✅ Python Strings – MCQs

1. What will be the output of the following code?

python

CopyEdit

```
str1 = "Python"
print(str1[1])
```

A. P
B. y
C. t
D. n

Answer: ✅ B. y

Explanation: Indexing starts from 0, so str1[1] is 'y'.

---

2. Which of the following string methods removes whitespace from the beginning and end of a string?

A. remove()
B. trim()
C. strip()
D. delete()

Answer: ✅ C. strip()

Explanation: strip() removes leading and trailing whitespace.

---

3. What will the following expression return?

python

CopyEdit

```
"Hello" + "World"
```

A. Hello World
B. HelloWorld
C. Hello+World
D. Error

Answer: ✅ B. HelloWorld

Explanation: The + operator concatenates strings without adding space.

---

4. Which operator is used to repeat a string multiple times in Python?

A. +
B. *
C. %
D. **

Answer: ✅ B. *

Explanation: * is used to repeat a string (e.g., "abc" * 3 → 'abcabcabc').

---

5. Which method is used to convert all characters of a string to uppercase?
A. upper()
B. uppercase()
C. capitalize()
D. toUpper()
Answer: ✅ A. upper()
Explanation: upper() converts all characters to uppercase.

---

6. What will the following code print?
python
CopyEdit
```
s = "banana"
print(s.count("a"))
```
A. 1
B. 2
C. 3
D. 4
Answer: ✅ C. 3
Explanation: There are three 'a' characters in "banana".

---

7. Which of the following is a correct way to create a multiline string in Python?
A. Use '''triple quotes'''
B. Use """triple quotes"""
C. Use \n for new lines
D. All of the above
Answer: ✅ D. All of the above
Explanation: Multiline strings can be created using triple quotes or by inserting \n.

---

8. Strings in Python are:
A. Mutable
B. Immutable
C. Static
D. Dynamic
Answer: ✅ B. Immutable
Explanation: Once a string is created, it cannot be changed (i.e., it's immutable).

---

9. What is the output of this code?
python
CopyEdit
```
msg = "Hello World"
print(msg.lower())
```
A. Hello World
B. hello world
C. HELLO WORLD
D. Error
Answer: ✅ B. hello world
Explanation: lower() converts all characters to lowercase.

---

10. What will len("Python") return?
A. 5
B. 6
C. 7
D. Error
Answer: ✅ B. 6
Explanation: "Python" contains 6 characters.

✅ Python String Methods – MCQs

1. What does the upper() method do in Python?
A. Converts all letters to lowercase
B. Returns the first uppercase letter
C. Converts all letters to uppercase
D. Capitalizes the first word
Answer: ✅ C. Converts all letters to uppercase

---

2. What is the output of this code?
python
CopyEdit
```
txt = "hello world"
print(txt.capitalize())
```
A. hello world
B. Hello World
C. HELLO WORLD
D. Hello world
Answer: ✅ D. Hello world
Explanation: capitalize() makes only the first character uppercase and the rest lowercase.

---

3. Which method would you use to find the position of a substring within a string?
A. locate()
B. find()
C. position()
D. indexOf()
Answer: ✅ B. find()
Explanation: find() returns the index of the first occurrence of a substring.

---

4. What is the output of the following code?
python
CopyEdit
```
s = "banana"
print(s.replace("a", "o"))
```
A. bonono
B. banana
C. bonana
D. Error
Answer: ✅ A. bonono
Explanation: replace() replaces all occurrences of "a" with "o".

---

5. Which method checks if all characters in a string are digits?
A. isnumeric()
B. isdigit()
C. isalnum()
D. isdecimal()
Answer: ✅ B. isdigit()
Explanation: isdigit() checks if all characters in the string are digits.

---

6. What will this code output?
python
CopyEdit
```
text = "   python   "
print(text.strip())
```
A. " python"
B. "python"
C. " python "
D. Error
Answer: ✅ B. "python"
Explanation: strip() removes leading and trailing whitespaces.

---

7. Which method returns True if all characters are alphabetic and there is at least one character?
A. isalpha()
B. isalnum()
C. ischar()
D. istext()
Answer: ✅ A. isalpha()

---

8. What does the split() method return?
A. A string
B. A tuple
C. A list
D. A dictionary
Answer: ✅ C. A list
Explanation: split() breaks a string into a list using a delimiter (default is space).

---

9. Which of the following methods will convert a list of words into a single string?
A. combine()
B. append()
C. join()
D. concat()
Answer: ✅ C. join()
Explanation: join() is used to join elements of a list into a single string.

---

10. What will s = "Hello123".isalnum() return?
A. True
B. False
C. 'Hello'
D. Error
Answer: ✅ A. True
Explanation: isalnum() returns True if the string contains only letters and digits.

✅ Python String Operations – MCQs

1. What will be the result of the following operation?
python
CopyEdit
"Hello" + "World"
A. Hello World
B. Hello+World
C. HelloWorld
D. Error
Answer: ✅ C. HelloWorld
Explanation: The + operator concatenates strings.

---

2. What does the * operator do when used with strings?
python
CopyEdit
print("Hi" * 3)
A. HiHiHi
B. Hi3
C. Hi Hi Hi
D. Error
Answer: ✅ A. HiHiHi
Explanation: The * operator repeats the string a specified number of times.

---

3. What will the output be?
python
CopyEdit
str = "Python"
print(str[1:4])
A. Pyt
B. yth
C. ytho
D. ytho
Answer: ✅ B. yth
Explanation: Slicing returns characters from index 1 up to (but not including) 4.

---

4. Which of the following is a correct way to check if the substring "py" is in "python"?
A. "py" in "python"
B. "py" exists "python"
C. "python".contains("py")
D. "python" has "py"
Answer: ✅ A. "py" in "python"
Explanation: in is the membership operator for strings.

---

5. What will the following return?
python
CopyEdit
s1 = "abc"
s2 = "abc"
print(s1 == s2)
A. True
B. False
C. abcabc
D. Error
Answer: ✅ A. True
Explanation: == compares the values of the strings.

---

6. What does this return?
python
CopyEdit
"Python"[-1]
A. P
B. o
C. n
D. Error
Answer: ✅ C. n
Explanation: Negative indexing counts from the end of the string.

---

7. What is the result of "abc" < "abd"?
A. True
B. False
C. None
D. Error
Answer: ✅ A. True
Explanation: Strings are compared lexicographically using ASCII values.

---

8. Which operator is used to check if two strings are not equal?
A. <>
B. ==!
C. !=
D. !==
Answer: ✅ C. !=
Explanation: != checks for inequality in Python.

---

9. What does the following code return?
python
CopyEdit
s = "Hello World"
print('W' in s)
A. True
B. False
C. W
D. Error
Answer: ✅ A. True
Explanation: The character 'W' exists in the string.

---

10. What is the result of this expression?
python
CopyEdit
"abc" * 0
A. abc
B. 0
C. "" (empty string)
D. Error
Answer: ✅ C. ""
Explanation: Repeating a string 0 times returns an empty string.

