National University of Singapore
School of Computing

CS3230 - Design and Analysis of Algorithms
Final Assessment

(Semester 1 AY2024/25)

Time Allowed: 150 minutes

INSTRUCTIONS TO CANDIDATES:

1. Do **NOT** open this assessment paper until you are told to do so.
2. This assessment paper contains TWO (2) sections.
   It comprises SIXTEEN (16) printed pages, including this page.
3. This is an **Open Book** Assessment.
   The only allowed electronic device is a non-programmable calculator.
4. For Section A, use the boxes at page 11 (use 2B pencil).
   For Section B, answer **ALL** questions within the **boxed spaces** at page 12-16.
   If you leave the boxed space blank, you will get automatic free 0.5 marks.
   However, if you write at least a single character and it is totally wrong, you will get 0 mark.
   You can use either pen or pencil. Just make sure that you write **legibly**!
5. Important tips: Pace yourself! Do **not** spend too much time on one (hard) question.
   Read all the questions first! Some questions might be easier than they appear.
6. You can assume that all **logarithms are in base** 2.
7. The total marks of this paper is 100 marks.
   It will then be scaled to 40% of the course weightage.

A MCQs ($20 \times 2 = 40$ marks)

1. There are two functions $f(n)$ and $g(n)$ and asymptotically $0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$.
   Which of the following statements is **FALSE**?

   a). $f(n) \in o(g(n))$

   b). $f(n) \in O(g(n))$

   c). $f(n) \in \Theta(g(n))$

   d). $f(n) \in \Omega(g(n))$

   e). None of the above.

2. For any function f and any number $c > 0$, define $f_c^*(n)$ as the smallest number of applications of f needed to transform n into a number that is at most c. More formally,

$$f_c^*(n) = \begin{cases} 0, & \text{if } n \le c, \\ 1 + f_c^*(f(n)), & \text{otherwise.} \end{cases}$$

   For example, if $f(n) = n/2$, then $f_1^*(n) = \lceil \log_2 n \rceil$.
   Which of the following statements is **TRUE**?

   a). If $f(n) = n^{0.1}$ and $g(n) = n^{0.9}$, then $f_7^*(n) \in o(g_7^*(n))$.

   b). If $f(n) = n^{0.1}$ and $g(n) = n^{0.9}$, then $f_7^*(n) \in \omega(g_7^*(n))$.

   c). If $f(n) = \log n$, then $f_3^*(n) \in o(f_3^*(2^n))$.

   d). If $f(n) = \log n$, then $f_3^*(n) \in \omega(f_3^*(2^n))$.

   e). None of the above.

3. Consider the recurrence relation: $T(n) = T(\frac{2n}{3}) + \Theta(1)$. Which statement is **TRUE**?

   a). $T(n) \in O(1)$

   b). $T(n) \in \Theta(\log n)$

   c). $T(n) \in \Theta(n)$

   d). $T(n) \in \omega(n)$

   e). None of the above.

4. Consider the recurrence relation: $T(n) = n^{2/3} \cdot T(n^{1/3}) + n$. Which statement is **TRUE**?

   a). $T(n) \in o(n)$

   b). $T(n) \in \Theta(n)$

   c). $T(n) \in \Theta(n \log n)$

   d). $T(n) \in \omega(n \log n)$

   e). None of the above.

5. Which of the following real-life Divide and Conquer (D&C) algorithms is an example of **case 3** of Master Theorem?

   a). Randomized Quick Select expected runtime of $T(n) = T(\frac{3n}{4}) + O(n)$

   b). Binary Search, $T(n) = T(\frac{n}{2}) + O(1)$

   c). Merge Sort, $T(n) = 2 \cdot T(\frac{n}{2}) + O(n)$

   d). Strassen's Matrix Multiplication, $T(n) = 7 \cdot T(\frac{n}{2}) + O(n^2)$

   e). Karatsuba's Polynomial Multiplication, $T(n) = 3 \cdot T(\frac{n}{2}) + O(n)$

