PYTHON TEST - 1.7 (MUTABLE AND IMMUTABLE TYPES)

Total points 50/50

Mutable and Immutable

STUDENT NAME *

VIVA

✓ 1. Which of the following is **mutable** in Python? * 1/1

- a) str
- b) tuple
- c) list ✓
- d) int

✓ 2. Which of the following is **immutable**? * 1/1

- a) set
- b) list
- c) dict
- d) tuple ✓

✓ 3. Strings in Python are: * 1/1

- ○ a) mutable
- ◉ b) immutable ✓
- ○ c) constant only
- ○ d) None of these

✓ 4. Which of the following data types are immutable? * 1/1

- ○ a) tuple
- ○ b) int
- ○ c) str
- ◉ d) All of these ✓

✓ 5. Which of the following is mutable? * 1/1

- ○ a) dict
- ○ b) list
- ○ c) set
- ◉ d) All of these ✓

✓ 6. What happens if you try to modify a string in Python? * 1/1

- a) It changes directly
- b) It throws an error ✓
- c) It returns None
- d) It becomes a list

✓ 7. Which of the following is mutable? * 1/1

- a) bytes
- b) bytearray ✓
- c) frozenset
- d) tuple

✓ 8. Which is immutable? * 1/1

- a) frozenset ✓
- b) set
- c) dict
- d) list

✓ 9. What is the mutability of a dictionary? * 1/1

- (●) a) Mutable ✓
- ( ) b) Immutable
- ( ) c) Depends on keys
- ( ) d) None

✓ 10. What is the mutability of an integer in Python? * 1/1

- ( ) a) Mutable
- (●) b) Immutable ✓

✓ 11. Why are strings immutable? * 1/1

- ( ) a) For memory efficiency
- ( ) b) For security reasons
- ( ) c) To allow string interning
- (●) d) All of the above ✓

✓ 12. What is the result of: * 1/1

s = "hello"

s[0] = "H"

- a) "Hello"
- b) "hello"
- c) Error ✓
- d) "hEllo"

✓ 13. Tuples are immutable because: * 1/1

- a) They are stored as read-only
- b) They prevent accidental modification
- c) They improve performance
- d) All of the above ✓

✓ 14. Which of these supports item reassignment? * 1/1

- a) str
- b) tuple
- c) list ✓
- d) int

✓ 15. Which is correct about tuples? * 1/1

- a) They can contain mutable objects
- b) They themselves cannot be changed
- c) Both (a) and (b) ✓
- d) None

✓ 16. A tuple inside a tuple is still: * 1/1

- a) Mutable
- b) Immutable ✓

✓ 17. A tuple containing a list is: * 1/1

- a) Fully immutable
- b) Partially mutable ✓
- c) Mutable

✓ 18. "abc" + "def" creates: * 1/1

- a) A new string object ✓
- b) Modifies original string
- c) Error

✓ 19. "python".replace("p", "P") results in: * 1/1

- (●) a) "Python" (new string created) ✓
- ( ) b) Modifies original string
- ( ) c) Error

✓ 20. Immutable objects in Python: * 1/1

- (●) a) Cannot be modified after creation ✓
- ( ) b) Can be changed with append()
- ( ) c) Are always faster
- ( ) d) Cannot be deleted

✓ 21. Which of the following is mutable? * 1/1

- ( ) a) list
- ( ) b) dict
- ( ) c) set
- (●) d) All of the above ✓

✓ 22. Lists are mutable because: * 1/1

- ( ) a) They allow appending and removing items
- ( ) b) They allow item assignment
- ( ) c) They change without creating new objects
- (●) d) All of the above ✓

✓ 23. What is the result of: * 1/1

```
a = [1,2,3]

b = a

b[0] = 10

print(a)
```

