Practice Module 2

Question 1: What is the output of the following code?

```
>>> try:
>>>     raise IOError
>>> except IOError:
>>>     raise RuntimeError from None
```

A. IOError

B. SyntaxError: invalid syntax

C. RuntimeError

D. No output

Question 2: What is the output of the following code?

```
>>> def this_fails():
>>>     x = 1/0
>>> try:
>>>     this_fails()
>>> except ZeroDivisionError:
>>>     pass
```

A. SyntaxError: invalid syntax

B. ZeroDivisionError: division by zero

C. NameError: name “ZeroDivisionError” is not defined

D. No ouput

Question 3: How will an exception handler be searched if an exception occurs? (Select two answers)

A. Only the first matched exception handler will be executed.

B. The closest exception handler based on __mro__ will be executed.

C. The expression-less except will be executed if no match was found.

D. Only the last matched exception handler will be executed.

Question 4: What is the output of the following code?

```
>>> try:
>>>     abcd
>>>     efgh
except:
>>>     pass
```

A. SyntaxError: invalid syntax

B. No output

C. NameError: name ‘UndefineException’ is not defined

D. Add () on Line 2 and 3 to fix the syntax error

Question 5: Which option(s) will print ELSE given the following code?

```
>>> try:
>>>     <<< INSERT CODE HERE >>>
>>> except ZeroDivisionError:
>>>     print('ZeroDivisionError')
>>> except TypeError:
>>>     print('TypeError')
>>> else:
>>>     print(‘ELSE')
```

A. replace <<< INSERT CODE HERE >>> with blank

B. raise Exception

C. pass

D. raise ZeroDivisionError

Question 6: What is the output of the following code?

```
>>> def f():
>>>     try:
>>>         raise ArithmeticError
>>>     except:
>>>         raise AssertionError
>>>     finally:
>>>         raise AttributeError
>>>     return
>>> f()
```

A. AssertionError

B. SyntaxError: invalid syntax

C. ArithmeticError

D. AttributeError

E. No output

Question 7: What is the output of the following code?

```
>>> try:
>>>     raise OSError
>>> finally:
>>>     pass
```

A. SyntaxError: invalid syntax

B. NameError: name ‘OSError’ is not defined

C. No output

D. OSError

Question 8: What is the output of the following code?

```
>>> try:
>>>     print("1", end='')
>>>     raise Exception
>>>     print("2", end='')
>>> except BaseException:
>>>     print("3", end='')
>>> else:
>>>     print("4", end='')
>>> finally:
>>>     print(“5")
```

A. 145

B. 135

C. 1245

D. 1235

Question 9: What is the output of the following code?

```
>>> def f():
>>>     try:
>>>         print('try ', end='')
>>>         raise ArithmeticError
>>>     except:
>>>         print('except ', end='')
>>>         raise AssertionError
```

```
>>>     finally:
>>>         print('finally')
>>>         return
>>> f()
```

A. try except <AssertionError logs>

B. try except finally <AssertionError logs>

C. try except <AssertionError logs> finally

D. try except finally

Question 10: What is the output of the following code?

```
>>> for x in range(1):
>>>     try:
>>>         print(x/x)
>>>     finally:
>>>         break
```

A. No output

B. 0

C. 1

D. ZeroDivisionError: division by zero

Question 11: Which option are valid replacements for the marker in the given code? (Select four answers)

```
>>> try:
>>>     x = 1/0
>>> <<< INSERT CODE HERE >>>
>>>     pass
```

A. except Exception:

B. except ArithmeticError:

C. except ArithmeticException:

D. except ZeroDivisionError:

E. except DivisionZeroError:

F. except MathError:

G. except BaseException:

Question 12: What is the output of the following code?

```
>>> try:
>>>     raise Exception
>>> except BaseException:
>>>     print("Spam")
>>> except Exception:
>>>     print("Ham")
>>> except:
>>>     print(“Eggs")
```