6. There are proposals to split the term D&C into **Divide** (into *two or more* subproblems of smaller size) and Conquer and **Decrease** (into *one* subproblem of smaller size) and Conquer?
   Which D&C algorithms below is **not** of type **Decrease** and Conquer.

   a). Randomized Quick Select

   b). Binary Search

   c). Exponentiation: Computing $a^n \pmod{m}$ in $\Theta(\log(n))$ time

   d). Factorial: Computing $n!$ in $\Theta(n)$ time

   e). Karatsuba's Polynomial Multiplication

7. Consider the following two assertions about a randomized procedure $\mathcal{A}$:

   A1: $\mathbf{E}[\text{runtime of } \mathcal{A}] < \infty$

   A2: $\lim_{t\to\infty} \mathbf{Pr}[\mathcal{A} \text{ does not finish by time } t] = 0$

   Which of the following statements is **TRUE**?

   a). (A1 implies A2) and (A2 implies A1).

   b). (A1 implies A2) and (A2 does not imply A1).

   c). (A1 does not imply A2) and (A2 implies A1).

   d). (A1 does not imply A2) and (A2 does not imply A1).

   e). None of the above.

8. Place n balls into n bins. Each ball placement is done independently and uniformly at random. For each $i \in \{1, 2, \ldots, n\}$, let X_i be the number of balls in the i-th bin. What is the expected value of $\sum_{i=1}^{n} \binom{X_i}{2}$?

   a). $o(n)$

   b). $\Theta(n)$

   c). $\Theta(n \log n)$

   d). $\omega(n \log n)$

   e). None of the above.

9. Suppose the top-down Dynamic Programming (DP) $Fib(n)$ is written in this way:

```
Fib(n)
    if n <= 1, return n
    else
        A[n-1] = Fib(n-1) # original line has "if A[n-1] == None" check
        if A[n-2] == None, A[n-2] = Fib(n-2)
        return A[n-1] + A[n-2]
```

Which of the following statements is **TRUE**?

a). $Fib(n)$ still runs in $O(n)$

b). $Fib(n)$ runs in slow $\Omega(2^{\frac{n}{2}})$ because the `A[n-1]` case is not memoized

c). $Fib(n)$ becomes wrong

d). $Fib(n)$ causes runtime error

e). None of the above

10. The goal of change-making problem is to find the minimum number of coins of denominations d that add up to n cents. If the denominations are $d = \{4, 3, 1, 5\}$, which value of n does **not** work with the greedy strategy of picking the largest denomination that is not larger than n? For example, when $n = 7$, the optimal strategy is $4 + 3 = 7$ (2 coins), but the greedy strategy will pick $5 + 1 + 1 = 7$ (3 coins).

a). $n = 1000$

b). $n = 56$

c). $n = 47$

d). $n = 38$

e). $n = 19$

11. MCQ 11 has wording/differing assumption issues and is thus not archived.

12. Consider the CD burning problem. Suppose Bob has a collection of music files that he wants to burn into CDs with the following constraints.

- Each music file cannot be split and hence cannot be burned into more than one CD.
- Each CD can contain at most two music files.

Given a set A of file sizes and a positive integer $k \geq \lceil |A|/2 \rceil$, define $\mathsf{MinSizeCD}(A, k)$ as the smallest number C such that if each CD has storage capacity C, then k CDs are sufficient to store all the music files described in A. Let $A = \{a_1, a_2, \ldots, a_n\}$ where $a_1 \geq a_2 \geq \cdots \geq a_n > 0$. Which of the following statements is **TRUE**?

a). If $\mathsf{MinSizeCD}(A, k) = a_1$, then $k = |A|$.

b). If $|A|$ is even and $k = |A|/2$, then $\mathsf{MinSizeCD}(A, k) = a_1 + a_n$.

c). If $\mathsf{MinSizeCD}(A, k) = a_1 + a_n$, then $k = |A|/2$.

d). If $k > |A|/2$, then $\mathsf{MinSizeCD}(A, k) = \max\{a_1, \mathsf{MinSizeCD}(A \setminus \{a_1\}, k-1)\}$.

e). None of the above.

