Python-PrepTerm
Quiz

Code:	MT2020132

1. Analyze the code:

```python
print("Recursive Function")
def factorial(n):
    return(n*factorial(n-1))
factorial(4)
```

   1. Recursive Function 24.

   2. Recursive Function.

   3. Function runs infinitely and causes a StackOverflowError.

   4. Syntax Error.

2. What is the output of the following code?

```python
eval("1 + 3 * 2")
```

   1. 1+6

   2. 4∗2

   3. 1+3∗2

   4. 7

3. nfig () in Python Tkinter are used for

   1. destroy the widget

   2. place the widget

   3. change property of the widget

   4. configure the widget

4. What is the following function compares elements of both dictionaries dict1, dict2?

   1. dict1.**cmp**(dict2)

   2. dict1.sort(dict2)

   3. **cmp**(dict1, dict2)

   4. None of the above.

5. What is output for **min**("hello world")

1. e
2. a blank space character
3. w
4. None of the above.

6. Using the pack manager, how you can you put the components in a container in the same row?

   1. Component.pack(side= ''LEFT'')
   2. Component.pack(''Left '')
   3. Component.pack(side=LEFT)
   4. Component.pack(Left−side)

7. What is the output of the code?

```python
def f():
    try:
        return(1)
    finally:
        return(2)
k=f()
print(k)
```

   1. 1 2
   2. 2 1
   3. 2
   4. Error

8. There are different basic operators in python and work according to the order of their precedence.

   Arrange the order of precedence of the following operators:

   1. Division
   2. Multiplication
   3. Parentheses
   4. Exponential
   5. Addition
   6. Subtraction

   1. i, ii, iii, iv, v, vi.
   2. iv, iii, ii, i, vi, v.
   3. iii, iv, i, ii, v, vi.
   4. iv, iii, i, ii, v, vi.

9. What is the output of **print** tinylist ∗ 2 **if** tinylist = [123, 'john']?

   1. [123, 'john', 123, 'john']\ lstinline

2. [123, 'john'] * 2\ lstinline

3. Error

4. None of the above.

10. rrect way to draw a line in canvas tkinter?

    1. line ()

    2. canvas. create_line ()

    3. create_line (canvas)

    4. None of the above

11. Essential thing to create a window screen using tkinter Python?

    1. call tk() function

    2. create a button

    3. To define a geometry

    4. All of the above

12. How many except statements can a try-except block have?

    1. zero

    2. one

    3. more than one

    4. more than zero

13. Which of the following operator in python evaluates to true if the variables on either side of the operator point to the same object and false otherwise?

    1. **

    2. //

    3. **is**

    4. **not in**

14. What is output of following code:

```
num=3
while True:
    if (num%0o12 == 0):
        break
print(num)
num += 1
```

    1. 3 4 5 6 7 8 9 10 11 12

    2. 3 4 5 6 7 8 9

    3. 3 4 5 6 7 8 9 10 11

    4. None of the above

15. What will be the output of the below given code?

```
colors = ["white", "Black", "Grey"]
x = "Red" not in colors
```

   1. Yes
   2. No
   3. Error: not in not defined
   4. True

16. What happens in the below code?

```
class A:
    def __init__(self, i=100):
        self.i=i
class B(A):
    def __init__(self,j=0):
        self.j=j
def main():
    b= B()
    print(b.i)
    print(b.j)
main()
```

   1. Class B inherits all the data fields of class A.
   2. Class B needs an Argument.
   3. The data field 'j' cannot be accessed by object b.
   4. Class B is inheriting class A but the data field 'i' in A cannot be inherited.

17. Which of the following environment variable for Python contains the path of an initialization file containing Python source code?

   1. PYTHONPATH
   2. PYTHONSTARTUP
   3. PYTHONCASEOK
   4. PYTHONHOME

18. What is the output of **print str ∗ 2 if str = 'Hello World!'**?

   1. Hello World!Hello World!
   2. Hello World! ∗ 2
   3. Hello World!
   4. None of the above.

19. For tuples and list which is correct?

   1. List and tuples both are mutable.
   2. List is mutable whereas tuples are immutable.

3. List and tuples both are immutable.

4. List is immutable whereas tuples are mutable.

20. Is the following Python code valid?

```python
try:
    # Do something
except:
    # Do something
finally:
    # Do something
```

1. no, there is no such thing as finally

2. no, finally cannot be used with except

3. no, finally must come before except

4. yes