COS10003 COMPUTER-LOGIC ESSENTIALS

ASSIGNMENT 3 ALGORITHM, GRAPHS AND COUNTING

Lecturer: Mr. Anh Ngoc Pham

Hai Hoang Le | s103542974

Counting

1.

a) i. The activities pattern is based on how many activities had to be finished in a day. We have 5 activities to finish as well as the order of the activities is concerned. So in this case, I will use the permutation:

1 activity done = P(5,1) = $\frac{5!}{(5-1)!} = 5$ ways

2 activity done = P(5,2) = $\frac{5!}{(5-2)!} = 20$ ways

3 activity done = P(5,3) = $\frac{5!}{(5-3)!} = 60$ ways

4 activity done = P(5,4) = $\frac{5!}{(5-4)!} =$120 ways

5 activity done = P(5,5) = $\frac{5!}{(5-5)!} = 120$ ways

The total different activities can be formed from these 5 activities =

= 5 + 20 + 60 + 120 +120

= 325 ways

There are total 325 ways that can be formed from 5 activities.

ii. From 5 activities pick 3 randomly activities and rearrange those. The number patterns can be formed from those 3 activities = P(5,3)

P(5,3) = = $\frac{5!}{(5-3)!} = 60$ ways

There are 60 ways that can be formed from 3 different activities.

iii. Let consider playing game is the first activities and unchanged, so we have 3 or 4 activities left because we have at least total of 4 activities.

The number of patterns = P(4,3) + P(4,4)

$$= \frac{4!}{(4-3)!} + \frac{4!}{(4-4)!}$$

= 24 + 24 = 48 ways

There are 48 ways can be formed in the case we have at least 4 activities and always starting with playing game.

b)

i. Because all 6 ingredients need to be used up, so there is a possibility that at least 1 ingredient used, some ingredients used, and all ingredients used. The order when adding ingredient will not affect the order the result so I will use the combination formula.

The total ways that those 6 ingredients can be combined =

= C(6,1) + C(6,2) + C(6,3) + C(6,4) + C(6,5) + C(6,6)

$$= \frac{6!}{(6-1)!*1!} + \frac{6!}{(6-2)!*2!} + \frac{6!}{(6-3)!*3!} + \frac{6!}{(6-4)!*4!} + \frac{6!}{(6-5)!*5!} + \frac{6!}{(6-6)!*6!}$$

= 6 + 15 + 20 + 15 + 6 + 1

= 63 ways

There are 63 ways that can combine when we have 6 ingredients.

ii. When select 3 from 6 ingredients, the order in not important. The number ways that can be formed =

= C(6,3)

$$= \frac{6!}{(6-3)!*3!}$$

= 20 ways

There are 20 ways that can combine 3 of 6 ingredients.

c)

i. We have 4 different locations around the house, 3 mini-figure per location

First location, choose 3 out of 12 mini-figure => C(12,3)

Second location, choose 3 out of the remaining 9 mini-figure => C(9,3)

Third location, choose 3 out of remaining 6 mini-figure => C(6,3)

Forth location, choose 3 out of remaining 3 mini-figure => C(3,3)

I assume that four locations can swap with each other so there will be P(4,4) ways to arrange the location

Number of different ways to arranged the mini-figures

= C(12,3) x C(9,3) x C(6,3) x C(3,3) x P(4,4)

= 220 x 84 x 20 x 1 x 24

= 8,870,400 ways

There are 8,870,400 ways that can arrange the mini-figure.

ii. Second approach:

Number ways that can arranged = $\frac{12!}{3! * 3! * 3! * 3!} * 4!$ = 8,870,400 ways

d) To solve this problem, I can apply the Pigeonhole principle which states that if n items are stored in m, with n > m, then at least one storage must store more than one items.

Considering the question, 14 days will be container for all 10 sessions.

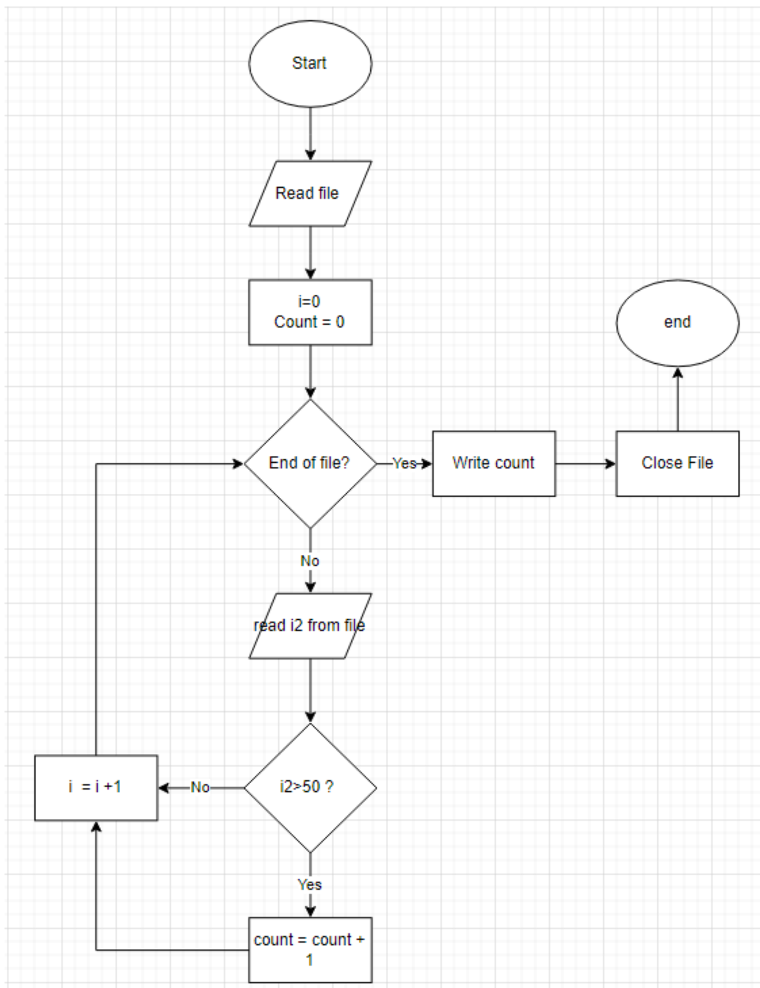
A container contains of 2 consecutive days (1 day to student then 1 with no study)