13. Consider a variant of the binary counter where a number is represented as a decimal number (base-10 integer) and not a binary number (base-2 integer). Changing a digit (a symbol in $\{0, 1, 2, \ldots, 9\}$) still costs one unit. Start from zero and perform n increments (+1). What are the amortized and worst-case costs per increment?

a). Amortized cost: $\Theta(\log n)$, worst-case cost: $\Theta(\log n)$.

b). Amortized cost: $\omega(1)$ and $o(\log n)$, worst-case cost: $\Theta(\log n)$.

c). Amortized cost: $\Theta(1)$, worst-case cost: $\Theta(\log n)$.

d). Amortized cost: $\Theta(1)$, worst-case cost: $\omega(1)$ and $o(\log n)$.

e). None of the above.

14. Consider a variant of the binary counter where not only increment (+1) but also decrement (-1) is allowed. To ensure that the number never goes below zero, we assume that if the current number is already zero, the decrement does not change the number. Start from zero and perform any n operations (increments or decrements). Which of the following statements is **TRUE**?

a). For some choices of n operations, the worst-case cost per operation can be $\omega(\log n)$.

b). For some choices of n operations, the amortized cost per operation can be $\Omega(\log n)$.

c). The amortized cost per operation is still $\Theta(1)$ for all choices of n operations.

d). While the amortized cost per operation is still $o(\log n)$ for all choices of n operations, the amortized cost per operation can be $\omega(1)$ for some choices of n operations.

e). None of the above.

15. Suppose there is a reduction from problem A to problem B satisfying the following property

- In $O(n^3)$ time, an instance I_A of size n for problem A can be reduced to an instance I_B of size $2n$ for problem B such that the following statement holds for (I_A, I_B):
  - Given a solution for I_B, a solution for I_A can be computed in $O(n^3)$ time.

Which of the following statements is **TRUE**?

a). If B is solvable in $O(n!)$ time, then A is solvable in $O(n!)$ time.

b). If B is solvable in $O(2^n)$ time, then A is solvable in $O(2^n)$ time.

c). If A is solvable in $O(n!)$ time, then B is solvable in $O(n!)$ time.

d). If A is solvable in $O(2^n)$ time, then B is solvable in $O(2^n)$ time.

e). None of the above.

16. Given any decision problems A and B with $A \leq_P B$ (Karp reduction).
Which of the following statements is **TRUE**?

a). ($A \in \mathsf{P}$ implies $B \in \mathsf{P}$) and ($A \in \mathsf{NP}$ implies $B \in \mathsf{NP}$).

b). ($A \in \mathsf{P}$ implies $B \in \mathsf{P}$) and ($B \in \mathsf{NP}$ implies $A \in \mathsf{NP}$).

c). ($B \in \mathsf{P}$ implies $A \in \mathsf{P}$) and ($A \in \mathsf{NP}$ implies $B \in \mathsf{NP}$).

d). ($B \in \mathsf{P}$ implies $A \in \mathsf{P}$) and ($B \in \mathsf{NP}$ implies $A \in \mathsf{NP}$).

e). None of the above.

17. Consider the three complexity classes for decision problems: P, NP, and $\mathsf{NP\text{-}complete}$.
Which of the following statements is **FALSE**?

a). If $\mathsf{P} \cap \mathsf{NP\text{-}complete} \neq \emptyset$, then $\mathsf{P} = \mathsf{NP}$.

b). For all $A \in \mathsf{P}$ and for all $B \in \mathsf{NP\text{-}complete}$, $A \leq_P B$ (Karp reduction).

c). $\mathsf{NP\text{-}complete} \neq \emptyset$.

d). $\mathsf{P} \subseteq \mathsf{NP}$.

e). None of the above.