✅ Python String Formatting – MCQs

1. What is the output of this code?
python
CopyEdit
name = "Alice"
print("Hello, %s!" % name)
A. Hello, name!
B. %s, Alice!
C. Hello, Alice!
D. Hello, %s!
Answer: ✅ C. Hello, Alice!
Explanation: %s is a placeholder for string in old-style formatting.

---

2. Which of the following is the correct way to format a float value to 2 decimal places using str.format()?
A. "{:.2f}".format(3.14159)
B. "{:2f}".format(3.14159)
C. "%.2f".format(3.14159)
D. format("%.2f", 3.14159)
Answer: ✅ A. "{:.2f}".format(3.14159)
Explanation: :.2f formats a float to 2 decimal places.

---

3. What will be the output of this code?
python
CopyEdit
```python
name = "Bob"
age = 25
print(f"My name is {name} and I am {age} years old.")
```
A. My name is {name} and I am {age} years old.
B. My name is Bob and I am 25 years old.
C. My name is name and I am age years old.
D. Error

Answer: ✅ B. My name is Bob and I am 25 years old.
Explanation: f-strings evaluate expressions inside {}.

---

4. What is the output of this code using .format()?
python
CopyEdit
```python
print("Welcome {} to {}".format("Alice", "Wonderland"))
```
A. Welcome Alice to Wonderland
B. Welcome {} to {}
C. Welcome Wonderland to Alice
D. Error

Answer: ✅ A. Welcome Alice to Wonderland
Explanation: The format() method replaces {} in order.

---

5. What will this code return?
python
CopyEdit
```python
value = 42
print("The answer is %d." % value)
```
A. The answer is 42.
B. The answer is value.
C. The answer is %d.
D. Error

Answer: ✅ A. The answer is 42.
Explanation: %d is used for integers in old-style formatting.

---

6. Which string formatting method is the most modern and recommended in Python 3.6+?
A. % formatting
B. str.format()
C. f-strings
D. .join()

Answer: ✅ C. f-strings
Explanation: f-strings are more readable and efficient in Python 3.6+.

---

7. How would you insert a variable into a string using str.format()?
A. "Hello, {0}".format(name)
B. "Hello, %s".format(name)
C. "Hello, {}".format[name]
D. "Hello, {name}".format(name)
Answer: ✅ A. "Hello, {0}".format(name)
Explanation: Positional arguments use index numbers inside {}.

---

8. What will be the output of this code?
python
CopyEdit
x = 3.4567
print(f"Rounded: {x:.1f}")
A. Rounded: 3.4
B. Rounded: 3.5
C. Rounded: 3.4567
D. Rounded: 3
Answer: ✅ B. Rounded: 3.5
Explanation: Rounds to 1 decimal place using :.1f.

---

9. Which format specifier is used to represent a string in % formatting?
A. %d
B. %f
C. %s
D. %x
Answer: ✅ C. %s
Explanation: %s is the format specifier for strings.

---

10. How can you include curly braces {} as literal characters in an f-string?
A. Use \{ and \}
B. Use double braces: {{ and }}
C. You can't include them
D. Use .escape() method
Answer: ✅ B. Use double braces: {{ and }}
Explanation: {{ and }} escape the braces in f-strings.

✅ Python String Operators – MCQs

1. Which of the following is used to concatenate two strings in Python?
A. *
B. %
C. +
D. &
Answer: ✅ C. +
Explanation: The + operator is used to join two or more strings.

---

2. What is the result of the following operation?
python
CopyEdit
```
"Hi" * 3
```
A. Hi3
B. Hi Hi Hi
C. HiHiHi
D. Error
Answer: ✅ C. HiHiHi
Explanation: The * operator repeats the string.

---

3. Which operator is used to check if a substring exists within a string?
A. has
B. includes
C. in
D. contains
Answer: ✅ C. in
Explanation: The in operator checks for membership in a string.

---

4. What will be the output of this code?
python
CopyEdit
```
"a" in "apple"
```
A. True
B. False
C. a
D. Error
Answer: ✅ A. True
Explanation: 'a' is present in 'apple'.

---

5. Which of the following is a valid use of string comparison in Python?
A. "cat" >= "dog"
B. "123" > 123
C. "hello" != "Hello"
D. Both A and C
Answer: ✅ D. Both A and C
Explanation: Python allows lexicographic comparison using relational operators.

---

6. What will this code output?
python
CopyEdit
```
"abc" == "abc"
```
A. abcabc
B. True
C. False
D. Error
Answer: ✅ B. True
Explanation: The == operator checks if two strings have the same value.

7. What does the expression "Python" != "python" return?
A. True
B. False
C. Error
D. None
Answer: ✅ A. True
Explanation: String comparison is case-sensitive in Python.

---

8. Which of the following operations will result in an error?
A. "abc" + "def"
B. "abc" * 2
C. "abc" - "a"
D. "abc" in "abcdef"
Answer: ✅ C. "abc" - "a"
Explanation: - is not a valid string operator.

---

9. What is the result of "a" * 0?
A. a0
B. 0
C. "" (empty string)
D. None
Answer: ✅ C. "" (empty string)
Explanation: Repeating a string zero times gives an empty string.

---

10. Which operator is used to test for string inequality in Python?
A. !==
B. <>
C. !=
D. not==
Answer: ✅ C. !=
Explanation: != checks for inequality between strings.

✅ Python f-Strings – MCQs

1. What is an f-string in Python?
A. A string with formatting syntax using %
B. A string with the format method
C. A string prefixed with f or F that supports expression interpolation
D. A string used in function definitions
Answer: ✅ C. A string prefixed with f or F that supports expression interpolation

---

2. What will be the output of this code?

python

CopyEdit

```
name = "Alice"
print(f"Hello, {name}!")
```

A. Hello, name!

B. Hello, {name}!

C. Hello, Alice!

D. Error

Answer: ✅ C. Hello, Alice!

Explanation: The {name} inside the f-string is evaluated.

---

3. Which of the following is NOT true about f-strings?

A. They were introduced in Python 3.6

B. They support inline expressions

C. They are slower than str.format()

D. They use curly braces {} for expression evaluation

Answer: ✅ C. They are slower than str.format()

Explanation: f-strings are generally faster and more readable.

---

4. What will this code print?

python

CopyEdit

```
x = 10
y = 20
print(f"{x} + {y} = {x + y}")
```

A. x + y = x + y

B. {x} + {y} = {x + y}

C. 10 + 20 = 30

D. Error

Answer: ✅ C. 10 + 20 = 30

---

5. How do you include literal {} in an f-string?

A. Use a backslash: \{}

B. Use double braces: {{}}

C. Use .format() instead

D. Use % formatting

Answer: ✅ B. Use double braces: {{}}

Explanation: {{ and }} allow literal curly braces in f-strings.

---

6. Which of the following f-strings will correctly format a float to 2 decimal places?

A. f"{value.2f}"

B. f"{value:.2f}"

C. f"{:2f}.value"

D. f"{value:2f}"

Answer: ✅ B. f"{value:.2f}"

Explanation: :.2f formats a float to two decimal places.

---

7. Can you use function calls inside f-strings?
A. No
B. Only simple functions
C. Yes
D. Only math functions
Answer: ✅ C. Yes
Explanation: f-strings can evaluate any valid Python expression.

