Four kinds of graphs

You want to improve the communications network in a small town. Historically, all communication between households in the city has taken place with the help of homing pigeons. In this manner, everyone could reach each other's houses directly. But homing pigeons are impractical, so you wish to find better ways for households to communicate with one another. You have explored the following possibilities:

- The new startup HamiltonBikes has begun building new bike routes and will operate a letter delivery service between all houses. They guarantee that you can take a bike ride around the whole city without having to pass the same house twice.
- Houses can be connected with network cables. In order to mitigate cost, redundant connections will be avoided.
- Some households are so close that neighbors can shout for quick, one-to-one communication.

You realize that each form of communication can be formalized as a graph where the nodes are the households and the edges are the connections between them. You identify four types of graphs, each of which corresponds to a form of communication:

Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

There is no need to memorize the graph types, they are repeated where needed.

(a)

Your first task is to check whether HamiltonBikes really delivers what they promise. They offer to send you a map of the bike paths and a suggested bike route which passes each house precisely once. You translate the problem into a decision problem:

Type2	
Instance:	A graph G
Question:	Does G contain a hamiltonian cycle?

You decide to create an efficient algorithm that verifies **Type2**. The algorithm takes a graph and a certificate as input, and answers YES if the certificate confirms that the graph is of type 2, NO otherwise. The algorithm should run in polynomial time.

HamiltonBikes delivers the certificate as an array C of nodes in G, so that each node is included exactly once. Write pseudocode for a verifier `Type2Verifier`.

ALGORITHM: VERIFIER FOR **TYPE2**

Input: A graph G and an array of nodes C
Output: Returns YES if C shows that G is of type 2, NO otherwise

```
1 Procedure Type2Verifier(G, C)
  |   // Fill in
```

(b)

Explain briefly (maximum 4 sentences) why the answer to the previous exercise shows that the decision problem **Type2** is in NP. Do not include a detailed runtime analysis of your implementation of `Type2Verifier`.

If your answer to the previous exercise has shortcomings, you can make necessary assumptions to answer this exercise.

(c)

Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

One advantage of the HamiltonBikes bike paths is redundancy. For example, if all the bike paths around a house have to be closed due to construction work, it is still possible to cycle around the city.

Explain briefly (maximum 4 sentences) why type 2 graphs are biconnected.

(d)

Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

The city council does not understand graph theory, so they want you to demonstrate how one can get around the city if construction work is being done. You decide you to write a program that prints two distinct paths from a given house to another.

You start with a graph $G = (V, E)$ of type 2, as well as two nodes s and t in the graph.

Write pseudocode for a procedure `TwoPaths` that prints two distinct paths from s to t.

You have been given a certificate C as described in exercise (a).

Do not worry about the print format; the order in which nodes are printed is most important.

ALGORITHM: PRINTS TWO DISTINCT PATHS FROM s TO t

Input: A graph G, two nodes s and t and a certificate C
Output: Prints two distinct paths from s to t

```
1 Procedure TwoPaths(G, C, s, t)
  |  // Fill in
```

(e)

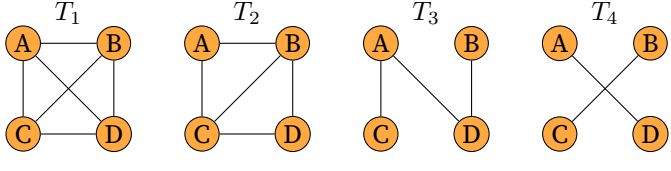
Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

We wish to get an overview of the methods of communication in the city. Therefore, we assign weights to the different edge types based on cost. Then we merge these into a graph S. We only keep the edge with the lowest weight if there are parallel edges.

Here are four sample graphs, one of each type, where

- edges in T_1 have weight 10
- edges in T_2 have weight 8
- edges in T_3 have weight 3
- edges in T_4 have weight 1



Fill out the weights of the edges in the graph S.

(f)

Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

You construct the merged graph G in the same manner as in the previous exercise, given the following graphs:

- G_1 of type 1, where every edge has weight 10,
- G_2 of type 2, where every edge has weight 8,
- G_3 of type 3, where every edge has weight 3,
- G_4 of type 4, where every edge has weight 1.

You now wish to to find the cheapest way to connect the houses in the city. Therefore, you have decided to use Kruskal's algorithm on the merged graph G. Answer the questions below and briefly justify your answer (maximum 4 sentences).

- Is there exactly one cheapest way to connect the houses in the city?
- Is it possible for everyone in the city to communicate only by shouting? In other words, could we have built a spanning tree for G, with only edges from G_4?
- Your colleague says that if G_4 has no edges, then the minimum spanning tree for G is exactly G_3. Is this correct?
- What is the weight of the *last* edge Kruskal's algorithm chooses when building the spanning tree?

(g)

Type 1 is an undirected complete graph (homing pigeons)
Type 2 is an undirected graph with a hamiltonian cycle (bike route)
Type 3 is an undirected tree (network cables)
Type 4 is an undirected non-connected graph (shouting)

All graphs in this exercise has more than two nodes.

Let $G = (V, E)$ be a graph and let s and t be nodes in V. We will now find the length of the shortest path from s to t.

- Which algorithm solves the problem most effectively if you know that G is of type 1 and all the edges have a weight of 10?
- Which algorithm solves the problem most effectively if you know that G is of type 2 and all the edges have a weight of 8?

(h)

You wish to find the shortest route of communication from the post office (A) to all houses in the city, formalized as nodes in the set V. You have represented all types of communication in the following sets of edges:

- E_1 contains an edge between every pair of nodes in V.
  All edges in E_1 has weight 10.
- $E_2 = \{\{A, B\}, \{B, C\}, \{C, D\}, \{D, E\}, \{E, F\}, \{F, G\}, \{G, H\}, \{E, H\}, \{F, H\}, \{F, A\}\}$
  All edges in E_2 has weight 8.
- $E_3 = \{\{A, C\}, \{B, C\}, \{C, D\}, \{D, E\}, \{E, F\}, \{F, G\}, \{G, H\}\}$
  All edges in E_3 has weight 3.
- $E_4 = \{\{A, D\}, \{C, D\}, \{B, D\}, \{F, G\}\}$
  All edges in E_4 has weight 1.
- $V = \{A, B, C, D, E, F, G, H\}$

Find the shortest paths from A to all other nodes. You can use edges from all four sets of edges. Remember that there can be several paths of communication between houses, i.e., there are parallel edges between the nodes.

Fill out the table with the correct minimal distance from A for each node in V.