18. Which of the following sorting algorithms is the best for sorting $n = 10^9$ (1 billion) 64-bit signed integers? Assume that your computer has 1 Gigabytes of RAM (i.e., it can store an array of $\approx 10^8$ 64-bit signed integers in RAM).

a). Insertion Sort

b). Merge Sort

c). Radix Sort with one pass of Counting Sort with $k = 2^{64}$

d). Radix Sort with four passes of Counting Sorts with $k = 2^{16}$

e). Radix Sort with twenty passes of Counting Sorts with $k = 10$ (base 10/Decimal)

19. *Dynamic* selection problem is the problem of finding the i-th smallest element of a data structure where new elements can be added and existing elements can be deleted or updated.
Assume that the data structure is initially empty.
Which of the following strategy solves the dynamic selection problem in $o(n)$ time per operation?

   a). This is not possible: the lower bound of selection is at least n steps

   b). Re-sort the array using insertion sort per each add/delete/update, then report index i

   c). Use Order Statistics Tree

   d). Use Quickselect after each add/delete/update

   e). Use median of medians selection algorithm after each add/delete/update

20. Consider a directed weighted graph $G = (V, E)$ and a source vertex s.
Which of the following statements about shortest paths on G from s is **FALSE**?

   a). Given positive weighted graph, Dijkstra's can fail to produce the shortest paths

   b). Given non-negative weighted graph, BFS algorithm can fail to produce the shortest paths

   c). If the graph has negative-weight cycle reachable from s, the shortest paths are ill defined

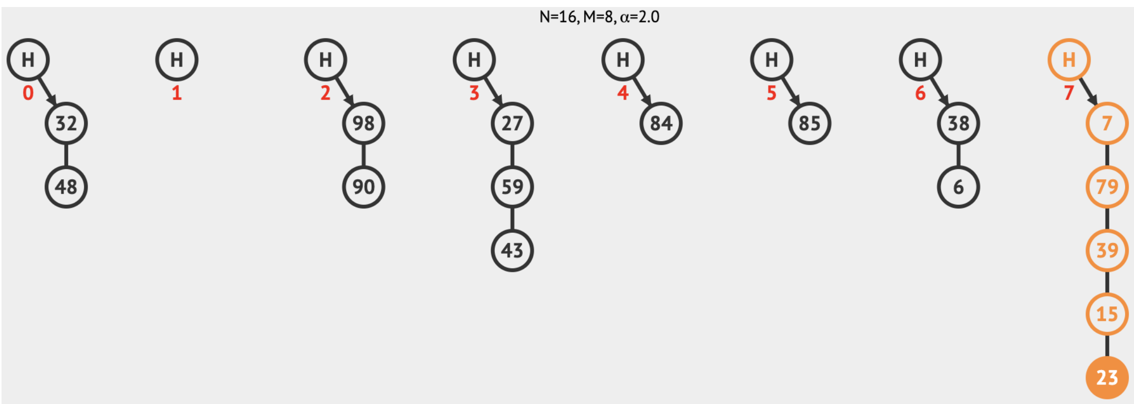
   d). Every subpath of a shortest path is also a shortest path

   e). Given negative weighted graph, Dijkstra's can fail to produce the shortest paths

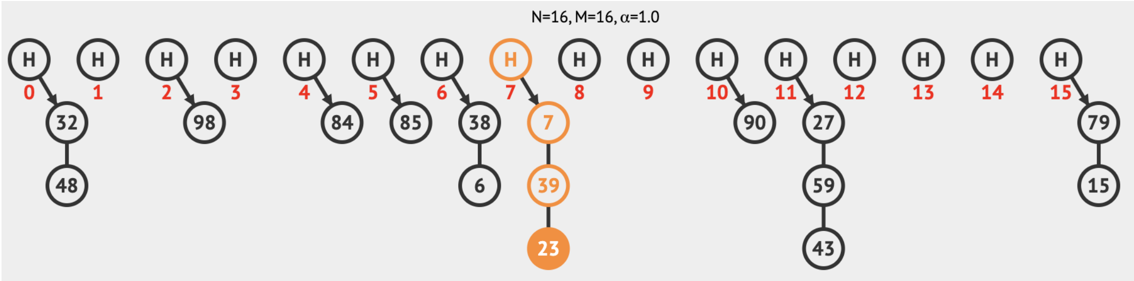
B Essay

B.1 Hash Table Rehash Amortized Analysis (15 marks)

