1.1 What Is Python? What are the benefits of using python?

Python is a programming language. It’s OOP and interpreted (doesn’t need to compile). It’s not typed like Java. Pluses: similar to spoken language, intuitive for that. Not typed can be a plus. Not compiled is a plus -> faster. Not typed can also be a minus due to possible errors.

1.2 What does ‘string’ method count() do, provide an example.

It counts how many times the text given in brackets appears in a strong.

EX:

```
txt = "I like music, chocolate, cinnamon and games."
x = txt.count("music")
print(x)
```

1.3 What do functions ord() and chr() do, provide examples

chr() is used for converting an Integer to a Character

ord() is used to convert a Character to an Integer

EX:

```
print(chr(0x41))
i = ord('A')
print(i)
```

1.4 Given string ‘2020-11-10_sales.csv’ how can I get ‘2020-11-10_sales’ using slicing ?

```
txt = '2020-11-10_sales.csv'
print(txt[:-4])
```

1.5 What is the difference between List and Tuple data types in Python?

Lists have [], tuples have (), tuples are faster, tuples are immutable, lists are changeable, lists are better for performing the DML operations (insert, delete etc.) and have some methods while tuples do not.

1.6 What is the difference between List methods append() and extend(), provide examples.

Append adds new element(s) to an existing lists, can be an object. It adds an item at the end of the list appending it. Extend also adds, but it adds elements from another iterable list.

```
# example of appending - we add an object to the list


my_list = ['an', 'na', 'lew']
print(my_list)
another_list = [1, 2, 3]
print(another_list)
my_list.append(another_list)
print(my_list)
```

```
# example of extending - we concatenate a list with another list (iterable)
my_list = ['foo', 'bar']
print(my_list)
another_list = [1, 2, 3]
print(another_list)
my_list.extend(another_list)
my_list = ['foo', 'bar', 1, 2, 3]
print(my_list)
```

1.7 What does ‘infinite loop’ mean and how to avoid it?

Infinite loop means a loop will go on forever, because it either does not have an exit condition or the exit condition is never met. We can avoid having our loop run indefinitely by 1 – providing an exit condition (such as “if i = 0” or “if you run to the end of the table then stop iterating”) and 2 – making sure that for our data the possibility of this condition becoming true exists.

1.8 What does function ‘parameters’ and function ‘arguments’ mean?

Function parameters are the variables (names of the variables) which the function can use. When declaring a function one should declare the parameters it (the function) can operate on. Parameters are the values these variables are given. These values are then worked by the function to provide an output (some result).

1.9 How to add a new key, value pair to a dictionary, provide examples.

Using either the subscript or the update() method.

```
dict = {'a1': 'nut choco', 'a2': 'snickers', 'a3': 'choco plum'}
print("Currently Possessed Sweets: ", dict)

# adding new key and value by substring

dict['a4'] = 'mars bar'
dict['a5'] = 'gummy bear'
print("Updated Sweets Repo: ", dict)

# adding new key and value using method update()

dict.update({'a6': 'cream fudge'})
print("Even Bigger Sweets Repo: ", dict)
```

1.10 Place these data types into one of the two categories

MUTABLE: list, set, dict

IMMUTABLE: bool, int, tuple, str

2 If you would need to count all of the capital letters in a file. How would you do it?

```
with open('my_file.txt') as fh:
text = fh.read()
big_letters_count = 0
for i in text:
    if i.isupper():
        big_letters_count += 1
print("Upper Letters in Given Text: ", big_letters_count)
```

3 Given any string, reverse it, so that it reads backwards, for example: if the string is “Apple”, you get “'elppA”. You can do this in any way you want.

```
def reverse(text):
  rev = ''
  for i in text:
    rev = i + rev
  return rev


text = 'Apple'

print("Given text: ", text)
print("Reversed text: ", reverse(text))
```

4 Write a function to find the longest word in the list of strings. Given [‘cat’, ‘horse’, ‘elephant’, ‘dog’] your function would return ‘elephant’.

```
def longest(collection):
  longest_word = max(collection, key=len)
  return longest_word


collection = ['cat', 'horse', 'elephant', 'dog']

print("Given text: ", collection)
print("Reversed text: ", longest(collection))
```

5 Write a simple calculator which can perform basic arithmetic operations like addition, subtraction, multiplication or division depending upon the user input.

Answer: below is a calculator that works. It’s the simpliest one I could think of, but it gives all results regardless of choice because I don’t know how to include “break” in Python, I know how to do this in Java. I don’t have time, I have like 15mins left now for SQL because I spend so much time editing this file ;_;

```
# SIMPLE CALCULATOR
print("Please select operation -\n "
"1. Add\n "
"2. Subtract\n"
"3. Multiply\n"
"4. Divide\n")

# Take input from the user
select = int(input("Select operations form 1, 2, 3, 4 :"))
number_1 = int(input("Enter first number: "))
number_2 = int(input("Enter second number: "))

### YOUR CODE GOES HERE ###

# Function to add two numbers
sum_result = number_1 + number_2

# Function to subtract two numbers
minus_result = number_1 - number_2

# Function to multiply two numbers
multiplication_result = number_1 * number_2

# Function to divide two numbers
division_result = number_1 / number_2

# Calculator logic

switcher = {
   1: print(sum_result),
   2: print(minus_result),
   3: print(multiplication_result),
   4: print(division_result)
   }

def switch(switcher):
   return switcher.get(select)()
```

6.1 What is MySQL?

It’s a language made by Oracle to make queries to databases called “Structured Query Language”. It makes questions, calls, and operations on database data and database structure as well.

6.2. Difference between CHAR and VARCHAR?

VARCHAR is longer, CHAR is up to 30 characters while varchar can take 0 to 255. There are also differences in storage and retrieval but that’s as of now a bit beyond my comprehension level.

6.3. What do you mean by % and _ in the LIKE statement?

% is a wildcard – can be any sign or a number of signs, can be at the beginning, in the middle, or at the end of a value. (I have to end this test now…)

4. What is the difference between WHERE and HAVING clause?
5. What does CHECK constraint do, give an example?
6. Name ANY 3 types of JOINs with simple examples (you can draw circle diagrams
or explain?
7. What are COMMIT and ROLLBACK in transaction?
8. What is the difference between these aggregate functions: COUNT() and SUM()?
9. What do you mean by Stored Procedures? How do we use it?
10. What is the difference between clustered and non-clustered indexes?