---

8. What will be the output?
python
CopyEdit
```
def greet():
    return "Hi"
print(f"Greeting: {greet()}")
```
A. Greeting: greet()
B. Greeting: Hi
C. Greeting: {greet()}
D. Error
Answer: ✅ B. Greeting: Hi

---

9. What will this code display?
python
CopyEdit
```
name = "Bob"
print(f"{name.lower()} is here")
```
A. BOB is here
B. bob is here
C. Bob is here
D. Error
Answer: ✅ B. bob is here
Explanation: You can call methods inside f-strings.

---

10. Which Python version introduced f-strings?
A. 2.7
B. 3.5
C. 3.6
D. 3.7
Answer: ✅ C. 3.6

✅ Interning in Python – MCQs

1. What is interning in Python?
A. The process of installing third-party packages
B. The method of converting integers to strings
C. Reusing immutable objects like strings to save memory
D. The way Python handles nested functions
Answer: ✅ C. Reusing immutable objects like strings to save memory
Explanation: Interning allows identical immutable objects (like strings) to share the same memory.

---

2. Which of the following objects are most commonly interned by Python?
A. Lists
B. Dictionaries
C. Strings
D. Tuples
Answer: ✅ C. Strings
Explanation: Strings are frequently interned for performance and memory efficiency.

---

3. What will be the result of the following code?
python
CopyEdit
a = "python"
b = "python"
print(a is b)
A. True
B. False
C. Error
D. Depends on memory
Answer: ✅ A. True
Explanation: Identical string literals are usually interned, so a and b point to the same memory.

---

4. Which module provides a method to manually intern a string in Python?
A. sys
B. string
C. re
D. keyword
Answer: ✅ A. sys
Explanation: sys.intern() can be used to manually intern strings.

---

5. What will this output?

python
CopyEdit

```
import sys
a = sys.intern("hello world")
b = sys.intern("hello world")
print(a is b)
```

A. False
B. hello world
C. True
D. Error

Answer: ✅ C. True
Explanation: sys.intern() ensures both strings refer to the same memory location.

---

6. Interning is mostly applied to which kind of strings by default in Python?
A. All strings
B. Multi-line strings
C. Short strings or identifiers
D. Strings with numbers only

Answer: ✅ C. Short strings or identifiers
Explanation: Python interns short strings or those resembling variable names automatically.

---

7. Which of these comparisons checks memory reference (not value)?
A. a == b
B. a is b
C. a.equals(b)
D. a in b

Answer: ✅ B. a is b
Explanation: is checks if two variables point to the same memory.

---

8. Why is string interning useful in Python?
A. It increases string size
B. It avoids variable name collisions
C. It speeds up dictionary lookups and saves memory
D. It helps with type conversions

Answer: ✅ C. It speeds up dictionary lookups and saves memory
Explanation: Interning reduces memory usage and improves performance in some cases.

---

9. Which of the following will NOT be interned by default?
A. "abc"
B. "hello world"
C. "identifier"
D. "foo123"

Answer: ✅ B. "hello world"
Explanation: Strings with spaces or non-identifier patterns are not interned automatically.

---

10. What happens if two identical strings are not interned?
A. is will return True
B. == will return False
C. is will return False
D. == will raise an error
Answer: ✅ C. is will return False
Explanation: is returns False if objects have the same value but different memory references.

Topic: Type Casting in Python
Q1. What is type casting in Python?
A. Converting a string to binary
B. Assigning multiple types to a variable
C. Converting one data type to another
D. Creating a new variable
✅ **Correct Answer:** C

Q2. What is the output of the following code?
python
CopyEdit
```
x = int(3.9)
print(x)
```
A. 4
B. 3.9
C. 3
D. Error
✅ **Correct Answer:** C
✔ Explanation: int() truncates the decimal part.

Q3. Which of the following will result in an error?
A. int("10")
B. float("5.6")
C. str(25)
D. int("3.14")
✅ **Correct Answer:** D
✔ Explanation: "3.14" is a float in string format and can't be directly converted to int.

Q4. What is the output of the following code?
python
CopyEdit
```
x = float(5)
print(x)
```
A. 5
B. 5.0
C. '5.0'
D. Error
✅ **Correct Answer:** B

Q5. What will str(10 + 5) return?
A. '10' + '5'
B. '10+5'
C. '15'
D. 15
✅ **Correct Answer:** C

Q6. What is the result of int(True) in Python?
A. 0
B. 1
C. True
D. Error
✅ **Correct Answer:** B
✔ Explanation: In type casting, True → 1, False → 0.

Q7. What is the output of the following code?
python
CopyEdit
```
a = "100"
b = 20
print(int(a) + b)
```
A. "10020"
B. 120
C. Error
D. None
✅ **Correct Answer:** B

Q8. Which function is used to convert an object to a string in Python?
A. str()
B. string()
C. toString()
D. repr()
✅ **Correct Answer:** A

Q9. What will be the output of this code?
python
CopyEdit
```
print(bool("False"))
```
A. False
B. True
C. Error
D. None
✅ **Correct Answer:** B
✔ Explanation: Any non-empty string, including "False", is truthy.

Q10. What is the output of the following code?
python
CopyEdit
print(type(float("10.5")))
A. <class 'int'>
B. <class 'float'>
C. <class 'str'>
D. <class 'bool'>
✅ **Correct Answer:** B

Implicit Type Casting in Python

Q1. What is implicit type casting in Python?
A. When the user manually converts one type to another
B. When Python automatically converts one data type to another
C. When data is encrypted during assignment
D. None of the above
✅ **Correct Answer:** B

Q2. What is the output of the following code?
python
CopyEdit
x = 10
y = 2.5
z = x + y
print(type(z))
A. <class 'int'>
B. <class 'float'>
C. <class 'str'>
D. Error
✅ **Correct Answer:** B
✔ Explanation: Python implicitly converts int to float for accurate computation.

Q3. Which of the following is an example of implicit type casting?
A. float("5.0")
B. int("10")
C. 3 + 4.5
D. str(123)
✅ **Correct Answer:** C
✔ Explanation: int (3) is implicitly converted to float (3.0).

Q4. What is the output of the following code?

python
CopyEdit

```
a = 5
b = 3.2
print(a * b)
```

A. 15.0
B. 16
C. 16.0
D. Error

✅ **Correct Answer:** C

✔ a is implicitly cast to float, and the result is a float.

Q5. Which of the following types of casting is done automatically by Python?

A. str to int
B. float to int
C. int to float
D. str to float

✅ **Correct Answer:** C

✔ Explanation: Only widening conversions (like int → float) happen implicitly.

Q6. What will be the result of the following?

python
CopyEdit

```
a = True
b = 5
print(a + b)
```

A. 6
B. 5
C. True5
D. Error

✅ **Correct Answer:** A

✔ True is implicitly cast to 1, so result is 1 + 5 = 6.

Q7. Implicit type casting occurs when:

A. Python changes the type to avoid data loss
B. Python changes the type to prevent type mismatch
C. Python explicitly uses int() or float()
D. The user changes the data type

✅ **Correct Answer:** B

Q8. Which of the following will cause implicit type casting?

