PYTHON TEST - 2.6 (SORTING TECHNIQUES)

Total points 50/50

Sorting Techniques in Python

STUDENT NAME *

VIVA

✓ 1. Which Python function is used to sort lists? * 1/1

- a) sorted()
- b) sort()
- [✓ c) Both a and b] ✓
- d) None

✓ 2. What is the difference between sort() and sorted() in Python? * 1/1

- ○ a) sort() returns a new list; sorted() sorts in place
- ● b) sort() sorts in place; sorted() returns a new list ✓
- ○ c) Both return a new list
- ○ d) Both sort in place

✓ 3. By default, sorting in Python is: * 1/1

- ○ a) Descending order
- ○ b) Lexicographical order
- ● c) Ascending order ✓
- ○ d) Random order

✓ 4. Which argument is used to reverse the sort order? * 1/1

- ○ a) order=reverse
- ● b) reverse=True ✓
- ○ c) desc=True
- ○ d) sort="desc"

✓ 5. Which algorithm is used internally by Python’s sort()? * 1/1

- a) Quick Sort
- b) Merge Sort
- [✓ c) Timsort] ✓
- d) Heap Sort

✓ 6. What will be the output of: * 1/1

```
x = [3, 1, 4, 2]

x.sort()

print(x)
```

- a) [3, 1, 4, 2]
- [✓ b) [1, 2, 3, 4]] ✓
- c) [4, 3, 2, 1]
- d) Error

✓ 7. What is the output? * 1/1

```
print(sorted("python"))
```

- a) ['p','y','t','h','o','n']
- [✓ b) ['h','n','o','p','t','y']] ✓
- c) ['y','t','p','o','n','h']
- d) Error

✓ 8. Which data types can be sorted using sorted()? * 1/1

- ○ a) List
- ○ b) Tuple
- ○ c) String
- [● d) All of the above] ✓

✓ 9. Which method is valid to sort a tuple? * 1/1

- ○ a) tuple.sort()
- [● b) sorted(tuple)] ✓
- ○ c) tuple.sorted()
- ○ d) Not possible

✓ 10. What does this return? * 1/1
sorted([1,2,3], reverse=True)

- ○ a) [1,2,3]
- [● b) [3,2,1]] ✓
- ○ c) [2,3,1]
- ○ d) Error

✓ 11. Which argument allows custom sorting in sorted()? * 1/1

- a) func
- [✓ b) key] ✓
- c) sorter
- d) custom

✓ 12. Output? * 1/1

```
words = ["banana","apple","cherry"]
print(sorted(words, key=len))
```

- a) ['banana','apple','cherry']
- [✓ b) ['apple','cherry','banana']] ✓
- c) ['cherry','banana','apple']
- d) Error

✓ 13. Sorting numbers by absolute value uses: * 1/1
sorted([-2,1,-3], key=abs)

- [✓ a) [1, -2, -3]] ✓
- b) [-2, -3, 1]
- c) [1, -2, -3]
- d) [1, -2, -3]

✓ 14. Which lambda function sorts list of tuples by second element? * 1/1

- ○ a) key=lambda x: x[0]
- [✓ ● b) key=lambda x: x[1]] ✓
- ○ c) key=lambda x: x
- ○ d) None

✓ 15. Which is stable sorting? * 1/1

- [✓ ● a) Sorting where order of equal elements is preserved] ✓
- ○ b) Sorting where order is reversed
- ○ c) Sorting with O(n^2) complexity
- ○ d) None

✓ 16. Output? * 1/1
sorted("hello")

- ○ a) ['h','e','l','l','o']
- [✓ ● b) ['e','h','l','l','o']] ✓
- ○ c) ['o','l','l','h','e']
- ○ d) Error

✓ 17. Sorting string list in descending order: sorted(["cat","dog","bat"], reverse=True) * 1/1

- a) ['cat','dog','bat']
- [✓ b) ['dog','cat','bat']] ✓
- c) ['dog','bat','cat']
- d) ['bat','cat','dog']

✓ 18. Which function converts sorted characters back to string? * 1/1

- [✓ a) join()] ✓
- b) append()
- c) str()
- d) concat()

✓ 19. Output? "".join(sorted("python")) * 1/1

- [✓ a) "hnopty"] ✓
- b) "python"
- c) "typhon"
- d) Error

✓ 20. Sorting is case-sensitive by default. * 1/1

[✓ a) True] ✓

b) False

✓ 21. Output? sorted(["Apple","banana","Cherry"], key=str.lower) * 1/1

a) ['Apple','banana','Cherry']

b) ['Apple','Cherry','banana']

[✓ c) ['banana','Apple','Cherry']] ✓

d) Error

✓ 22. Which sorts dictionary by keys? sorted({3:"c",1:"a",2:"b"}) * 1/1

[✓ a) [1,2,3]] ✓

b) ['a','b','c']

c) [(1,'a'),(2,'b'),(3,'c')]

d) Error

✓ 23. How do you sort dictionary by values? * 1/1

- ○ a) sorted(dict.values())
- ● b) sorted(dict.items(), key=lambda x:x[1]) ✓
- ○ c) dict.sort()
- ○ d) dict.sorted()