A. Eggs

B. Ham

C. SyntaxError

D. Spam

Question 13: What is the output of the following code?

```
>>> try:
>>>     raise Exception
>>> except:
>>>     print("Spam", end=")
```

```
>>> except BaseException:
>>>     print("Ham", end='')
>>> except Exception:
>>>     print(“Eggs")
```

A. Eggs

B. SyntaxError

C. Spam Ham Eggs

D. Spam Ham

Question 14: What is the output of the following code?

```
>>> try:
>>>     raise Exception('spam', 'eggs')
>>> except Exception as inst:
>>>     x, y = inst.args
>>> x, y
```

A. SyntaxError: invalid syntax

B. ValueError: too many values to unpack (expected 2)

C. ('spam', ‘eggs')

D. TypeError: ‘tuple’ object does not support item assignment

Question 15: What is the output of the following code?

```
>>> for x in range(1):
>>>     try:
>>>         print(x/x)
>>>     finally:
>>>         continue
```

A. 0

B. 1

C. No output

D. ZeroDivisionError: division by zero

Question 16: Which option will print (‘spam’, ‘eggs’) based on the following code? (Select two answers)

```
>>> try:
>>>     raise Exception('spam', 'eggs')
>>> except Exception as exception:
>>>     << INSERT CODE HERE >>
```

A. print(exception.args)

B. print(exception.params)

C. print(exception)

D. print(exception.iterable[:])

Question 17: What is the output of the following code?

```
>>> def f():
>>>     try:
>>>         return 0
>>>     finally:
>>>         return 1
>>> f()
```

A. 0

B. 1

C. SyntaxError: invalid syntax

D. ZeroDivisionError: division by zero

Question 18: What is the output of the following code?

```
>>> try:
>>>     a = 1/'0'
>>> except (ZeroDivisionError, TypeError) as e:
>>>     print(type(e))
```

A. Invalid Syntax

B. <class 'ZeroDivisionError'>

C. The script will run but will not print anything

D. <class 'TypeError'>

Question 19: What is the output of the following code?

```
>>> try:
>>>     raise Exception(1, 2, 3)
>>> except Exception as e:
>>>     print(len(e.args))
```

A. 3

B. 2

C. 0

D. 1

Question 20: Which of the statements below is valid? (Select two answers)

A. The finally branch in a try block will only be executed if the exception did not occur

B. The finally branch in a try block will only be executed if an exception occurs.

C. The finally branch in a try block is always executed.

D. The finally branch in a try block is required because it is always executed.

E. The finally branch in a try block is optional.

Question 21: What is the output of the following code?

```
>>> try:
>>>     raise UndefinedException
>>> except NameError:
>>>     print('NameError')
>>> except UndefinedException:
>>>     print('UndefinedException')
>>> except:
>>>     pass
```

A. UndefineException

B. NameError

C. SyntaxError: invalid syntax

D. No output

Question 22: Where will the code that may have an exception should be placed?

A. except:

B. else:

C. finally:

D. try:

Question 23: The else in the try-except-else is …? (Select two answers)

A. Executed when no matched exception is found in except

B. Executed if no exception is raised in try

C. Executed if break or continue is called in try

D. Optional

Question 24: Which option are valid given the following code? (Select two answers)

```
>>> assert x != 0
```

A. NameError if x is defined

B. AssertionError if x == 0

C. Missing parenthesis in the call to the assert function

D. AssertionError if x != 0

E. AssertionError if x is not defined

Question 25: What is the output of the following code?

```
>>> def f():
>>>     try:
>>>         1/0
>>>     finally:
>>>         return 0
>>> f()
```

A. None

B. 0

C. ZeroDivisionError: division by zero

D. SyntaxError: invalid syntax

Question 26: What is the output of the following code?

```
>>> try:
>>>     raise ValueError
>>> except TypeError, ValueError:
>>>     raise
```