A. "10" + 5
B. 10 + 5.5
C. int("5.5")
D. str(123)

✅ **Correct Answer:** B

Q9. In implicit casting, which of the following statements is TRUE?
A. Python always converts float to int
B. Python always converts int to string
C. Python converts smaller data types to larger data types when needed
D. Python does not allow implicit casting
✅ **Correct Answer:** C

Q10. What will be the type of the result in this operation?
python
CopyEdit
result = 4 + True
print(type(result))
A. <class 'bool'>
B. <class 'str'>
C. <class 'int'>
D. <class 'float'>
✅ **Correct Answer:** C
✔ Because True is implicitly treated as 1, and both are int.

Q1. What is explicit type casting in Python?
A. When Python automatically converts types during operations
B. When the user manually converts a value from one type to another
C. When strings are encrypted
D. When Python stores string values as float
✅ **Correct Answer:** B

Q2. Which of the following is an example of explicit type casting?
A. a = 5 + 3.2
B. b = int("10")
C. c = True + 2
D. d = 3 * 2.5
✅ **Correct Answer:** B

Q3. What will be the output of the following code?
python
CopyEdit
x = float("5.6")
print(x)
A. 5.6
B. '5.6'
C. 5
D. Error
✅ **Correct Answer:** A

Q4. What is the result of the following code?

python

CopyEdit

```
x = "10"
y = int(x)
print(y + 5)
```

A. 105

B. 15

C. "10" + 5

D. Error

✅ **Correct Answer:** B

Q5. Which function is used to convert a number to a string in Python?

A. int()

B. float()

C. str()

D. char()

✅ **Correct Answer:** C

Q6. What will the following code output?

python

CopyEdit

```
print(int(3.99))
```

A. 4

B. 3.99

C. 3

D. Error

✅ **Correct Answer:** C

✔ Explanation: int() truncates the decimal part, it doesn't round.

Q7. Which of these conversions is NOT possible using explicit type casting in Python?

A. int("10")

B. float("3.14")

C. int("3.14")

D. str(100)

✅ **Correct Answer:** C

✔ Explanation: "3.14" is a string representing a float and can't be directly cast to int.

Q8. What will this code print?
python
CopyEdit
a = bool(0)
print(a)
A. True
B. False
C. 0
D. Error
✅ **Correct Answer:** B
✔ Explanation: bool(0) is False; only non-zero values are True.

Q9. Which function is used to convert a value to a float in Python?
A. floatify()
B. toFloat()
C. float()
D. decimal()
✅ **Correct Answer:** C

Q10. What will be the output of the following code?
python
CopyEdit
a = "20.5"
b = float(a)
c = int(b)
print(c)
A. 20.5
B. 21
C. Error
D. 20
✅ **Correct Answer:** D
✔ Explanation: float("20.5") → 20.5, then int(20.5) → 20 (truncated).

MCQs on Loops in Python

---

1. Which of the following loops is not supported by Python?
A. for loop
B. while loop
C. do-while loop
D. All of the above
✅ **Answer:** C. do-while loop

---

2. What will the following code output?

python

CopyEdit

```
for i in range(3):
   print(i, end=" ")
```

A. 0 1 2

B. 1 2 3

C. 0 1 2 3

D. Error

✅ **Answer:** A. 0 1 2

---

3. Which function is used to create a sequence of numbers in a for loop?

A. sequence()

B. list()

C. range()

D. array()

✅ **Answer:** C. range()

---

4. How many times will this loop execute?

python

CopyEdit

```
i = 0
while i < 5:
   i += 1
```

A. 4

B. 5

C. Infinite

D. 0

✅ **Answer:** B. 5

---

5. What is the output of this code?

python

CopyEdit

```
for i in range(1, 10, 3):
   print(i, end=" ")
```

A. 1 2 3

B. 1 4 7

C. 1 4 7 10

D. 1 4 7 10 13

✅ **Answer:** B. 1 4 7

---

6. Which keyword is used to skip the current iteration in a loop?

A. exit

B. skip

C. continue

D. pass

✅ **Answer:** C. continue

---

7. What does the following code print?

python

CopyEdit

```
for i in range(5):
   if i == 3:
      break
   print(i)
```

A. 0 1 2

B. 0 1 2 3

C. 1 2 3

D. 1 2

✅ **Answer:** A. 0 1 2

---

8. What is the purpose of the else block in a loop?

A. Executes when the loop condition is False

B. Executes when the loop ends normally (no break)

C. Executes when the loop contains a continue

D. Executes only when loop never runs

✅ **Answer:** B. Executes when the loop ends normally (no break)

---

9. What will be the output?

python

CopyEdit

```
x = 0
while x < 3:
   print(x)
   x += 1
else:
   print("Done")
```

A. 0 1 2

B. 0 1 2 Done

C. Done

D. Infinite loop

✅ **Answer:** B. 0 1 2 Done

---

10. Which of these is an infinite loop?

A. for i in range(10):

B. while True:

C. while i < 10:

D. for i in range(1,10,2):

✅ **Answer:** B. while True:

MCQs on Iteration in Python

---

1. What does iteration mean in Python?
A. Repeating a block of code a fixed number of times
B. Converting data types
C. Stopping program execution
D. Writing functions
✅ **Answer:** A. Repeating a block of code a fixed number of times

---

2. Which Python statement is used for iteration?
A. repeat
B. loop
C. for
D. goto
✅ **Answer:** C. for

---

3. What will this code output?
python
CopyEdit
```
for char in "abc":
    print(char.upper(), end=" ")
```
A. abc
B. A B C
C. ABC
D. a b c
✅ **Answer:** B. A B C

---

4. Which of these data types can be iterated using a for loop?
A. List
B. Dictionary
C. String
D. All of the above
✅ **Answer:** D. All of the above

---

5. What is the output of the following code?
python
CopyEdit
```
fruits = ['apple', 'banana', 'cherry']
for fruit in fruits:
    print(fruit[0])
```
A. apple banana cherry
B. a b c
C. fruit
D. ['a', 'b', 'c']
✅ **Answer:** B. a b c

---

6. How do you iterate over both index and value in a list?
A. for i in list:
B. for i, val in enumerate(list):
C. for val in list, i:
D. for i = 0 to len(list):
✅ **Answer:** B. for i, val in enumerate(list):

---

7. What is the output of this dictionary iteration?
python
CopyEdit
```
person = {'name': 'Alice', 'age': 25}
for key in person:
    print(key)
```
A. Alice 25
B. name age
C. ['name', 'age']
D. ('name', 'age')
✅ **Answer:** B. name age

---

8. What will this code print?
python
CopyEdit
```
for i in range(0, 10, 2):
    print(i, end=", ")
```
A. 0, 2, 4, 6, 8,
B. 0 2 4 6 8
C. 1 3 5 7 9
D. 0 1 2 3 4
✅ **Answer:** A. 0, 2, 4, 6, 8,

---

9. What is iter() used for in Python?
A. To break a loop
B. To repeat a string
C. To get an iterator from an iterable
D. To create a tuple
✅ **Answer:** C. To get an iterator from an iterable

