Grade 10 Computing

Unit 2 · Graphs: Maps, Routes and Networks

ANSWER KEY · FOR TEACHERS

Full marks 33

Give the marks for the points in the marking scheme; an answer in other words that says the same is right.

1 Key words

1. node
2. edge
3. degree
4. directed
5. weighted
6. queue
7. tree
8. leaf

Each carries 0.5 mark. **K · 4**

2 Label the diagram

1. root →
2. a leaf
3. Not a tree
4. a loop: two routes from Kathmandu to Thimi

Each carries 0.5 mark. **K · 2**

3 Multiple choice

i. **c.** who follows whom on a social media app **U · 1**

Following goes one way: you can follow someone who doesn't follow you back, so each follow is an arrow. A road, sitting side by side and a shared border all go both ways, so their edges are plain lines.

ii. **c.** a dictionary with each village's list of neighbors **A · 1**

A dictionary of neighbors finds any village's neighbors in one lookup and stores only the trails that exist. The table would have 25 million cells, almost all 0. The list of trails would have to be searched from start to end for each village.

iii. **b.** on the back of the list; the next is taken from the front **U · 1**

New places join the back, and visits are taken from the front: first in, first out, a queue. That is what makes the search go out in rings. Taking from the back instead gives a different search, which dives deep down one path first (see "careful" below).

iv. **b.** about twice as much **A · 1**

Every town is visited once and every list of neighbors read once, so the work grows in step with the size of the graph: linear growth, in Grade 9's words. Twice the towns and roads, about twice the work.

v. **c.** 24 **A · 1**

Every node except the root has exactly one edge up to its parent: $25 - 1 = 24$.

4 What does it print?

MARKING SCHEME **U · 3**

- ✓ `Canteen is a dead end`, the only place with one neighbor (degree 1).
- ✓ `5 paths`: the lists hold 10 names, and each path is in two lists.
- ✓ `False 3`: Lab isn't in Gate's list (no path joins them), and Library's degree is 3.

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking

5 Paint it, ring by ring

ANSWER **A · 4**

The steps of each square, as the flood fill of the unit (`flood_fill(picture, (0, 0))`) gives them, # for black and `.` for the square never reached:

```
OUTPUT
 0  1  2  #  8  9 10 11
 1  #  3  #  7  #  #  #
 2  #  4  5  6  #  .  #
 3  #  #  #  7  #  #  #
 4  5  6  #  8  9 10 11
 #  #  7  8  9  #  # 12
```

ii. The bottom-right square, after **12** steps; **27** squares are painted.

iii. The white square in the third row, seventh column: black squares surround it on all four sides, so no edge leads to it; it is an area of its own.

MARKING SCHEME

- ✓ The first ring numbers right: 0 to 6 down the left side and through the middle gap (0, 1, 2; 1, 3; 2, 4, 5, 6).
- ✓ The two branches from the 6: up to the top right (7, 8, 9, 10, 11) and down and round the bottom (7, 8, 9 … 11, 12).
- ✓ The bottom-right square last, after 12 steps; 27 squares in all.
- ✓ The walled-in square (row 3, column 7) is never painted, because no white neighbor joins it to the rest.

6 Trace a search

ANSWER **A · 4**

Visit	Steps	New places (steps)	List still to visit, front first
(start)		Hari (0)	Hari
Hari	0	Bimal (1)	Bimal
Bimal	1	Asha, Nima (2)	Asha, Nima
Asha	2	Dawa, Gita (3)	Nima, Dawa, Gita
Nima	2	none (Bimal, Dawa and Gita already seen)	Dawa, Gita
Dawa	3	none	Gita
Gita	3	none	(empty)

So Gita is 3 steps from Hari, by Hari → Bimal → Asha → Gita.

MARKING SCHEME

- ✓ Bimal (1), then Asha and Nima (2), each added once, in the order of Bimal's list.
- ✓ Dawa and Gita (3) added when Asha is visited; nothing new when Nima is visited.
- ✓ The list taken from the front and added at the back, empty after the sixth visit.
- ✓ The steps of the visits right: 0, 1, 2, 2, 3, 3 (each new place one more than the place it was seen from).

