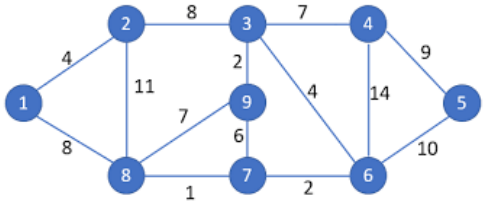
Started on	Sunday, 13 November 2022, 12:31 PM
State	Finished
Completed on	Sunday, 13 November 2022, 12:46 PM
Time taken	14 mins 53 secs
Grade	**20,00** out of 20,00 (**100**%)

Question 1

Correct

Mark 4,00 out of 4,00

Considere a figura seguinte e indique qual a **afirmação verdadeira**:



Select one:

a. Nenhuma das apresentadas; ✔

b. É um grafo com um vértice isolado e sem arestas paralelas;

c. É um dígrafo valorado e sem vértices isolados;

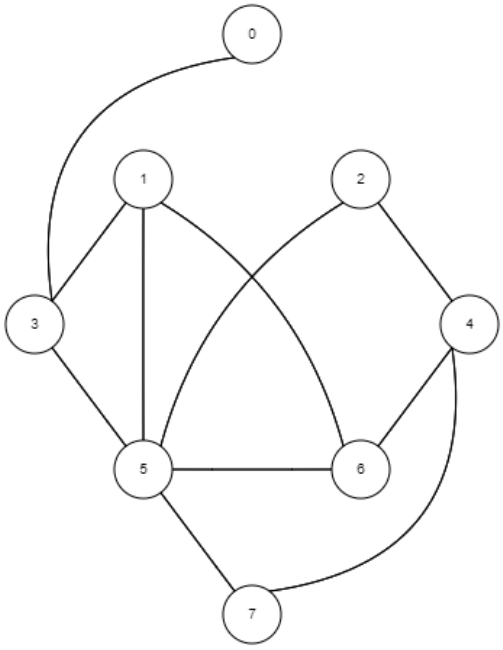
d. É um grafo rotulado e com arestas paralelas;

Question 2

Correct

Mark 4,00 out of 4,00

Considere o grafo da figura abaixo. Indique qual a sequência que se considera correta, aplicando a estratégia de travessia **BFS** (breadth-first search), **partindo do vértice** 5. Nota: Considere que os vértices adjacentes a um vértice são sempre percorridos do numero menor para o maior.



Select one:

a. Nenhuma das apresentadas;

b. `5 1 2 3 6 7 0 4`; ✔

c. `5 1 3 0 7 4 2 6`;

d. `5 7 4 6 1 3 0 2`;

Question 3

Correct

Mark 4,00 out of 4,00

Indique qual dos seguintes pseudo-códigos traduz o algoritmo que **determina o grau de um vértice**.

```
Algorithm: Algorithm_1(graph, vertex)
    input: graph, vertex to calculate the degree
    output: natural number - degree of the vertex
BEGIN
    count <- 0
    FOR EACH edge e in incident_edges(vertex)
        w <- opposite(vertex, e)
        IF w = vertex THEN
            count <- count + 1
        END IF
    END FOR
    RETURN count
END

Algorithm: Algorithm_2(graph, vertex)
    input: graph, vertex to calculate the degree
    output: natural number - degree of the vertex
BEGIN
    count <- 0
    FOR EACH vertex w in graph
       count <- count + 1
    END FOR
    RETURN count
END

Algorithm: Algorithm_3(graph, vertex)
    input: graph, vertex to calculate the degree
    output: natural number - degree of the vertex
BEGIN
    count <-0
    FOR EACH edge e in incident_edges(vertex)
        count <- count + 1
    END FOR
    RETURN count
END

Algorithm: Algorithm_4(graph, vertex)
    input: graph, vertex to calculate the degree
    output: natural number - degree of the vertex
BEGIN
    count <-0
    FOR EACH vertex v in graph
        FOR EACH vertex w in graph
            IF adjacente(v, w) THEN
               count <- count + 1
        END FOR
    END FOR
    RETURN count
END
```

Select one:

- a. `Algorithm_3`; ✔
- b. `Algorithm_2`;
- c. `Algorithm_4`;
- d. `Algorithm_1`;

Question 4

Correct

Mark 4,00 out of 4,00

Considere as classes abaixo apresentadas. Qual o *output* do programa ao executar o método `main`?

```
public class Local {
    private String name;
    public Local(String name) {
        this.name = name;
    }
    public String getName() { return this.name; }
}

public class Bridge {
    private String name;
    public Bridge(String name) {
        this.name = name;
    }
    public String getName() { return this.name; }
}

public class Main {
    public static void main(String[ ] args) {
        Graph<Local, Bridge> graph = new GraphEdgeList<>();
        Vertex<Local> a = graph.insertVertex(new Local("A"));
        Vertex<Local> b = graph.insertVertex(new Local("B"));
        Vertex<Local> c = graph.insertVertex(new Local("C"));
        Vertex<Local> d = graph.insertVertex(new Local("D"));
        Vertex<Local> e = graph.insertVertex(new Local("E"));

        graph.insertEdge(a, b, new Bridge("a"));
        graph.insertEdge(a, b, new Bridge("b"));
        graph.insertEdge(a, c, new Bridge("c"));
        graph.insertEdge(c, d, new Bridge("g"));
        graph.insertEdge(a, d, new Bridge("e"));
        graph.insertEdge(d, e, new Bridge("f"));

        for(Edge<Bridge,Local> edge : graph.incidentEdges(d)) {
            Vertex<Local> v = graph.opposite(d, edge);
            System.out.print(v.element().getName() + " ");
        }
    }
}
```

Select one:

- a. Nenhum dos apresentados;
- b. `A E C` (não interessa a ordem); ✔
- c. `D E A C` (não interessa a ordem);
- d. `g e f` (não interessa a ordem);

Question 5

Correct

Mark 4,00 out of 4,00

Considere a seguinte tabela resultante de execução do algoritmo *Dijkstra* num grafo com 8 vértices (0 - 7), tendo como vértice de origem o vértice 5. **Qual o custo** do menor caminho para chegar ao vértice 1?

Vertex	Cost	Previous Vertex
0	3	2
1	6	6
2	1	5
3	14	1
4	Infinity	null
5	0	null
6	2	5
7	18	3

Select one:

a. Infinito, pois não existe caminho possível;

b. 8;

c. 6; ✔

d. 9;