---

10. Which of the following will raise a TypeError?
A. for x in [1, 2, 3]:
B. for x in "hello":
C. for x in 123:
D. for x in (1, 2, 3):
✅ **Answer:** C. for x in 123:

✅ MCQs on Lists in Python

---

1. Which of the following is the correct way to create a list in Python?

A. list = (1, 2, 3)
B. list = {1, 2, 3}
C. list = [1, 2, 3]
D. list = <1, 2, 3>
✅ **Answer:** C. list = [1, 2, 3]

---

2. What is the output of the following code?

python
CopyEdit
```
my_list = [10, 20, 30]
print(my_list[1])
```
A. 10
B. 20
C. 30
D. IndexError
✅ **Answer:** B. 20

---

3. Which method adds a new element at the end of a list?

A. add()
B. append()
C. insert()
D. extend()
✅ **Answer:** B. append()

---

4. What does the len() function return when used on a list?

A. Number of bytes in the list
B. Number of elements in the list
C. Highest number in the list
D. None of the above
✅ **Answer:** B. Number of elements in the list

---

5. What will be the output?

python
CopyEdit
```
a = [1, 2, 3]
a[1] = 100
print(a)
```
A. [1, 2, 3]
B. [100, 2, 3]
C. [1, 100, 3]
D. Error
✅ **Answer:** C. [1, 100, 3]

---

6. How do you add multiple items to a list at once?
A. append([4, 5])
B. extend([4, 5])
C. insert([4, 5])
D. add([4, 5])
✅ **Answer:** B. extend([4, 5])

---

7. Which of the following removes all items from a list?
A. list.delete()
B. list.clear()
C. list.remove()
D. list.popall()
✅ **Answer:** B. list.clear()

---

8. What is the result of the following code?
python
CopyEdit
```
nums = [1, 2, 3]
print(nums * 2)
```
A. [1, 2, 3, 1, 2, 3]
B. [2, 4, 6]
C. [[1, 2, 3], [1, 2, 3]]
D. Error
✅ **Answer:** A. [1, 2, 3, 1, 2, 3]

---

9. Which method removes an item by value from the list?
A. del()
B. pop()
C. remove()
D. discard()
✅ **Answer:** C. remove()

---

10. What will this output?
python
CopyEdit
```
lst = [1, 2, 3, 4, 5]
print(lst[1:4])
```
A. [2, 3, 4]
B. [1, 2, 3, 4]
C. [1, 2, 3]
D. [2, 3, 4, 5]
✅ **Answer:** A. [2, 3, 4]

MCQs on List Methods in Python

1. Which list method is used to add a single element to the end of the list?
A. add()
B. append()
C. insert()
D. extend()
✅ **Answer:** B. append()

---

2. What is the output of this code?
python
CopyEdit
```
a = [1, 2, 3]
a.append([4, 5])
print(a)
```
A. [1, 2, 3, 4, 5]
B. [1, 2, 3, [4, 5]]
C. [1, 2, 3, (4, 5)]
D. Error
✅ **Answer:** B. [1, 2, 3, [4, 5]]

---

3. Which method is used to insert an element at a specific index?
A. append()
B. add()
C. insert()
D. extend()
✅ **Answer:** C. insert()

---

4. What is the difference between append() and extend()?
A. append() adds multiple elements, extend() adds one
B. append() adds one element, extend() adds elements from an iterable
C. append() replaces the list, extend() creates a new list
D. Both do the same thing
✅ **Answer:** B. append() adds one element, extend() adds elements from an iterable

---

5. Which method removes and returns the last element of a list by default?
A. delete()
B. remove()
C. pop()
D. discard()
✅ **Answer:** C. pop()

---

6. What will this print?
python
CopyEdit
```
a = [1, 2, 3, 4]
a.remove(3)
print(a)
```
A. [1, 2, 4]
B. [1, 2, 3]
C. [1, 2, 3, 4]
D. Error
✅ **Answer:** A. [1, 2, 4]

---

7. Which method is used to remove all elements from a list?
A. delete()
B. popall()
C. remove()
D. clear()
✅ **Answer:** D. clear()

---

8. What does the sort() method do?
A. Returns a sorted copy of the list
B. Sorts the list in place
C. Sorts and reverses the list
D. None of the above
✅ **Answer:** B. Sorts the list in place

---

9. What does the following code print?
python
CopyEdit
```
a = [3, 1, 2]
a.sort(reverse=True)
print(a)
```
A. [1, 2, 3]
B. [3, 2, 1]
C. [2, 3, 1]
D. Error
✅ **Answer:** B. [3, 2, 1]

---

10. Which method reverses the order of a list in place?
A. flip()
B. reverse()
C. sort(reverse=True)
D. invert()
✅ **Answer:** B. reverse()

Iterating Over Lists in Python

1. What is the output of this code?

```
python
CopyEdit
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

A. apple banana cherry
B. ['apple', 'banana', 'cherry']
C. Each fruit printed on a new line
D. Error

✅ **Answer:** C. Each fruit printed on a new line

---

2. Which method can be used to loop through a list and get both index and value?

A. enumerate()
B. index()
C. zip()
D. range()

✅ **Answer:** A. enumerate()

---

3. What does the following code print?

```
python
CopyEdit
nums = [10, 20, 30]
for i in range(len(nums)):
    print(nums[i])
```

A. 10 20 30
B. 0 1 2
C. [10, 20, 30]
D. Error

✅ **Answer:** A. 10 20 30

---

4. Which loop structure is commonly used to iterate over a list when index is not needed?

A. for i in range(len(list)):
B. for item in list:
C. while i < len(list):
D. loop list:

✅ **Answer:** B. for item in list:

---

5. How many times will the following loop run?
python
CopyEdit
```
colors = ["red", "green", "blue"]
for color in colors:
    print(color)
```
A. 0
B. 1
C. 3
D. Depends on user input
✅ **Answer:** C. 3

---

6. What is the output of this code?
python
CopyEdit
```
names = ["John", "Jane", "Jack"]
for index, name in enumerate(names):
    print(index, name)
```
A. John Jane Jack
B. 0 John 1 Jane 2 Jack
C. name index
D. Error
✅ **Answer:** B. 0 John 1 Jane 2 Jack

---

7. Which of the following is NOT a correct way to iterate through a list?
A. for i in my_list:
B. for i in range(len(my_list)):
C. while i < len(my_list):
D. for i = 0 to len(my_list):
✅ **Answer:** D. for i = 0 to len(my_list):

---

8. What does the following code do?
python
CopyEdit
```
items = ["a", "b", "c"]
for i in reversed(items):
    print(i)
```
A. Prints a b c
B. Prints c b a
C. Error
D. Reverses the list permanently
✅ **Answer:** B. Prints c b a

---

9. Which function is used to iterate over multiple lists at once?
A. map()
B. zip()
C. enumerate()
D. all()
✅ **Answer:** B. zip()

---

10. What will this code print?
python
CopyEdit
```
numbers = [1, 2, 3]
squares = []
for num in numbers:
    squares.append(num ** 2)
print(squares)
```
A. [1, 2, 3]
B. [2, 4, 6]
C. [1, 4, 9]
D. [1, 2, 3, 4, 5, 6]
✅ **Answer:** C. [1, 4, 9]

MCQs on Tuples in Python

1. Which of the following is the correct way to create a tuple in Python?
A. t = (1, 2, 3)
B. t = [1, 2, 3]
C. t = {1, 2, 3}
D. t = tuple[1, 2, 3]
✅ **Answer:** A. t = (1, 2, 3)

---

2. What is the output of the following code?
python
CopyEdit
```
t = (1, 2, 3)
print(t[1])
```
A. 1
B. 2
C. 3
D. Error
✅ **Answer:** B. 2

---

3. Tuples are _______ in Python.
A. Mutable
B. Immutable
C. Optional
D. Convertible
✅ **Answer:** B. Immutable

---

4. What will be the type of this variable?

python

CopyEdit

