Question 1 (1 marks)
How does python pass parameters to a function? Can the function change the value of the parameter in the outer scope?

Answer: Passes the parameters by call by assignment. It is if the when the function is called, there are assignments from the parameters passed in to the local variables. It a parameter is mutable, then the function can change it and the effect will be seen in the outer scope.

Question 2 (1 marks)
What does the following output

```python
def incr(x, y):
    x += y
    return x

a = [1, 2, 3]
b = incr(a, [4, 5])
print(a)
print(b)
c = 5
d = incr(c, 2)
print(c)
print(d)
e = "123"
f = incr(e, "45")
print(e)
print(f)
```

Answer:

Question 3 (1 marks)
What does the following output

```python
for i in range(10):
    print(i)
    i += 1
```

Answer:
Graph Representation

Question 4 (1 marks)
Describe how to represent a graph through an adjacency list representation and through an adjacency matrix representation. Your answers should be detailed enough that someone could implement it.

Answer:

Question 5 (2 marks)
The choice of representation can affect running time. Give an algorithm (or refer to one given in this exam) and explain how its running time differs under the two representations. Make sure you give the actual running time under both representations.

Answer:

Question 6 (1 marks)
For any graph, give the strongest relationship that is true between the number of vertices and number of edges (using the quantities below) and involving ≤:

| log(V) | |V|-1 | V | V log(V) | V^2 |
| log(E) | |E|-1 | E | E log(E) | E^2 |

Students found this question confusing.

Answer: \( |E| \leq |V|^2 \)

Question 7 (1 marks)
If an undirected graph is connected (every node is reachable from every other node), give another relationship.

Students found this question confusing. Didn’t realize it is connected to the previous question.

Answer: \( |V| - 1 \leq |E| \)

Depth-First Search

Question 8 (2 marks)
Fill in the boxes for the code of Depth-First Search.

DFS(G)
1 for each vertex \( u \in G.V \)
2 \( u.\text{color} = \text{WHITE} \)
3 \( u.\pi = \text{NIL} \)
4 time = 0
5 for each vertex \( u \in G.V \)
6 if \( u.\text{color} = \) [ ]
7 DFS-VISIT(G,u)

DFS-VISIT(G,u)
1 time = time + 1
2 u.d = [ ]
3 u.color = [ ]
4 for each \( v \in G[u] \)
5 if \( v.\text{color} = \) [ ]
6 \( v.\pi = \) [ ]
7 DFS-VISIT(G,v)
8 u.color = [ ]
9 time = time + 1
10 u.f = [ ]
Answer:

DFS(G)
1 for each vertex $u \in G.V$
2 $u.color = \text{WHITE}$
3 $u.\pi = \text{NIL}$
4 time = 0
5 for each vertex $u \in G.V$
6 if $u.color == \text{WHITE}$
7 DFS-VISIT(G,u)

DFS-VISIT(G,u)
1 time = time + 1
2 $u.d = \text{time}$
3 $u.color = \text{GRAY}$
4 for each $v \in G[u]$
5 if $v.color == \text{WHITE}$
6 $v.\pi = u$
7 DFS-VISIT(G,v)
8 $u.color = \text{BLACK}$
9 time D time C 1
10 $u.f = \text{time}$

Question 9 (1 marks)
Parenthesis Theorem: In any depth-first of a (directed or undirected) graph $G = (V,E)$, for any two vertices $u$ and $v$, exactly one of the following three conditions holds. List those 3 possibilities.

Answer:

Topological Sort

Question 10 (1 marks)
Let $G = (V,E)$ be a directed acyclic graph, and a topological sort puts the vertices in the order $v_1, v_2, ..., v_n$. What must be true about the edges of $G$?

Students found this question confusing. Perhaps say: what must be true about the edges and the order of the vertices.

Answer: If $(v_i, v_j) \in E, i < j$

Question 11 (1 marks)
Explain how to use depth first search to produce a topological sort.

Answer:

Minimum Spanning Trees

Question 12 (2 marks)
A cut $(S, V-S)$ of an undirected graph $G = (V,E)$ is a partition of $V$. We say that an edge $(u,v) \in E$ crosses the cut if:

Answer: one of its endpoints is in $S$ and the other in $V-S$

We say that a cut respects a set $A$ of edges if:

Answer: no edge in $A$ crosses the cut

An edge is a light edge crossing a cut if:
Answer: its weight is the minimum any edge crossing the cut

Let $A$ be a set of edges that is a subset of a minimum spanning tree. An edge $e$ is a **safe edge** for $A$ if:

Answer: $A \cup \{e\}$ is a subset of a minimum spanning tree

Theorem 23.1:
Let $G = (V, E)$ be a connected, undirected graph with $w : E \to \mathbb{R}$.
Let $A$ be subset of $E$ that is included in some min-spanning tree for $G$.
Let $(S, V - S)$ be any cut of $G$ that respects $A$.
Let $(u, v)$ be a light edge crossing $(S, V - S)$.
Then edge $(u, v)$ is:

Answer: a safe edge for $A$

**Question 13** (1.5 marks)
Here is Prim’s algorithm for determining a minimum span. On the graph below, starting Prim’s algorithm with vertex $a$ (which is highlighted), indicate which edges are added to the minimum spanning tree (my making them darker), and the order that they are added (by adding a number to each). The graph is included twice in case you make an error.