So, m = $\frac{14}{2}$ = 7. We have 10 study sessions so n = 10, so $\frac{n}{m} = \frac{10}{7}$ = 1.42, we can round up to 2. To sum up, this person has to study at least 2 consecutive days without resting within 14 days to complete his study sessions.

Algorithms

2.

a.



b. The complexity of my diagram is O(n) because as the value of i rises, the runtime rises linearly. The value i is the number of line in the file. The parameters are the if and else statement.

3)

a. The worst case time running of X: $O(n^2)$ should run faster then the Y: $O(n^3)$. However, when the inputs for X and Y are the same which is greater than 1, the runtime of X: $O(n^2)$ is smaller than the Y: $O(n^3)$ as the n go to infinite. So the algorithm X should be recommended.

b. With the X: O(n^2) it should run quicker than the new Y: O(n). However, according to the order of growth table, when the inputs n will go to infinite as the time run will increase with the O(n^2) which much longer than the Y: O(n). I would switch the case and Using the algorithm Y with complexity O(n).

4) Recursive function:

```
Function sum_odd(n)
    If n <= 0 then
        Return 0
    Else if  binary_ones(n) % 2 equal 1 then
        Return n + sum_odd(n-1)
    Else
        Return sum_odd(n-1)
    End if
```

Graph and Trees

5)

a.

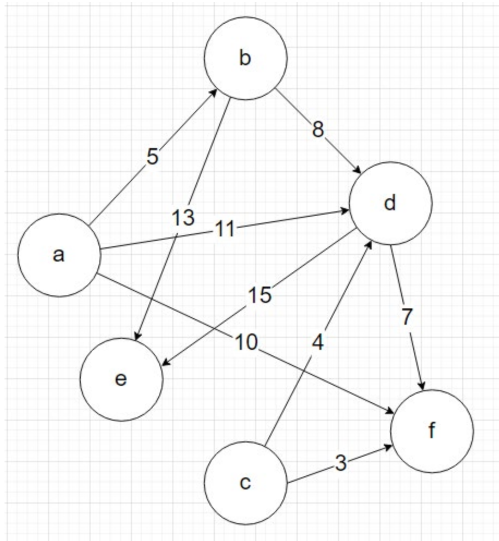
V = {a, b, c, d, e, f}

E = {{a, b}, {a, f}, {a, d}, {b, e}, {b, d}, {c, f}, {c, d}, {d, e}, {d, f}}

W (a, b) = 5, W (a, f) = 10, W (a, d) = 11

W (b, e) = 13, W (b, d) = 8, W (c, d) = 4

W (c, f) = 3, W (d, e) = 15, W (d, f) = 7



Graph of (G(V,E,w))

b.

i.

V = {a, b, c, d, e, f}

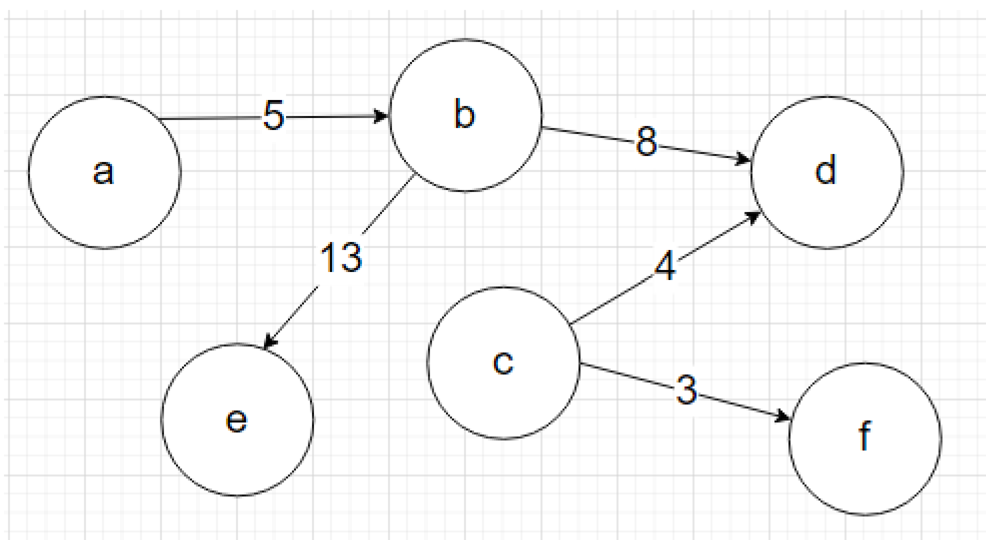
Set S with ordered weight:

