ANSWERS

A4.1 Graphs

Trees and graphs · A level · OCR H446 1.4.2, AQA 7517 4.2.4.1, Eduqas A500QS 1.1 · about 20 min

BugBotLab

What this lesson is about

Vertices and edges; directed, undirected and weighted graphs; degree, the handshake lemma and typical uses.

- Vertices and edges
- Undirected, directed and weighted
- A graph in Python: a list of edges
- Degree and the handshake lemma
- Where graphs are used

Questions

5 marks in all

1. What makes a graph a weighted graph? [1 mark]

- ☐ **A** Every edge has a number such as a distance or cost
- ☐ **B** Every vertex has at least one edge
- ☐ **C** Every edge has a direction
- ☐ **D** The graph has more edges than vertices

> **Answer:** A. A weight is a value on an edge. Direction is a separate property: a graph can be weighted and undirected.

2. Which of these is best modelled by a directed graph? [1 mark]

- ☐ **A** Web pages and the hyperlinks between them
- ☐ **B** Roads that can be driven in both directions
- ☐ **C** Pairs of computers joined by network cables
- ☐ **D** Countries that share a border

> **Answer:** A. A page can link to another page that does not link back, so each link has a direction.

3. An undirected graph has five vertices with degrees 3, 2, 2, 4 and 1. How many edges does it have? [1 mark]

> **Answer:** 6. The degrees add up to 12, and every edge adds 2 to that total, so there are 12 / 2 = 6 edges.

4. What does this program print? [1 mark]

```
EDGES = [("A", "B"), ("B", "C"), ("C", "A"), ("C", "D")]
degree = {}
for u, v in EDGES:
    degree[u] = degree.get(u, 0) + 1
    degree[v] = degree.get(v, 0) + 1
print(degree["C"], sum(degree.values()))
```

> **Answer:**
>
> ```
> 3 8
> ```
>
> C is an end of three edges, and the degrees add up to twice the 4 edges.

5. Which of these statements are true? [1 mark]

Tick every answer that is true.

- ☐ **A** An edge is also called an arc
- ☐ **B** A vertex is also called a node
- ☐ **C** In an undirected graph an edge can be followed in both directions
- ☐ **D** Every graph must be connected

> **Answer:** A, B, C. A graph does not have to be connected: it can be in several separate parts.

The task: degrees and the handshake

Five zones, A to E, are joined by the weighted, undirected edges in `EDGES`. Each edge is a tuple `(end, other end, weight)`, and the weights are whole numbers of seconds. Work out the degree of every vertex from `EDGES`. Print one line per vertex, in alphabetical order, in the form `A: degree 2`. Then print three more lines: `edges:` and the number of edges, `sum of degrees:` and the total of all the degrees, and `total weight:` and the sum of every edge's weight. Do not type any of the numbers: work them out, so the program would still be right if `EDGES` changed.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

EDGES = [("A", "B", 4), ("A", "C", 7), ("B", "C", 2), ("B", "D", 5), ("C", "D", 3), ("C",
"E", 6), ("D", "E", 1)]

degree = {}
```

The hint students can ask for: Every edge touches two vertices, so each edge adds one to the degree of both of its ends. Keep a count per vertex, then add up the counts and the weights.

A solution

```
from bugbot import *
connect()

EDGES = [("A", "B", 4), ("A", "C", 7), ("B", "C", 2), ("B", "D", 5), ("C", "D", 3), ("C",
"E", 6), ("D", "E", 1)]

degree = {}
total = 0
for u, v, w in EDGES:
    degree[u] = degree.get(u, 0) + 1
    degree[v] = degree.get(v, 0) + 1
    total = total + w
for vertex in sorted(degree):
    print(f"{vertex}: degree {degree[vertex]}")
print("edges:", len(EDGES))
print("sum of degrees:", sum(degree.values()))
print("total weight:", total)
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.