Marks: 100

TN SKILL DEVELOPMENT CORPORATION

B2025_034_4542-3776 – PYTHON DEVELOPER

MODULE I EXAM

ONE MARKS **(20*1 = 20)**

1. What is the output of this code?

```
x = 5
y = 2
print(x // y + x % y)
```

A. 2.5

B. 3

C. 2

D. 3.5

2. Which method is used to add an element to a list?

A) append()

B) add()

C) insert()

D) extend()

3. What does this code print?

```
text = "hello"
print(text[1:4])
```

A) "hel"

B) "ell"

C) "ello"

D) "hell"

4. How do you create a dictionary?

A) {key: value}

B) (key, value)

C) [key: value]

D) <key: value>

5. What is the output?

```
numbers = [1, 2, 3, 4, 5]
print(numbers[-2])
```

A) 1

B) 4

C) 5

D) Error

6. Which loop is used to iterate over a sequence?

A) for loop

B) while loop

C) do-while loop

D) repeat loop

7. What does this function return?

```
def multiply(a, b=2):
    return a * b
print(multiply(3))
```

A) 6

B) 5

C) Error

D) None

8. How do you read user input?

A) input()
B) read()
C) get_input()
D) scan()

9. What is the output?

python

```python
x = 10
if x > 5:
    print("A")
elif x > 8:
    print("B")
else:
    print("C")
```

A) A
B) B
C) C
D) A B

10. Which operator checks if two values are equal?

A) =
B) ==
C) ===
D) !=

11. What is the output?

```python
def modify_list(lst):
    lst.append(4)
```

```
        return lst

my_list = [1, 2, 3]
new_list = modify_list(my_list)
print(my_list)
```

A) [1, 2, 3]
B) [1, 2, 3, 4]
C) [4]
D) Error

12. What does list comprehension do?

```
squares = [x**2 for x in range(5)]
```

A) Creates [0, 1, 4, 9, 16]
B) Creates [1, 4, 9, 16, 25]
C) Creates [0, 1, 2, 3, 4]
D) Error

13. What is the output?

```
a = [1, 2, 3]
b = a
b[0] = 10
print(a)
```

A) [1, 2, 3]
B) [10, 2, 3]
C) [10, 10, 10]
D) Error

14. Which method is used to handle exceptions?

A) try-except

B) catch-throw

C) error-handle

D) try-catch

15. What does this code do?

```
with open('file.txt', 'r') as f:
    content = f.read()
```

A) Writes to a file

B) Reads from a file and automatically closes it

C) Appends to a file

D) Deletes a file

16. What is the output?

```
def generator():
    for i in range(3):
        yield i


g = generator()
print(list(g))
```

A) [0, 1, 2]

B) 0 1 2

C) <generator object>

D) Error

17. Which keyword is used to define a function in Python?

A) function

B) def

C) define

D) func

18. What does @staticmethod do?

A) Creates a method that doesn't receive self

B) Creates a private method

C) Creates a class method

D) Creates an abstract method

19. What is the output?

```
from collections import Counter
words = ['apple', 'banana', 'apple', 'orange', 'banana', 'apple']
count = Counter(words)
print(count.most_common(1))
```

A) [('apple', 3)]

B) {'apple': 3}

C) 3

D) ['apple']

20. What does this lambda function do?

```
lambda x: x % 2 == 0
```

A) Squares a number

B) Checks if number is even

C) Doubles a number

D) Checks if number is odd

SHORT ANSWERS: (Any Ten) **10*3 = 30**

1. Explain the difference between is and == operators in Python with suitable examples.

2. What are mutable and immutable objects in Python? Give two examples of each and explain why understanding this distinction is important.

3. Explain the concept of "interning" in Python. How does it affect the behavior of integers and strings?

4. What is the difference between *args and **kwargs in Python function definitions? Provide a scenario where you would use each.

5. Differentiate between class variables and instance variables in Python. When would you use each?

6. What are Python decorators? Explain how they work with a simple example.

7. Explain the concept of abstract base classes (ABCs) in Python. What problem do they solve?

8. Differentiate between syntax errors and exceptions in Python. Give examples of each.

9. Compare lists and tuples in Python. When would you prefer tuples over lists?

10. Explain Python's exception handling mechanism. What is the purpose of else and finally clauses?

11. Explain the difference between lists and tuples in Python with examples.

12 What are list comprehensions? Provide examples.

5 MARK (Any Four) **4*5 = 20**

1. Simple Bank Account System

```
class BankAccount:
    def __init__(self, account_holder, balance=0):
        self.account_holder = account_holder
        self.balance = balance
    def deposit(self, amount):
        # Add amount to balance
        pass

    def withdraw(self, amount):
        # Subtract amount from balance if sufficient funds
        pass

    def check_balance(self):
        # Return current balance
        pass

    def account_info(self):
        # Return account holder and balance
        pass

# Test
account = BankAccount("Alice", 1000)
account.deposit(500)
account.withdraw(200)
print(account.check_balance())
print(account.account_info())
```

2. Polymorphism with Inheritance

Create a class Employee with a method get_details().

1. Subclass Manager overrides get_details() to include department.
2. Subclass Developer overrides get_details() to include programming language.
3. Call get_details() on a list of mixed objects.