S = {{c, f}, {c, d}, {a, b}, {d, f}, {b, d}, {a, f}, {a, d}, {b, e}, {d, e}}

Weight	Start	End
3	c	f
4	c	d
5	c	b
7	d	f
8	b	d
10	a	f
11	a	d
13	b	e
15	d	e

Ordered Edges	Kept	Discard
{c, f}	X	
{c, d}	X	
{a, b}	X	
{d, f}		X
{b, d}	X	
{a, f}		X
{a, d}		X
{b, e}	X	
{d, e}		x

ii.

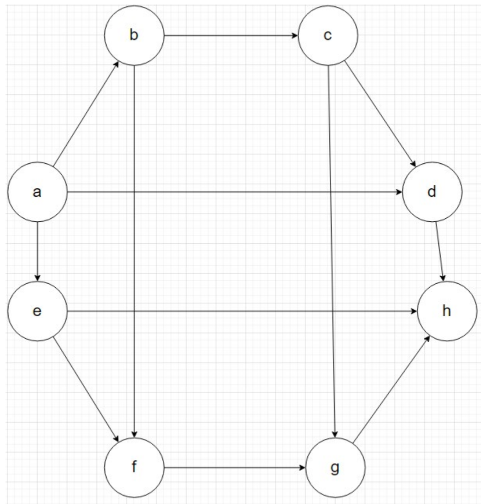


Graph of (F(V,S,w))

6)

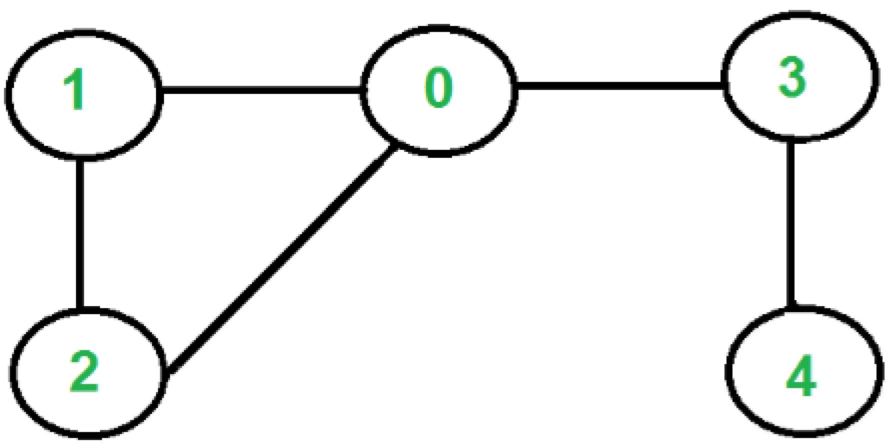
V = {a, b, c, d, e, f, g, h}

E = {{a, b}, {b, c}, {c, d}, {a, d}, {e, f}, {f, g}, {g, h}, {e, h}, {a, e}, {b, f }, {c, g}, {d, h}}



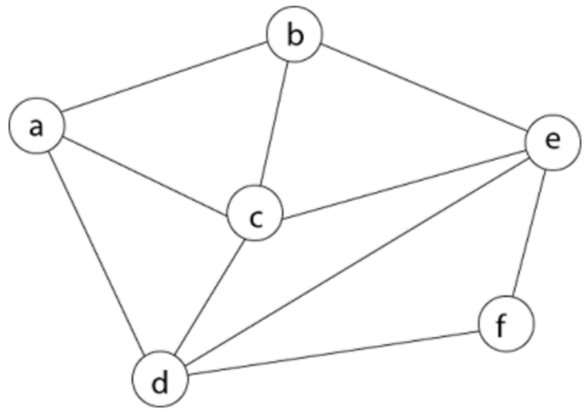
Graph(V,E)

A graph has Eulerian cycle if and only if the degree of every vertex is even in this case, it doesn’t have the Eulerian cycle because all nodes in this graph have 3 degrees (odd number)



Example of Eulerian cycle

For the Hamiltonian cycle, which said that all of the vertices in the graph must be visited exactly one. So, I need to know the start point and the end point. I choose a as the start point and I found the order of cycle is a > b >c > d > h > g > f > e > a, I try with the other point such as b and the order cycle is b > c > d > h > g > f > e > a >b. So the graph below has the Hamiltonian cycle.



Example of Hamiltonian cycle