you return 1-dollar coin, remainder: $0.65
    - then you return 60-cent coin, remainder: $0.05
    - then you return 5-cent coin, remainder: $0.

    Therefore, the solution is 4 coins, which happens to be optimal.

    a) Provide another change amount that the Greedy Strategy will be successful to discover the optimal solution.
    b) Provide a change amount whereby the Greedy Strategy will fail to discover the optimal solution. In other words, there exists another solution that would return fewer coins than the greedy solution.

56. You are the manager of the Singapore ski team for the Winter Olympics, and your team consists of 5 members of heights h1, h2, h3… h5, and you are provided with 5 skis of length l1, l2, l3… l5. There are hence 5! (or 120) unique combinations of skiers to skis. To be able to ski as quickly as possible, a skier should get skis the length of which matches his height. Hence, to ensure an Olympic medal, you need to assign skis to each skier to minimize the sum of the absolute differences between the height of the skier and the length of his ski.

    Clearly describe a greedy algorithm to get a reasonably good ski/skier assignment. What is the worst-case complexity of your algorithm?

Tier-3 Questions

Below are some coding challenges. These are optional, but if you can provide evidence of your attempts and achieve good scores, you will earn some class participation points.

11. (**Buying Toys**): Given a list of toy prices and an amount to spend, determine the maximum number of gifts to buy
    Link: https://www.hackerrank.com/challenges/mark-and-toys/problem?isFullScreen=true

12. (**Triangle with Maximum Possible Perimeter**): Given an array of stick lengths, use 3 of them to construct a non-degenerate triangle with the maximum possible perimeter.
    Link: https://www.hackerrank.com/challenges/maximum-perimeter-triangle/problem?isFullScreen=true

13. (**Power Plants):** Determine the fewest number of power plants needed to provide electricity to the entire list of cities. that number. If it cannot be done, return -1.

    Link: https://www.hackerrank.com/challenges/pylons/problem?isFullScreen=true

~End