```
t = (5)
print(type(t))
```

A. <class 'tuple'>

B. <class 'int'>

C. <class 'list'>

D. <class 'str'>

✅ **Answer:** B. <class 'int'>

(You must add a comma to make it a tuple: (5,))

---

5. Which method is used to count the number of occurrences of a value in a tuple?

A. count()

B. index()

C. find()

D. match()

✅ **Answer:** A. count()

---

6. Which of the following will raise an error?

A. t = (1, 2, 3)

B. len(t)

C. t[0] = 10

D. t.index(2)

✅ **Answer:** C. t[0] = 10

(*Because tuples are immutable*)

---

7. What is the output?

python

CopyEdit

```
t = (10, 20, 30)
print(len(t))
```

A. 2

B. 3

C. 1

D. Error

✅ **Answer:** B. 3

---

8. Which method returns the index of the first occurrence of a value?

A. find()

B. index()

C. search()

D. locate()

✅ **Answer:** B. index()

---

9. Can a tuple contain elements of different data types?
A. No
B. Yes
C. Only strings
D. Only numbers
✅ **Answer:** B. Yes

---

10. What is the output of the following code?
python
CopyEdit
t1 = (1, 2)
t2 = (3, 4)
print(t1 + t2)
A. [1, 2, 3, 4]
B. (1, 2, 3, 4)
C. [(1, 2), (3, 4)]
D. Error
✅ **Answer:** B. (1, 2, 3, 4)

MCQs on Dictionaries in Python

1. Which of the following correctly defines a dictionary in Python?
A. d = [1: "one", 2: "two"]
B. d = (1, "one", 2, "two")
C. d = {1: "one", 2: "two"}
D. d = <1: "one", 2: "two">
✅ **Answer:** C. d = {1: "one", 2: "two"}

---

2. What is the output of this code?
python
CopyEdit
d = {"a": 1, "b": 2}
print(d["b"])
A. "b"
B. 1
C. 2
D. KeyError
✅ **Answer:** C. 2

---

3. What is the data type of a dictionary key?
A. Only strings
B. Only integers
C. Any immutable type
D. Any type
✅ **Answer:** C. Any immutable type

---

4. Which method returns all the keys from a dictionary?
A. keys()
B. get()
C. values()
D. items()
✅ **Answer:** A. keys()

---

5. Which method is used to get the value of a specified key, returning None if the key doesn't exist?
A. index()
B. get()
C. find()
D. search()
✅ **Answer:** B. get()

---

6. What will be the output of this code?
python
CopyEdit
```
d = {"x": 10}
d["x"] = 20
print(d)
```
A. {"x": 10}
B. {"x": 20}
C. {"x": 10, "x": 20}
D. Error
✅ **Answer:** B. {"x": 20}

---

7. Which method removes all items from a dictionary?
A. delete()
B. pop()
C. clear()
D. remove()
✅ **Answer:** C. clear()

---

8. What does d.items() return?
A. A list of values
B. A list of keys
C. A view of (key, value) pairs
D. A list of indices
✅ **Answer:** C. A view of (key, value) pairs

---

9. How can you check if a key exists in a dictionary?
A. key in dict
B. dict.has_key(key)
C. key.exists(dict)
D. dict.contains(key)
✅ **Answer:** A. key in dict

---

10. What is the output of this code?
python
CopyEdit
d = {"a": 1, "b": 2}
print(d.pop("a"))
A. "a"
B. 1
C. 2
D. None
✅ **Answer:** B. 1

MCQs on Dictionary Methods in Python

1. What does the get() method do in a dictionary?
A. Gets all values in the dictionary
B. Returns the value for a key, or a default value if the key is not found
C. Gets all keys in the dictionary
D. Raises an error if the key is not found
✅ **Answer:** B. Returns the value for a key, or a default value if the key is not found

---

2. What is the output of this code?
python
CopyEdit
d = {"a": 1, "b": 2}
print(d.get("c", 0))
A. None
B. KeyError
C. 0
D. "c"
✅ **Answer:** C. 0

---

3. Which method adds key-value pairs from one dictionary to another?
A. merge()
B. add()
C. append()
D. update()
✅ **Answer:** D. update()

---

4. What does the keys() method return?
A. A list of all values
B. A view object of all keys
C. A tuple of all keys
D. A sorted list of keys
✅ **Answer:** B. A view object of all keys

---

5. Which dictionary method is used to remove a key and return its value?
A. del()
B. remove()
C. pop()
D. clear()
✅ **Answer:** C. pop()

---

6. What does popitem() do?
A. Removes a specific item
B. Removes and returns an arbitrary key-value pair (Python < 3.7) or the last inserted one (Python ≥ 3.7)
C. Clears all items
D. Removes a value by index
✅ **Answer:** B. Removes and returns an arbitrary key-value pair (pre-3.7) or the last inserted one (3.7+)

---

7. Which of the following methods returns a view of dictionary's key-value pairs?
A. values()
B. keys()
C. items()
D. get()
✅ **Answer:** C. items()

---

8. What is the output of this code?
python
CopyEdit
```
d = {"x": 100}
d.setdefault("y", 200)
print(d)
```
A. {"x": 100}
B. {"x": 100, "y": 200}
C. {"y": 200}
D. Error
✅ **Answer:** B. {"x": 100, "y": 200}

---

9. What does the clear() method do?
A. Deletes the dictionary
B. Removes the first item
C. Removes all key-value pairs
D. Removes only None values
✅ **Answer:** C. Removes all key-value pairs

---

10. Which method returns a list of all the values in a dictionary?
A. get()
B. keys()
C. values()
D. items()
✅ **Answer:** C. values()

MCQs: Iterating Over a Dictionary in Python

---

1. Which method allows you to iterate over both keys and values of a dictionary?
A. values()
B. keys()
C. items()
D. get()
✅ **Answer:** C. items()

---

2. What is the output of this code?
python
CopyEdit

```
d = {"a": 1, "b": 2}
for key in d:
    print(key)
```

A. 1 2
B. a b
C. ('a', 1), ('b', 2)
D. None
✅ **Answer:** B. a b
(□*Iterating directly over a dictionary gives the keys*)

---

3. Which of the following correctly iterates over dictionary values only?
A. for val in d.items():
B. for val in d.keys():
C. for val in d.values():
D. for val in d.get():
✅ **Answer:** C. for val in d.values():

---

4. What is the output of this code?

python
CopyEdit