✓ 24. What is the output? * 1/1

```
nums = [5,2,9]

nums.sort()

print(nums)
```

- ● a) [2,5,9] ✓
- ○ b) [9,5,2]
- ○ c) [5,2,9]
- ○ d) Error

✓ 25. Which sorts list of numbers as strings? * 1/1
sorted([1,100,12], key=str)

- ○ a) [1,12,100]
- ● b) [100,12,1] ✓
- ○ c) [1,100,12]
- ○ d) Error

✓ 26. Which is fastest built-in sorting in Python? * 1/1

- a) Quick Sort
- b) Merge Sort
- [✓ c) Timsort] ✓
- d) Heap Sort

✓ 27. Time complexity of Timsort (average case)? * 1/1

- a) O(n^2)
- [✓ b) O(n log n)] ✓
- c) O(log n)
- d) O(n)

✓ 28. Space complexity of Timsort? * 1/1

- a) O(1)
- [✓ b) O(n)] ✓
- c) O(n log n)
- d) O(n^2)

✓ 29. Which sorting algorithm is stable? * 1/1

- a) Selection Sort
- b) Insertion Sort
- c) Merge Sort
- [✓ d) All of the above] ✓

✓ 30. Is Quick Sort stable in Python? * 1/1

- a) Yes
- [✓ b) No] ✓

✓ 31. Output? * 1/1

nums = [10,2,33] print(sorted(nums, key=lambda x: str(x)))

- a) [2,10,33]
- b) [10,2,33]
- [✓ c) [10,33,2]] ✓
- d) Error

✓ 32. Sort list in place in descending order: * 1/1

○ a) list.sort(reverse=True)

○ b) sorted(list, reverse=True)

● c) Both a and b ✓

○ d) None

✓ 33. Sorting list of sets: * 1/1
sorted([{1,2},{3},{0}], key=len)

○ a) [{3},{1,2},{0}]

● b) [{0},{3},{1,2}] ✓

○ c) [{3},{0},{1,2}]

○ d) Error

✓ 34. Output? * 1/1
names = ["Tom","anna","BOB"] print(sorted(names, key=str.lower))

○ a) ['Tom','anna','BOB']

● b) ['anna','BOB','Tom'] ✓

○ c) ['BOB','Tom','anna']

○ d) Error

✓ 35. Sorting custom objects requires: * 1/1

- a) __cmp__ method
- [✓ b) key function or __lt__ method] ✓
- c) compare() function
- d) None

✓ 36. Which function reverses a list without sorting? * 1/1

- a) list.sort(reverse=True)
- [✓ b) list[::-1]] ✓
- c) reversed(list)
- d) Both b and c

✓ 37. Sorting a generator is possible using: * 1/1

- a) sort()
- [✓ b) sorted()] ✓
- c) Both
- d) None

✓ 38. Output? * 1/1
sorted("Sorting123", key=str.isdigit)

- a) Letters first, then digits
- [✓ b) Digits first, then letters] ✓
- c) Random
- d) Error

✓ 39. Which returns an iterator instead of list? * 1/1

- a) sorted()
- b) sort()
- [✓ c) reversed()] ✓
- d) None

✓ 40. Which method sorts only lists? * 1/1

- [✓ a) sort()] ✓
- b) sorted()
- c) Both
- d) None

✓ 41. Sorting with key = str.lower makes it case-insensitive. * 1/1

[✓ a) True] ✓

b) False

✓ 42. Which function gives indices of sorted order? * 1/1

[✓ a) argsort() in NumPy] ✓

b) sorted()

c) sort()

d) None

✓ 43. Output? *
sorted([True, False, True]) 1/1

a) [True, False, True]

[✓ b) [False, True, True]] ✓

c) [True, True, False]

d) Error

✓ 44. Sorting dictionary by key length: sorted({"one":1,"three":3,"two":2}, key=len) * 1/1

- [✓ a) ['one','two','three']] ✓
- b) ['three','two','one']
- c) ['one','three','two']
- d) Error

✓ 45. Sorting complex numbers works by: * 1/1

- a) Default
- [✓ b) Giving key (abs)] ✓
- c) Not possible
- d) Both a and b

✓ 46. Sorting None values in list raises error. * 1/1

- [✓ a) True] ✓
- b) False

✓ 47. Output? * 1/1
sorted(["a10","a2","a1"], key=lambda x:int(x[1:]))

[✓ ['a1','a2','a10']] ✓

b) ['a10','a2','a1']

c) ['a2','a1','a10']

d) Error

✓ 48. Sorting integers vs strings comparison is allowed. * 1/1

a) True

[✓ b) False] ✓

✓ 49. Sorting function with key must return: * 1/1

a) Boolean

b) Integer

[✓ c) Comparable value] ✓

d) Any type

✓ 50. Which Python module provides advanced sorting for large arrays? * 1/1

- ○ a) math
- [✓ ● b) numpy] ✓
- ○ c) random
- ○ d) statistics