3. Simple Shopping Cart System

python

```
class ShoppingCart:
    def __init__(self):
        self.items = []

    def add_item(self, item_name, price, quantity=1):
        # Add item to cart
        pass

    def remove_item(self, item_name):
        # Remove item from cart
        pass

    def calculate_total(self):
        # Calculate total price of all items
        pass

    def show_cart(self):
        # Display all items in cart
        pass

# Test
cart = ShoppingCart()
cart.add_item("Apple", 1.5, 3)
cart.add_item("Banana", 0.5, 6)
print(cart.calculate_total())
cart.show_cart()
```

4. Dictionary & String Manipulation

Write a Python program that:

1. Accepts a sentence from the user.
2. Counts the frequency of each word in the sentence.
3. Stores the results in a dictionary and prints it.

Example:

text

```
Input: "python is easy and python is fun"
Output: {'python': 2, 'is': 2, 'easy': 1, 'and': 1, 'fun': 1}
```

5. Number Guessing Game

python

```
import random

class NumberGame:
    def __init__(self):
        self.target_number = 0
        self.attempts = 0

    def start_game(self, min_num=1, max_num=100):
        # Set random target number between min and max
        pass

    def guess(self, number):
        # Check if guess is correct, too high, or too low
        pass

    def get_attempts(self):
        # Return number of attempts
        pass

# Test
game = NumberGame()
game.start_game(1, 50)
print(game.guess(25))
print(game.guess(40))
print(f"Attempts: {game.get_attempts()}")
```

6. Exception Handling & User Input

Write a Python program that:

1. Continuously asks the user to input two numbers until valid numbers are provided.
2. Divides the first number by the second number.
3. Handles the following exceptions:
   `ZeroDivisionError` → print "Cannot divide by zero. Try again."
   `ValueError` → print "Invalid input, please enter numeric values."
   `KeyboardInterrupt` → print "Program interrupted by user" and exit gracefully.
4. After a successful division, prints the result formatted to 2 decimal places.
5. Asks the user if they want to perform another division (y/n). If y, repeat; if n, exit.

10 Marks(Any 3) **3 * 10 = 30**

1. File Handling & Text Processing

Write a Python program that:

1. Reads a text file named input.txt.
2. Counts the number of lines, words, and characters in the file.
3. Writes the output in a new file output.txt in the following format:

    text

```
Number of lines: X
Number of words: Y
Number of characters: Z
```

2. STUDENT MANAGEMENT SYSTEM

Problem: Create a complete student management system with file handling.

```
"""
Create a program that:
1. Allows adding new students (roll_no, name, marks)
2. Saves data to a file 'students.txt'
3. Can read and display all students
4. Can search student by roll number
5. Can update student marks
6. Can delete student record
7. Calculate class average
8. Generate report of students with marks > 75%
"""


# Your implementation here
def add_student():
    pass
```

```
def view_all_students():
    pass

def search_student():
    pass

def update_marks():
    pass

def delete_student():
    pass

def class_average():
    pass

def top_performers():
    pass

# Main menu driven program
def main():
    pass

if __name__ == "__main__":
    main()
```

3 CONTACT MANAGEMENT SYSTEM

Problem Statement:

Create a simple contact book to store and manage contacts.

Requirements:

1. Add contact (name, phone, email)
2. View all contacts
3. Search contact by name
4. Update contact
5. Delete contact
6. Count total contacts
7. Check if contact exists
8. Show contacts alphabetically
9. Save contacts to file
10. Load contacts from file

Expected Output:

text

```
=== CONTACT BOOK ===
1. Add Contact
2. View All Contacts
3. Search Contact
4. Update Contact
5. Delete Contact
6. Total Contacts
7. Exit

Enter choice: 1
Enter name: Alice
Enter phone: 9876543210
Enter email: alice@email.com
Contact added successfully!
```

5 . TO-DO LIST APPLICATION

Problem Statement:

Create a simple to-do list manager.

Requirements:

1. Add task
2. View all tasks
3. Mark task as completed
4. Delete task
5. Show pending tasks
6. Show completed tasks
7. Count tasks
8. Clear all tasks
9. Save tasks to file
10. Set task priority (High/Medium/Low)

Expected Output:

text

```
=== TO-DO LIST ===
1. Add Task
2. View All Tasks
3. Mark Completed
4. Delete Task
5. Pending Tasks
6. Completed Tasks
7. Total Tasks
8. Exit

Enter choice: 1
Enter task: Complete Python project
```

```
Priority (H/M/L): H
Task added!
```

5 TEMPERATURE CONVERTER

Problem Statement:

Create a temperature conversion tool with multiple features.

Requirements:

1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
3. Celsius to Kelvin
4. Kelvin to Celsius
5. Show conversion table for a range
6. Store conversion history
7. Most frequent conversion type
8. Clear history
9. Save conversions to file
10. Temperature category (Freezing/Cold/Warm/Hot)

Expected Output:

text

```
=== TEMPERATURE CONVERTER ===
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
3. Show Conversion Table
4. Conversion History
5. Exit

Enter choice: 1
Enter temperature in Celsius: 25
```

25°C = 77°F (Warm)