Hash table can be implemented in several ways. Below, we show Separate Chaining implementation. The search performance of such hash table is influenced by the load factor $\alpha = \frac{N}{M}$ (which is also the expected chain length) where N is the number of keys and M is the current table size. For example, given the following hash table with $\alpha = \frac{16}{8} = 2$ (the expected chain length is 2), searching for element 23 can take up to 5 steps as it is contained at a chain/list of length 5 (all 5 of those keys are hashed to the same base address 7 via $h(v) = v\%M$ hash function).



If all of these $N = 16$ keys are rehashed into a **twice bigger** hash table of size $M = 16$, then α decreases to $\alpha = \frac{16}{16} = 1$ and as we can see, the expected chain length is now 1.



A strategy to keep α small is to **double** the hash table size M and rehash the existing N keys if the next insertion causes $\alpha > 2$. A strategy to keep M small is to **halve** the hash table size M and rehash the existing N keys if the next deletion causes $\alpha < \frac{1}{2}$. However, each of rehashing is an expensive $O(N)$ operation. Show that the amortized cost of insertion and deletion are still $O(1)$ even if we rehash all N keys when needed. You can use either aggregate, accounting, or potential method.

Clarification: As discussed in the class, under reasonable assumptions, it is possible to show that the expected cost of both insertion and deletion is at most $\alpha + 1$, where α is the load factor at the beginning of the operation. As randomness is not the focus of this problem, in your solution, you may assume (without a proof) that the cost of both insertion and deletion is at most $\alpha + 1$, which is always $O(1)$, as α is upper bounded by a constant by the design of the data structure. Also, the hash table is initially empty ($N = 0, M = 1$). Hash table size M will never drop to 0.

B.2 Activity Selection Problem (15 marks)

We are given a set of activities $A = \{a_1, a_2, \ldots, a_n\}$, where activity a_i takes place during the time interval $[s_i, f_i)$, with $s_i < f_i$.

- We say that two activities a_i and a_j overlap if $[s_i, f_i) \cap [s_j, f_j) \neq \emptyset$, meaning that their time intervals overlap.
  - We emphasize that each activity a_i overlaps with itself a_i.
- We say that a subset $D \subseteq A$ of activities is dominating if for every activity $a_i \in A \setminus D$, there exists an activity $a_j \in D$ such that a_i and a_j overlap.

We aim to compute a smallest dominating subset $D \subseteq A$ of activities.

B.2.1 Incorrect Attempt (6 marks)

Show that the following greedy algorithm is incorrect:

- $D \leftarrow \emptyset$.
- $B \leftarrow A$.
- While $B \neq \emptyset$, do the following steps.
  - Select an activity $a \in A \setminus D$ to maximize the number of activities in B overlapping with a.

  - Add a to D.
  - For each activity $b \in B$ overlapping with a, remove b from B.
- Return D.

Intuitively, this greedy algorithm aims to maximize the progress in each iteration. For example, the output of the algorithm on the input

$$A = \{[1,2),[2,3),[3,4),[4,5),[5,6),[1,3),[1,4),[3,6)\}$$

is $D = \{[1,4),[3,6)\}$, which is optimal. The algorithm selects $a = [1,4)$ in the first iteration. After that, we have $D = \{[1,4)\}$ and $B = \{[4,5),[5,6)\}$. The algorithm selects $a = [3,6)$ in the second iteration. After that, we have $D = \{[1,4),[3,6)\}$ and $B = \emptyset$.