A. TypeError

B. ValueError

C. No output

D. SyntaxError: invalid syntax

Question 27: What is the output of the following code if spam.txt does not exist?

```
>>> import sys
>>> try:
>>>     f = open('spam.txt')
>>>     s = f.readline()
>>> except:
>>>     raise
```

A. The script will run but will not print anything

B. Compile time error

C. FileNotFoundError: [Errno 2] No such file or directory: 'spam.txt'

D. "None" will be printed

Question 28: Which option(s) are valid except for ZeroDivisionError to be accessed as variable e? (Select two answers)

A. except (ZeroDivisionError as e):

B. except (ZeroDivisionError) as e:

C. except ZeroDivisionError(e):

D. except ZeroDivisionError e:

E. except ZeroDivisionError as e:

Question 29: What is the result of the following code? (Select two answers)

```
>>> assert(False, 'Trigger Assertion')
```

A. SyntaxError: invalid syntax

B. No output

C. Assertion is always TRUE

D. Trigger Assertion

Question 30: What will happen if spam.py is run?

```
# spam.py
>>> try:
>>>     print(x)
>>> except:
>>>     print("An exception occurred”)
```

A. An exception occurred will printed

B. None will be printed

C. Compile time error

D. The script will run but will not print anything

Question 31: Where will the code that handles the exception should be placed?

A. else:

B. except:

C. try:

D. finally:

Question 32: What is the output of the following code?

```
>>> try:
>>>     raise UndefinedException
>>>  except:
>>>     pass
```

A. Add () on Line 2 to fix the syntax error

B. NameError: name ‘UndefinedException’ is not defined

C. SyntaxError: invalid syntax

D. No output

Question 33: Which of the statements below is valid?

>>> spam = 0

>>> assert spam == 0

A. The word True will be printed on the screen

B. Missing parenthesis in call to assert error will be displayed

C. No AssertionError will be triggered since the expression is True

D. AssertionError will be triggered because the expression is True

Question 34: If there is more than 1 except clause, what happens after a try clause executes? (Select two answers)

A. None of the except is executed

B. At least 01 except is executed

C. Exactly 01 of the except is executed

D. Not more than 01 except is executed

Question 35: What is the output of the following code?

>>> type(Exception().args)

A. <class 'list'>

B. <class 'str'>

C. <class 'tuple'>

D. <class 'dict'>

Question 36: What is the output of the following code?

>>> try:

>>> raise IOError

>>> except IOError as e:

>>> raise RuntimeError from e

A. RuntimeError

B. SyntaxError: invalid syntax

C. IOError

D. No output

Question 37: What is the expected output of the following code?

```
>>> x, y = 3.0, 0.0
>>> try:
>>>     z = x / y
>>> except ArithmeticError:
>>>     z = -1
>>> else:
>>>     z = -2
>>> print(z):
```

A. +INF

B. -1

C. -2

D. An error message appears on the screen.

Question 38: What is the expected output of the following code?

```
>>> x, y = 0.0, 3.0
>>> try:
>>>     z = x/y
>>> except ArithmeticError:
>>>     z = -1
>>> else:
```

```
>>>     z = -2
>>> print(z)
```

A. +INF

B. -1

C. An error message appears on the screen

D. -2

Question 39: What is the expected output of the following code?

```
>>> def fun(x):
>>>     return 1/x


>>> def mid_level(x):
>>>     try:
>>>         fun(x)
>>>     except:
>>>         raise AssertionError
>>>     else:
>>>         return 0


>>> try:
>>>     x = mid_level(0)
>>> except Exception:
>>>     x = -1
>>> except:
>>>     x = -2
```

>>> print(x)

A. 0

B. -1

C. -2

D. An error message appears on the screen

Question 40: What is the expected output of the following code?

>>> consts = [3.141592, 2.718282]

>>> try:

>>> print(consts.index (314e-2))

>>> except Exception as exception:

>>> print(exception.args)

>>> else:

>>> print(“success”)

A. -1

B. False

C. (‘success’)

D. (‘3.14 is not in list’,)