Variables

1. Naming rules
   1. Syntax: Start with underscore or letter. Cannot start with number
      1. Legal: \_spam, spam, spam\_1
      2. Illegal: 1\_spam, spam$, @#$
   2. Case matters: A and a are different names.
   3. Reserved words are off-limits.
      1. **True | False**
      2. class
      3. finally
      4. None
      5. continue
      6. **for | while**
      7. **def**
      8. from
      9. and
      10. del
      11. global
      12. as
      13. **if | elif | else**
      14. assert
      15. **import**
      16. pass
      17. **break**
      18. in
      19. raise
      20. is
      21. **return**
      22. lambda
      23. **try | excep**t
      24. nonlocal
      25. **while**
      26. not
      27. with
      28. or
      29. yield
      30. **print**
      31. exec | eval

Key Concepts:

1. Polymorphism
   1. Same operator behaving differently with different data types
   2. e.g.
      1. + is addition for numbers but concatenation for strings.
         1. 2 + 2 returns 4
         2. “a” + “b” returns “ab”
      2. \* is multiplication for numbers and repetition for strings.
         1. 2 \* 2 returns 4
         2. “a” \* 2 returns “aa”
   3. Really powerful concept in Python
2. **Immutability**
   1. Data types which cannot be changed in place after they have been created
   2. Every time its value is changed, a new object is created and assigned the same variable name but with a different memory address.
   3. strings, numbers and tuples are immutable.
   4. lists, dictionaries and sets are mutable. They can be changed in place.
3. Dynamic typing
   1. No constraint on data type
   2. Pros and cons of dynamic typing
      1. Re-assign a variable data type unintentionally
      2. Simple and clean - change variable type - flexible and readable
4. Automatic Memory Management/Garbage Collection:
   1. Python has a feature known as garbage collection that cleans up unused memory as your program runs and frees you from having to manage such details in your code.

Object Types in detail:

1. Numbers
   1. integers that have no fractional part, floating-point numbers that do
   2. int(), float()
   3. Supports all mathematical operations: +, -, \*, /, \*\*, %, // among others
   4. Libraries included: math, random
      1. <https://en.wikipedia.org/wiki/Pseudorandom_number_generator>
      2. Statistics
   5. Questions: (Do not use inbuilt function)
      1. Source: https://www.geeksforgeeks.org/python-programming-examples/
      2. Write a function to
         1. add three numbers.
         2. multiply two numbers.
         3. subtract two numbers.
         4. divide first number by second number and return the quotient and remainder separately.
         5. find the square of a number.
         6. find the greater of the two numbers given. | max() conditionals and looping
         7. find the cube of a number.
         8. find the area of a square.
         9. find the area of a circle.
         10. find the simple interest given principal, interest rate and time period.
         11. find the compound interest given principal, interest rate and time period.
         12. print a random integer between two given integers. | random - figure out which function of random module
         13. print 5 random integers between two given integers.
             1. random module
             2. for loop
             3. while loop
             4. loop inside function
             5. loop outside function
         14. check if a number is prime or not. | loop + conditionals + modulus operator
         15. compute 5/0 and use try/except to catch the exceptions. Also, print the exception.
2. String
   1. used to record both textual information (your name, for instance) as well as arbitrary collections of bytes (such as an image file’s contents).
   2. Formally strings are a sequence of one-character strings i.e. a positionally ordered collection of other objects.
   3. Python is 0-indexed language
   4. Sequences maintain a left-to-right order among the items they contain: their items are stored and fetched by their relative positions.
      1. sample\_string = “Welcome to CachePrep! ”
      2. It is an ordered sequence of 1-character strings => “W”, “e”, “l” and so on.
      3. The order is from left to right.
   5. Supports operations such as indexing, slicing, len, concatenation, repetition
      1. Indexing: returns the character present at the given index i.e. position sample\_string[0] returns “W”
      2. Slicing: sample\_string[0:3] returns “Wel”
      3. len(sample\_string) returns length of string
      4. Concatenation: Adding two strings to create a third
         1. sample\_string\_2 = “Let us crack that interview! ”
         2. result = sample\_string + sample\_string\_2
         3. The result holds: “Welcome to CachePrep! Let us crack that interview! ”
      5. Repetition
         1. sample\_string \* 2 returns “Welcome to CachePrep! Welcome to CachePrep! ”
   6. Questions: (Solve these questions without using inbuilt functions/methods)
      1. Write a function to
         1. return 3rd letter of the given string or None if the given string is less than 3 characters. | Index
         2. return odd indexed characters from a given string or None if the string is empty. | Index with Integer operations | for + integer
         3. return the 3rd to 5th character of the given string or None if the string is less than five characters. | Slicing with length and conditionals
         4. return the joined string given two input strings. | Concatenate
         5. return all the words of a given string split on space (“ “) without using the inbuilt method of .split(). | Loop and conditionals
         6. return the number of times “a” appears in the given string.
         7. return the number of times vowels appear in the given string. | list
         8. return the character which appears the most number of times in the given string.
         9. return the number of times each character appears in the string.
         10. return the string after removing the character “i” from the given string.
         11. return the string after removing all vowels from the given string.
         12. return the string converted to title case, lowercase and uppercase.
         13. check if a string is palindrome or not. | interview question - advanced
         14. print the company name of a given email address where the email addresses are in the "username@companyname.com" format and are composed of only alphabets.
         15. convert snake case to pascal case. | cache\_prep\_interview\_preparation to CachePrepInterviewPreparation
3. Booleans
   1. True | False
   2. Essentially just the integers 1 and 0 with custom display logic
   3. Any data type in python that is empty is treated as a False value. eg.: [], {}, “”
   4. None: special placeholder object commonly used to initialize names and objects
      1. sample\_object = None
4. Lists
   1. Positional and ordered collection
   2. Same list can hold python objects of different types
      1. sample\_list = [1, “a”, [1, “a”]]
         1. It has a number at position 1 which has index 0, alphabet “a” at position 2 which is index 1 and list [1, “a”] at position 3 which is index 2.
   3. Mutable i.e. they can be modified in place such as new elements can be added or existing elements can be removed.
   4. Similar to arrays in other languages
   5. Stack - extends to the last and pops from the end
   6. Some available methods:
      1. .append(new\_element)
      2. .insert - to add new item at an arbitrary position
      3. .remove() - to remove a given item by value
      4. .extend() - to add multiple items at the end
      5. .pop() to remove the last element
      6. .reverse() to reverse the order of elements
      7. .sort()
      8. enumerate() function
      9. Looping using: for element in sample\_list
      10. Indexing - same as string
      11. Slicing - same as string
   7. List comprehension expression
      1. col2 = [row[1] for row in M]
      2. They are a way to build a new list by running an expression on each item in a sequence, one at a time, from left to right.
      3. Enclosing a comprehension in parentheses can also be used to create generators that produce results on demand.
   8. Questions: Write a function to:
      1. sum all the elements of a list.
      2. find the largest element in the given list.
      3. to find the second largest number in a list.
      4. find the smallest element in the given list.
      5. to interchange first and last elements in a list.
      6. Python program to swap 3rd and 5th elements of a list else return None if the list has less than 3 elements.
      7. to find the length of the given list without using the len() function.
      8. to check if an element exists in the given list.
      9. print the elements of the list in the reverse order.
      10. find the product of all the numbers in the list.
      11. to print even numbers in a list
      12. to print odd numbers in a list
      13. to print values present at even index in the list (0, 2, 4 etc)
      14. to print values present at odd index in the list (0, 2, 4 etc)
      15. to print positive numbers in a list
      16. to print negative numbers in a list
      17. to print all positive numbers in a range
      18. to print all negative numbers in a range
      19. to print all the words in uppercase in a given list.
      20. to find which elements present in the first list are not present in the second list.
      21. to print the number of times each element occurs in the list.
5. Dictionaries
   1. They are a mapping of key-value pairs. Unlike lists, they are not positional or have any concept of order or index.
   2. Dictionaries, the only mapping type in Python’s core objects set, are also mutable: like lists, they may be changed in place and can grow and shrink on demand.
   3. Some available methods:
      1. .keys()
      2. .items()
      3. .values()
      4. .get()
      5. Indexing using square brackets - []
   4. Comprehension expression
6. Sets
   1. Unordered collections of unique and immutable objects
   2. Sets are much like the keys of a valueless dictionary
7. Tuples
   1. Immutable sequence
   2. Like list but cannot be modified in place (immutable)
   3. Coded inside parentheses (1, “a”, [1, “a”])
   4. A tuple with single item requires a trailing comma. (1,)
   5. Tuples provide a sort of data integrity constraint. Any attempt to change value or assign elements within it to a different value throws an error.
      1. TypeError: 'tuple' object does not support item assignment
8. Files
   1. to create a file object, you call the built-in open function, passing in an external filename and an optional processing mode as strings.
   2. open | close | with
   3. f = open(‘sample\_file.txt’, ‘w’)
   4. r = read, w = write, rb = read binary, wb = write binary
9. Binary Bytes Files
   1. Python’s struct module can both create and unpack packed binary data