B.2.2 Correct Algorithm (9 marks)

Design a polynomial-time algorithm that outputs a smallest dominating subset of activities.
Proof of correctness and time complexity analysis are not required for this question.

Hint: Modify the incorrect greedy algorithm above by using a different way to select $a \in A \setminus D$.

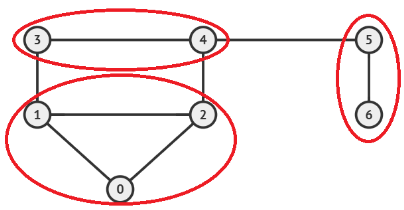
B.3 `3-Clique-Cover` (20 marks)

Given an **undirected, unweighted, and connected** graph $G = (V, E)$ with at least 2 vertices, let S be the set of (all possible) cliques in G. `3-Clique-Cover` is a decision problem to decide whether it is possible to choose **at most 3 cliques** from S such that every vertex in V appears in **exactly one of the cliques**.

We also define the **distance** between 2 distinct vertices in G as the number of edges in the (unweighted) shortest path connecting the 2 vertices. The diameter of a graph, $diam(G)$, is then the **maximum distance** between any 2 vertices in G.

B.3.1 Manual Test Cases (6 marks)

Show your understanding of the definition of $diam(G)$ and `3-Clique-Cover` above on the following graphs (see box B.3.1 of the Answer Sheet). For each graph, state the diameter of the graph and whether it can be covered by **at most 3 cliques**. If it can be covered, please circle the cliques used to cover the graph. Each correct answer worth 1 mark. For the example graph G below, the answer is: ($diam(G) = 4$, YES) and the 3 cliques used to cover G are circled. Note that if the answer is a NO, you need to clearly state it, i.e., ($diam(G)$, NO) and do not circle anything.



B.3.2 Prove `3-Clique-Cover` is in NP (3 marks)

Prove that `3-Clique-Cover` is in NP.

B.3.3 Prove `3-Clique-Cover` is NP-complete (6 marks)

Prove that `3-Clique-Cover` is NP-complete by reduction from `3-Coloring`, which asks whether the vertices of a graph $G = (V, E)$ can be colored using at most 3 colors such that no two adjacent vertices have the same color. You are told that the `3-Coloring` problem is NP-complete.

B.3.4 Solve Two Special Cases (5 marks)

Design the best algorithms to solve `3-Clique-Cover` on G with the following diameters:
(the given graph G is guaranteed to have the stated $diam(G)$, you do not have to check for this again).
Prove the correctness of your algorithms.
Finally, analyze the time complexities of your algorithms using Θ notation.

1. $diam(G) = 1$ (2 marks)
2. $diam(G) \geq 6$ (3 marks)

B.4 Computing Diameter of a Directed Weighted Graph (10 marks)

The definitions in this question are similar *but slightly different* from the previous Section B.3.

Given a **directed, weighted (ones, any positive weights, zeros, or negative weights), and not necessarily connected** graph $G = (V, E)$ with at least 2 vertices, we define the **distance** between 2 distinct vertices in G as the path weight in the **weighted** shortest path connecting the 2 vertices. The diameter of a graph, $diam(G)$, is the **maximum distance** between any 2 vertices in G, which can be ∞ if G is disconnected.

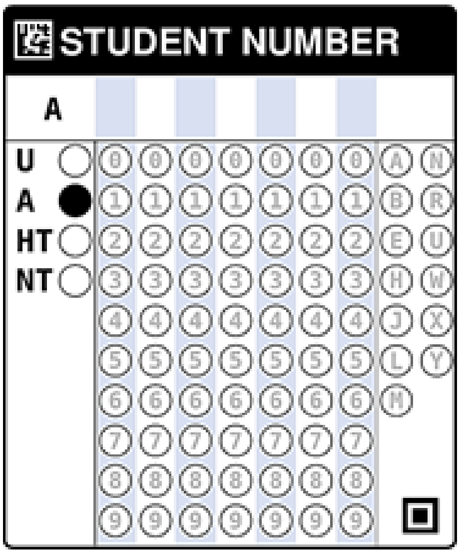
Additional constraints:

- The number of edges in the graph satisfies $|E| \in \Theta(|V|^{1.5})$.
- Also, there is **no negative weight cycle** in the graphs.