MST-PRIM($G, w, r$)
1. for each $u \in G.V$
2. $u.key = \infty$
3. $u.\pi = \text{NIL}$
4. $r.key = 0$
5. $Q = G.V$
6. while $Q \neq \emptyset$
7. $u = \text{EXTRACT-MIN}(Q)$
8. for each $v \in G.Adj[u]$
9. if $v \in Q$ and $w(u, v) < v.key$
10. $v.\pi = u$
11. $v.key = w(u, v)$

![Graph](image-url)
Question 14  (1.5 marks)
Here is Krushal's algorithm for determining a minimum span. On the graph below, indicate which edges are added to the minimum spanning tree (by making them darker), and the order that they are added (by adding a number to each). The graph is included twice in case you make an error.

MST-KRUSKAL(G, w)
1  A = ∅
2  for each vertex v ∈ G.V
3     MAKE-SET(v)
4  sort the edges of G.E into nondecreasing order by weight w
5  for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
6     if FIND-SET(u) ≠ FIND-SET(v)
7       A = A ∪ {(u, v)}
8     UNION(u, v)
9  return A
Single Source Shortest Paths

Question 15  (1.5 marks)

Lemma 24.1
Given a weighted direct graph, if there is a shortest path from \( u \) to \( v \) and it goes through \( x \) and \( w \), the subpath from \( x \) to \( w \) is a shortest path from \( x \) to \( w \).

Prove this lemma.

Answer:

Question 16  (2 marks)

Let \( \delta(s,v) \) be the weight of the shortest path from the source \( s \) to any vertex \( v \). Single source shortest path algorithms use \( v.d \) to store the current estimate of the weight from \( s \) to \( v \). And \( v.d \) and \( v.\pi \) are subsequently only changed by the relax algorithm given below. Because of this, a number of properties hold of \( v.d, v.\pi \) and \( \delta(s,v) \). Fill in the blanks of these properties.

\[ 
\text{INITIALIZE-SINGLE-SOURCE}(G,s)
\]

1. \textbf{for} each vertex \( v \in G.V \)
2. \( v.d = \infty \)
3. \( v.\pi = \text{NIL} \)
4. \( s.d = 0 \)

\[ 
\text{RELAX}(u,v,w)
\]

1. \textbf{if} \( v.d > u.d + w(u,v) \)
2. \( v.d = u.d + w(u,v) \)
3. \( v.\pi = u \)

Triangle Inequality:
For any edge \((u,v) \in E\), \( \delta(s,v) \leq \delta(s,u) + \boxed{ } \)
Upper-bound Property
\[ v.d = \delta(s, v) \] for all \( v \in V \). \( v.d \) only decreases in value. (Fill in a relationship: = > < ≥ ≤.)

No-path property
If there is no path from \( s \) to \( v \) then we always have \( v.d = \delta(s, v) = \) \[ \]

Convergence Property
If \( s \leadsto u \rightarrow v \) is a shortest path in \( G \) for some \( u, v \in V \)
and if \( u.d = \delta(s, u) \) at any time prior to relaxing edge \( (u, v) \)
then \( v.d = \) \[ \] at all times afterward.

Path-relaxation Property
If \( p = \langle v_0, v_1, ..., v_k \rangle \) is a shortest path from \( s = v_0 \) to \( v_k \)
And we relax the edges of \( p \) in the order \[ \]
then \( v_k.d = \delta(s, v_k) \).

Predecessor-subgraph property
Once \( v.d = \) \[ \] for all \( v \in V \), predecessor subgraph is a shortest-paths tree rooted at \( s \)

All Pairs Shortest Paths

Question 17 (1.5 marks)
All pairs shortest paths rely on dynamic programming. The textbook gives two different recursion relationships that can be derived. Give one of them. Give a written description of it as well as a mathematical formula.

Answer:

Question 18 (1 marks)
What will be the complexity of the code will be that is based on that relationship. Justify your answer.

Answer:

Question 19 (1.5 marks)
Explain how your program will determine that the best path from 2 to 3 has weight 6, and how it will determine that the weight of the best path from 3 to 2 has weight -1. Start with what the original estimate will be for the shortest path, and then explain how it changed.

Answer:
String Matching

Question 20 (1.5 marks)
Let \( P \) be a pattern of \( m \) characters, and \( T \) be a text of \( n \) characters.
Explain how the Rabin-Karp Algorithm works.

Answer:

Question 21 (1 marks)
Explain how the Rabin-Karp algorithm is able to do the matching step in time close to \( O(n) \), regardless of how large \( m \) is.

Answer:

Question 22 (1 marks)
Give the finite state automata to recognize the string \( abab \)