- ○ a) [1,2,3]
- ◉ b) [10,2,3] ✓
- ○ c) Error

✓ 24. Sets are: * 1/1

- ◉ a) Mutable ✓
- ○ b) Immutable

✓ 25. Frozensets are: * 1/1

- ○ a) Mutable
- ◉ b) Immutable ✓

✓ 26. Dict keys must be: * 1/1

- ○ a) Mutable
- ◉ b) Immutable ✓
- ○ c) Either

✓ 27. Dict values can be: * 1/1

- a) Only immutable
- b) Only mutable
- c) Any data type ✓

✓ 28. Adding new elements to a dictionary is possible because it is: * 1/1

- a) Mutable ✓
- b) Immutable

✓ 29. Which of the following is a mutable mapping type? * 1/1

- a) dict ✓
- b) list
- c) set
- d) tuple

✓ 30. Which is immutable? * 1/1

- a) set
- b) frozenset ✓
- c) dict
- d) list

✓ 31. Immutable objects are stored in: * 1/1

- (●) a) Constant memory locations ✓
- ( ) b) Variable memory locations

✓ 32. Mutable objects share references because: * 1/1

- (●) a) They can change in place ✓
- ( ) b) They are copied always
- ( ) c) They are immutable

✓ 33. When you modify a list, Python: * 1/1

- ( ) a) Creates a new object
- (●) b) Modifies the existing object ✓

✓ 34. When you modify a string, Python: * 1/1

- (●) a) Creates a new object ✓
- ( ) b) Modifies the existing object

✓ 35. Which type is hashable? * 1/1

- a) list
- b) dict
- c) tuple ✓
- d) set

✓ 36. Hashable objects must be: * 1/1

- a) Mutable
- b) Immutable ✓

✓ 37. Can a list be used as a dictionary key? * 1/1

- a) Yes
- b) No ✓

✓ 38. Can a tuple be used as a dictionary key? * 1/1

- a) Yes ✓
- b) No

✓ 39. Why can't lists be dictionary keys? * 1/1

- (●) a) They are mutable ✓
- ( ) b) They are too big
- ( ) c) They are unordered

✓ 40. Immutable types in Python are generally: * 1/1

- (●) a) Faster ✓
- ( ) b) Slower

✓ 41. Which of the following are immutable types? * 1/1

- (●) a) int, float, str, tuple ✓
- ( ) b) list, set, dict
- ( ) c) list, tuple, str
- ( ) d) dict, int, set

✓ 42. Which of the following are mutable types? * 1/1

- (●) a) list, dict, set, bytearray ✓
- ( ) b) str, tuple, frozenset
- ( ) c) int, float, str
- ( ) d) tuple, dict, str

✓ 43. What happens when two variables reference the same list and one is changed? *1/1

- (●) a) Both see the change ✓
- ( ) b) Only one changes
- ( ) c) Error

✓ 44. Immutable objects are safe to use as: * 1/1

- ( ) a) List elements
- ( ) b) Dictionary keys
- ( ) c) Set elements
- (●) d) All of the above ✓

✓ 45. Mutable default arguments in functions can cause: * 1/1

- ( ) a) Errors
- (●) b) Unexpected behavior ✓
- ( ) c) Faster execution
- ( ) d) Nothing

✓ 46. The id() of an immutable object changes when modified. * 1/1

- ( ) a) True
- (●) b) False ✓

✓ 47. The id() of a mutable object changes when modified. * 1/1

- ○ a) True
- ◉ b) False ✓

✓ 48. Which of these will create a new object? * 1/1

- ◉ a) x = "hello"; x += "world" ✓
- ○ b) x = [1,2]; x.append(3)

✓ 49. Which of these modifies in place? * 1/1

- ◉ a) list.append() ✓
- ○ b) str.replace()

✓ 50. Which statement is correct? * 1/1

- ◉ a) All immutable objects are hashable ✓
- ○ b) All mutable objects are hashable
- ○ c) Only mutable objects can be dict keys
- ○ d) Only lists are immutable