7 Find and fix the bug

MARKING SCHEME **A · 3**

- ✓ Line 15.
- ✓ `pop(0)`, taking the place at the front of the list (a queue).
- ✓ `pop()` takes the newest place (a stack): the search dives round the ring one way first, so a place is first reached by a long way, and its steps are not the fewest.

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking

8 Write a program

ANSWER A · 4

```
roads = {
    "Kathmandu": ["Lalitpur", "Kirtipur", "Budhanilkantha", "Thimi", "Sankhu"],
    "Lalitpur": ["Kathmandu", "Kirtipur", "Godawari", "Thimi"],
    "Kirtipur": ["Kathmandu", "Lalitpur", "Pharping"],
    "Budhanilkantha": ["Kathmandu"],
    "Thimi": ["Kathmandu", "Lalitpur", "Bhaktapur"],
    "Sankhu": ["Kathmandu"],
    "Godawari": ["Lalitpur"],
    "Pharping": ["Kirtipur"],
    "Bhaktapur": ["Thimi", "Nagarkot", "Banepa"],
    "Nagarkot": ["Bhaktapur", "Dhulikhel"],
    "Banepa": ["Bhaktapur", "Dhulikhel"],
    "Dhulikhel": ["Nagarkot", "Banepa"],
}

def farthest(graph, start):
    """The place most steps from start, and its number of steps."""
    steps = {start: 0}
    to_visit = [start]
    while len(to_visit) > 0:
        place = to_visit.pop(0)
        for nxt in graph[place]:
            if nxt not in steps:
                steps[nxt] = steps[place] + 1
                to_visit.append(nxt)
    far = start
    for place in steps:
        if steps[place] > steps[far]:
            far = place
    return far, steps[far]

print(farthest(roads, "Kathmandu"))
print(farthest(roads, "Pharping"))
```

```
OUTPUT
('Dhulikhel', 4)
('Dhulikhel', 6)
```

Also right: keeping the last place visited (breadth-first search visits the farthest last), or `max(steps, key=steps.get)`.

MARKING SCHEME

- ✓ A dictionary of steps with the start at 0, and a list still to visit holding the start.
- ✓ A loop that takes the front of the list with `pop(0)` and looks at each neighbor.
- ✓ Each neighbor not seen yet gets the visited place's steps + 1 and goes on the back of the list.
- ✓ The place with the most steps found (a highest-so-far loop, the last place visited, or `max`), and both it and its steps returned.

9 Short answer

MARKING SCHEME U · 2

- ✓ The app weights each road by its time (or length) and chooses the smallest total, so 5 short or fast roads can take less time than 3 slow or long ones.
- ✓ Breadth-first search counts every road as one step, so it finds the fewest roads, not the shortest or quickest route (that needs nearest-first search by weight, Dijkstra's algorithm).

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking

10 Think

ANSWER **HA · 2**

The degrees are Gate 2, Office 2, Library 3, Lab 2, Canteen 1. Exactly two places, Library and Canteen, have an odd degree, so by Euler's rule the walk is possible, starting at one of them and ending at the other: for example Canteen → Library → Gate → Office → Lab → Library.

MARKING SCHEME

- ✓ Yes, because exactly two places (Library and Canteen) have an odd degree; the walk must start at one and end at the other.
- ✓ A walk that uses each of the 5 paths once, such as Canteen → Library → Gate → Office → Lab → Library.

11 Activity: Search the valley

(not marked)

WHAT STUDENTS SHOULD FIND

- Dhulikhel: 4, Kathmandu → Thimi → Bhaktapur → Nagarkot → Dhulikhel (the way through Banepa is 4 too, but Nagarkot comes first in Bhaktapur's list).
- Banepa: 3, Kathmandu → Thimi → Bhaktapur → Banepa.
- Godawari: 2, Kathmandu → Lalitpur → Godawari.
- Pharping: 2, Kathmandu → Kirtipur → Pharping.
- Sankhu: 1, Kathmandu → Sankhu.

Marks by level	Knowledge	Understanding	Application	Higher ability	**Total**
Marks	6	7	18	2	**33**

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking