**Multiple Choice Answers Question 1 – 10**

1 C

2 D

3 C

4 B

5 C

6 A

7 B

8 D

9 D

10 B

**Question 11**

Latency – ping (plus response time). Measure of round-trip time between request and response.

Jitter – latency variance / delay inconsistencies. Rate at which ping changes **over time**.

Best effort QoS can be impacted by these factors during periods of high congestion (hardware or ISP differences aside). Results in high lag, slower app response, etc.

IP – Internet Protocol. Protocol for addressing and routing data on the internet.

 TCP Transmission Control Protocol – divides files into numbered data packets, forwards them individually to the IP layer. It then waits until packets have arrived, acknowledges unreceived packets, and retransmits those that aren’t.

UDP User Datagram Protocol – like TCP but doesn’t acknowledge, resend or wait. Just keeps sending. Low latency. Good for gaming.

HTTP – Hypertext Transfer Protocol – transfers hypermedia documents. Generally used for websites.

FTP – File Transfer Protocol – transfers files.

**Question 12**



**Question 13**

|  |  |  |
| --- | --- | --- |
| Character (string) | Result (array values) | Valid (Boolean) |
| T | T | True |
| 1 | T1 | False |
| 2 | T12 | False |
| X | T12S | True |
| 4 | T12S4 | False |
| 5 | T12S45 | True |

* Allows letters in barcode outside of character 0 (error)
* Does not return empty array if invalid (error)
* Describes Valid as True when barcode isn’t during runtime (anomaly)
* Continues to process even after invalid barcode (inefficient)

There are many methods to code a solution to this problem. Here is one attempt from a student:

|  |
| --- |
| BEGIN barcode scanner algorithm SET barcode = empty array SET processed = False SET valid = True SET counter = 0 WHILE NOT processed INPUT next character in barcode sequence into variable character IF counter == 0 THEN IF character == "T" OR character == "S" THEN SET barcode[counter] == character ELSE SET valid = False SET processed = True END IF  ELSE IF *digit*(character) == True THEN SET barcode[counter] == character ELSE SET valid = False SET processed = True END IF  END IF  SET counter = counter + 1 IF counter == 6 THEN SET processed = True END IF  END WHILE IF valid == True THEN RETURN barcode ELSE RETURN empty array END IFEND barcode scanner algorithm |

In Python, student attempted this as well, not part of the exam but FYI. Does this work? Try it:

|  |
| --- |
| barcode = []processed = Falsevalid = Truecounter = 0while not processed: character = input("Next char in barcode sequence: ") if counter == 0: if character.upper() in ["T","S"]: barcode.append(character) else: valid = False processed = True  else: if character.isdigit(): barcode.append(character) else: valid = False processed = True   counter = counter + 1 if counter == 6: processed = Trueprint("valid: ", valid) |

**Question 14**

 Join condition was problem:

SELECT SUM(retailPrice \* quantity) AS Profit

FROM ( Pricing JOIN Orders

 ON Pricing.juice == Orders.juice)

 JOIN Updates

 ON Orders.orderNum == Updates.orderNum

WHERE status == "complete"

Without selling any juices with chia seed in them:

SELECT SUM(retailPrice \* quantity) AS Profit

FROM ( Pricing JOIN Orders

 ON Pricing.juice == Orders.juice)

 JOIN Updates

 ON Orders.orderNum == Updates.orderNum

WHERE status == "complete" AND Orders.juice NOT IN (

 SELECT juice

 FROM Recipes

 WHERE ingredient == "Chia"

)

**Question 15**

Taken straight from class notes:

BEGIN **Vigenère**
 VARIABLE counter = 0
 INPUT plaintext as Array
 INPUT key1 as Array
 key1 = KEYWRAP(plaintext, key1)
 VARIABLE ciphertext = []
 WHILE counter < length(plaintext)
 base = BASE26(plaintext[counter])
 key = BASE26(key1[counter])
 shift = (base + key) mod 26
 ciphertext[counter] = UNICODE\_CHARACTER(shift)
 counter = counter + 1
 END WHILE
END **Vigenère**

BEGIN KEYWRAP(plaintext, key):
 index = 0
 counter = length(key)
 WHILE counter < length(plaintext)
 value = key[index]
 key[counter] = value
 IF index < length(key) THEN
 index = index + 1
 ELSE
 index = 0
 END IF
 counter = counter + 1
 END WHILE
 RETURN key
END KEYWRAP

BEGIN BASE26(Unicode char)
 RETURN ordinal(char) – ordinal("A")
END BASE26

BEGIN UNICODE\_CHARACTER(Base26 integer)
 RETURN character(integer + ordinal("A"))
END UNICODE\_CHARACTER

**Question 16**

Pseudocode for Luhn:

|  |
| --- |
| BEGIN luhn INPUT card numbers INTO card\_array REVERSE card\_array SET sum TO 0 SET counter TO 0 WHILE counter < **LENGTH**(card\_array) IF counter **MOD** 2 <> 0 THEN SET double = card\_array[counter] \* 2 IF double > 9 THEN **SPLIT** doubleINTOdigit1ANDdigit2 SET sum = sum + digit1 + digit2 ELSE SET sum = sum + double END IF ELSE SET sum = sum + card\_array[counter] END IF SET counter = counter + 1 END WHILE IF card\_sum % 10 == 0 THEN OUTPUT "VALID" ELSE OUTPUT "INVALID" END IFEND luhn |

In Python, not part of the exam but FYI:

|  |
| --- |
| card = input("number: ")card\_array = [int(char) for char in str(card)]card\_array.reverse()card\_sum = 0for index, number in enumerate(card\_array): if index % 2 != 0: double = number \* 2 if double > 9: split = [int(char) for char in str(double)] card\_sum += split[0] + split[1] else: card\_sum += double else: card\_sum += numberif card\_sum % 10 == 0 and card != "0": print("VALID")else: print("INVALID") |

Online transaction safety and security, answer could include:

* TLS (the new SSL) – encrypts data sent using asymmetric encryption RSA (very strong). Server should redirect all traffic through HTTPS.
* Hashing (min SHA512) with a salt of passwords that require session login.
* Encryption of any stored credit cards server-side must be strong – AES 128 or higher (preferably AES 256)
* Other procedures client and server-side could be mentioned (clear cache / don’t store passwords client-side, server-side security audits / vulnerability testing, etc.)