```
d = {"x": 10, "y": 20}
for k, v in d.items():
    print(k, v)
```

A. x y
B. 10 20
C. ('x', 10), ('y', 20)
D.
nginx
CopyEdit

```
x 10
y 20
```

✅ **Answer:** D.
nginx
CopyEdit

```
x 10
y 20
```

---

5. Which loop correctly prints all keys in a dictionary?
A. for k in d.values()
B. for k, v in d:
C. for k in d:
D. for k in d.items():
✅ **Answer:** C. for k in d:

---

6. Which of the following will raise an error?
A. for key in d:
B. for key in d.keys():
C. for val in d.values():
D. for k, v in d:
✅ **Answer:** D. for k, v in d:
(☐*This needs .items() to unpack into key and value*)

---

7. What is the correct way to iterate through both key and value using indexing?
A. for i in range(len(d)):
B. for k in d:
C. for k, v in d.items():
D. for k in d.get():
✅ **Answer:** C. for k, v in d.items():

---

8. What type is returned by d.items()?
A. list
B. tuple
C. dict_items
D. set
✅ **Answer:** C. dict_items
(☐*It's a view object*)

---

9. What is the output of the following code?
python
CopyEdit
```
d = {"one": 1, "two": 2}
for i in d.items():
    print(i)
```
A. one 1 two 2
B. (one, 1), (two, 2)
C.
bash
CopyEdit
```
('one', 1)
('two', 2)
```
D. Error
✅ **Answer:** C.
bash
CopyEdit
```
('one', 1)
('two', 2)
```

---

10. Which method would you use to iterate through keys in insertion order (Python 3.7+)?
A. sorted(d.keys())
B. d.values()
C. d.items()
D. Direct iteration using for key in d
✅ **Answer:** D. Direct iteration using for key in d
(☐*Since Python 3.7+, dicts maintain insertion order by default*)

MCQs on Sets in Python

1. What is a set in Python?
A. An ordered collection of items
B. A mutable, unordered collection with no duplicate elements
C. A dictionary with unique keys
D. An immutable collection of duplicates
✅ **Answer:** B. A mutable, unordered collection with no duplicate elements

---

2. Which of the following is a valid way to create a set?
A. s = []
B. s = {}
C. s = set([1, 2, 3])
D. s = dict()
✅ **Answer:** C. s = set([1, 2, 3])
(□{} *creates an empty* ***dict****, not a set.*)

---

3. What will be the output of this code?
python
CopyEdit
s = {1, 2, 2, 3}
print(len(s))
A. 4
B. 3
C. 2
D. Error
✅ **Answer:** B. 3
(□*Duplicate 2 is removed in a set*)

---

4. Which of the following operations is used to add a single element to a set?
A. add()
B. append()
C. extend()
D. insert()
✅ **Answer:** A. add()

---

5. Which method removes and returns a random element from a set?
A. remove()
B. discard()
C. pop()
D. del()
✅ **Answer:** C. pop()

---

6. Which method is used to remove an element without raising an error if it's not present?
A. remove()
B. discard()
C. pop()
D. del()
✅ **Answer:** B. discard()

---

7. What is the result of the following code?
python
CopyEdit
s1 = {1, 2, 3}
s2 = {3, 4, 5}
print(s1 & s2)
A. {1, 2, 3, 4, 5}
B. {1, 2}
C. {3}
D. Error
✅ **Answer:** C. {3}
(☐*& is the intersection operator*)

---

8. Which operator is used for union of two sets?
A. +
B. |
C. &
D. ^
✅ **Answer:** B. |

---

9. Which of the following is not a valid set operation in Python?
A. union()
B. intersection()
C. combine()
D. difference()
✅ **Answer:** C. combine()

---

10. What is the output?
python
CopyEdit
s = set("hello")
print(s)
A. {'hello'}
B. ['h', 'e', 'l', 'l', 'o']
C. A set of unique characters from the string
D. Error
✅ **Answer:** C. A set of unique characters from the string

MCQs on Unique Elements in Sets in Python

1. What will be the output of this code?

python
CopyEdit
s = {1, 2, 2, 3, 4, 4}
print(s)

A. {1, 2, 2, 3, 4}
B. {1, 2, 3, 4}
C. {1, 2, 2, 4, 3}
D. {1, 2, 3, 4, 4}

✅ **Answer:** B. {1, 2, 3, 4}

(*Sets automatically remove duplicate elements*)

---

2. Which of the following operations can be used to remove duplicate elements from a list in Python?

A. list()
B. set()
C. dict()
D. sorted()

✅ **Answer:** B. set()

(*Converting a list to a set removes duplicates since sets only store unique elements*)

---

3. What is the output of the following code?

python
CopyEdit
s = set([1, 2, 2, 3, 4, 4, 5])
s.add(5)
print(s)

A. {1, 2, 2, 3, 4, 4, 5}
B. {1, 2, 3, 4, 5, 5}
C. {1, 2, 3, 4, 5}
D. Error

✅ **Answer:** C. {1, 2, 3, 4, 5}

(*Sets maintain uniqueness, so adding a duplicate element does not change the set*)

---

4. Which of the following will ensure only unique elements are in a list?

A. list(set(list))
B. set(list)
C. unique(list)
D. remove_duplicates(list)

✅ **Answer:** A. list(set(list))

(*Converting the list to a set removes duplicates, then converting it back to a list*)

---

5. What is the result of this code?

python
CopyEdit
s = set("apple")
print(s)

A. {'a', 'p', 'l', 'e'}
B. {'a', 'p', 'p', 'l', 'e'}
C. {'apple'}
D. Error

✅ **Answer:** A. {'a', 'p', 'l', 'e'}

(☐*The string "apple" is converted to a set, which removes duplicates and retains unique characters*)

---

6. How does a set in Python handle duplicate elements when added?

A. It stores the element multiple times
B. It only stores the first occurrence of the element
C. It raises a ValueError
D. It overwrites the previous occurrence of the element

✅ **Answer:** B. It only stores the first occurrence of the element

---

7. Which of the following will correctly return a set of unique elements from a list?

A. set([1, 2, 2, 3, 4])
B. list([1, 2, 2, 3, 4])
C. set(list([1, 2, 2, 3, 4]))
D. list(set([1, 2, 2, 3, 4]))

✅ **Answer:** D. list(set([1, 2, 2, 3, 4]))

---

8. What will be the result of this operation on a set?

python
CopyEdit
s = {1, 2, 3}
s.add(2)
print(s)

A. {1, 2, 2, 3}
B. {1, 2, 3}
C. {2, 3, 1, 2}
D. {2, 3, 1}

✅ **Answer:** B. {1, 2, 3}

(☐*Adding a duplicate element does not change the set, as it only keeps unique elements*)

---

9. What is the output of this code?
python
CopyEdit
```
s = set([1, 1, 2, 3, 3, 4])
s.remove(3)
print(s)
```
A. {1, 2, 3, 4}
B. {1, 2, 4}
C. {1, 2, 3}
D. {1, 2, 3, 4, 3}
✅ **Answer:** B. {1, 2, 4}
(□*remove() removes the element if it exists, so 3 is removed*)

---

10. How can you ensure that a list contains only unique elements?
A. Use list(set(list))
B. Use set(list)
C. Use list()
D. Use list(unique(list))
✅ **Answer:** A. list(set(list))
(□*Converting a list to a set removes duplicates, then converting it back to a list gives a unique collection*)

MCQs on Frozenset in Python

1. What is a frozenset in Python?
A. A mutable set
B. An immutable set
C. A list that behaves like a set
D. A dictionary-like set
✅ **Answer:** B. An immutable set
(□*Frozensets are like sets, but they cannot be modified after creation*)

---

2. How do you create a frozenset from a list?
A. frozenset([1, 2, 3])
B. set([1, 2, 3])
C. frozenset{1, 2, 3}
D. list.frozenset(1, 2, 3)
✅ **Answer:** A. frozenset([1, 2, 3])

---

3. What is the main difference between a frozenset and a set in Python?
A. Frozensets allow duplicates, while sets do not
B. Frozensets are immutable, whereas sets are mutable
C. Frozensets are faster than sets
D. Sets are unordered, while frozensets are ordered
✅ **Answer:** B. Frozensets are immutable, whereas sets are mutable

---

4. Which of the following operations cannot be performed on a frozenset?
A. add()
B. remove()
C. discard()
D. pop()
✅ **Answer:** A. add()
(☐*Frozensets are immutable, so you cannot add new elements*)

---

5. What is the output of this code?
python
CopyEdit
```
fs = frozenset([1, 2, 3])
fs.add(4)
print(fs)
```
A. frozenset([1, 2, 3, 4])
B. frozenset([1, 2, 3])
C. Error
D. frozenset([4])
✅ **Answer:** C. Error
(☐*The add() method cannot be used on a frozenset because it is immutable*)

---

6. Which of the following is a valid operation for frozensets?
A. frozenset1.add(frozenset2)
B. frozenset1.remove(3)
C. frozenset1.update(frozenset2)
D. frozenset1 & frozenset2
✅ **Answer:** D. frozenset1 & frozenset2
(☐*Frozensets support set operations like intersection, union, and difference*)

---

7. What is the result of this operation on a frozenset?
python
CopyEdit
```
fs1 = frozenset([1, 2, 3])
fs2 = frozenset([3, 4, 5])
print(fs1 | fs2)
```
A. {1, 2, 3, 4, 5}
B. frozenset([1, 2, 3, 4, 5])
C. frozenset([3])
D. Error
✅ **Answer:** B. frozenset([1, 2, 3, 4, 5])
(☐*The | operator is for union, and the result is a frozenset*)

---

8. Which of the following is the correct way to create an empty frozenset?
A. frozenset()
B. frozenset([])
C. frozenset{}
D. set()
✅ **Answer:** A. frozenset()
(*Creating an empty frozenset requires calling frozenset() without any arguments*)

---

9. What will be the output of this code?
python
CopyEdit
fs = frozenset([1, 2, 3, 4])
print(len(fs))
A. 4
B. None
C. Error
D. frozenset([1, 2, 3, 4])
✅ **Answer:** A. 4
(*The length of a frozenset can be determined using len()*)

---

10. Can frozensets be used as keys in a dictionary?
A. Yes
B. No
C. Only if they are empty
D. Only if they are converted to sets
✅ **Answer:** A. Yes
(*Frozensets are immutable and hashable, so they can be used as dictionary keys*)

MCQs on Set Methods in Python

1. Which of the following methods adds an element to a set?
A. insert()
B. append()
C. add()
D. extend()
✅ **Answer:** C. add()

---

2. What does the remove() method do in a set?
A. Adds an element to the set
B. Removes an element from the set
C. Removes all elements from the set
D. Checks if an element is present in the set
✅ **Answer:** B. Removes an element from the set
(*remove() raises an error if the element is not found*)

---

3. What will be the output of the following code?

python
CopyEdit