Now, design an algorithm to compute $diam(G)$.
You may use any algorithm(s) that you have learned in class.

The Answer Sheet

Write your Student Number in the box below using **(2B) pencil**.
Do NOT write your name.



Write your MCQ answers in the special MCQ answer box below for automatic grading.
We do not manually check your answer.
Shade your answer properly (use (2B) pencil, fully enclose the circle; select just one circle).

No	1	2	3	4	5	6	7	8	9	10
A	○	○	○	○	○	○	○	○	○	○
B	○	○	○	○	○	○	○	○	○	○
C	○	○	○	○	○	○	○	○	○	○
D	○	○	○	○	○	○	○	○	○	○
E	○	○	○	○	○	○	○	○	○	○

No	11	12	13	14	15	16	17	18	19	20
A	○	○	○	○	○	○	○	○	○	○
B	○	○	○	○	○	○	○	○	○	○
C	○	○	○	○	○	○	○	○	○	○
D	○	○	○	○	○	○	○	○	○	○
E	○	○	○	○	○	○	○	○	○	○

Box B.1. Hash Table Rehash Amortized Analysis

Grading scheme: Your answer will be graded using the special marking scheme below.

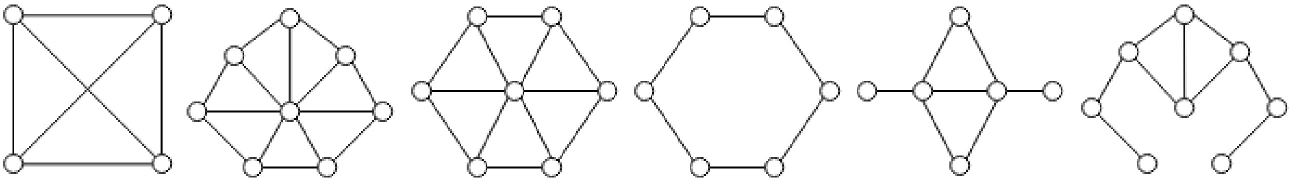
Your amortized analysis	Mark(s)
is wrong	0
is blank	0.5
~~only works for deletion~~	~~4~~
only works for insertion	11
works for both insertion and deletion	15

Tips: Focus on only insertion (11 marks) if considering deletion complicates your amortized analysis.

Box B.2.1. Incorrect Attempt

Box B.2.2. Correct Algorithm

Box B.3.1. Manual Test Cases



Box B.3.2. Prove `3-Clique-Cover` is in NP

Box B.3.3. Prove `3-Clique-Cover` is NP-complete

Box B.3.4.1. Solve Special Case $diam(G) = 1$

Box B.3.4.2. Solve Special Case $diam(G) \geq 6$

Box B.4. Computing Diameter of a Directed Weighted Graph

Grading scheme: Your answer will be graded using the 7-categories special marking scheme below.

Time Complexity	Works-on	Mark(s)
-	wrong answer (only for **unweighted** graphs, or other issue(s))	0
-	blank	0.5
$\omega(\lvert V\rvert^{2.5})$ and $O(\lvert V\rvert^{4})$	all general **non-negative** weighted graphs	6
$O(\lvert V\rvert^{2.5})$	all general **non-negative** weighted graphs	7
$\omega(\lvert V\rvert^{3})$ and $O(\lvert V\rvert^{4})$	all general weighted graphs	8
$\omega(\lvert V\rvert^{2.5})$ and $O(\lvert V\rvert^{3})$	all general weighted graphs	9
$O(\lvert V\rvert^{2.5})$	all general weighted graphs	10

– END OF PAPER; All the Best –