```
s = {1, 2, 3}
s.add(4)
print(s)
```

A. {1, 2, 3, 4}
B. {1, 2, 3, 4, 4}
C. {1, 2, 3}
D. Error

✅ **Answer:** A. {1, 2, 3, 4}

(□*Sets automatically remove duplicates, so no second 4*)

---

4. Which method is used to remove a specified element from the set without raising an error if the element is not present?

A. discard()
B. remove()
C. pop()
D. clear()

✅ **Answer:** A. discard()

(□*discard() will not raise an error if the element doesn't exist in the set*)

---

5. Which method returns the number of elements in a set?

A. len()
B. count()
C. size()
D. length()

✅ **Answer:** A. len()

---

6. What is the output of the following code?

python
CopyEdit

```
s = {1, 2, 3}
s.remove(2)
print(s)
```

A. {1, 2, 3}
B. {1, 3}
C. {2, 3}
D. Error

✅ **Answer:** B. {1, 3}

(□*remove() removes the specified element if it exists*)

---

7. Which method is used to remove and return a random element from a set?
A. remove()
B. pop()
C. discard()
D. clear()
✅ **Answer:** B. pop()
(☐*pop() removes and returns an arbitrary element from the set*)

---

8. What does the clear() method do in a set?
A. Clears a single element from the set
B. Clears all elements from the set
C. Removes duplicates from the set
D. Prints the set
✅ **Answer:** B. Clears all elements from the set

---

9. What will be the output of the following code?
python
CopyEdit
```
s1 = {1, 2, 3}
s2 = {3, 4, 5}
print(s1.union(s2))
```
A. {1, 2, 3, 4, 5}
B. {3}
C. {1, 2, 3}
D. Error
✅ **Answer:** A. {1, 2, 3, 4, 5}
(☐*union() returns a set with all elements from both sets*)

---

10. Which method is used to get the intersection of two sets?
A. add()
B. union()
C. intersection()
D. difference()
✅ **Answer:** C. intersection()
(☐*intersection() returns the set of common elements between two sets*)

---

11. What is the output of the following code?
python
CopyEdit
```
s1 = {1, 2, 3}
s2 = {2, 3, 4}
print(s1.intersection(s2))
```
A. {1, 2, 3}
B. {2, 3}
C. {4}
D. {1, 4}
✅ **Answer:** B. {2, 3}
(☐*The intersection returns the elements common to both sets*)

---

12. What does the difference() method do in a set?
A. Returns the union of two sets
B. Returns elements in the first set but not in the second set
C. Returns the intersection of two sets
D. Adds elements to the set
✅ **Answer:** B. Returns elements in the first set but not in the second set

---

13. What is the result of this code?
python
CopyEdit
```
s1 = {1, 2, 3}
s2 = {2, 3, 4}
print(s1.difference(s2))
```
A. {1}
B. {2, 3}
C. {4}
D. {1, 2, 3, 4}
✅ **Answer:** A. {1}
(☐*The difference() method returns elements in s1 that are not in `s2*)

---

14. Which method checks if a set is a subset of another set?
A. issubset()
B. issuperset()
C. union()
D. intersection()
✅ **Answer:** A. issubset()
(☐*issubset() checks if all elements of one set are contained within another*)

---

15. What does the isdisjoint() method do in sets?
A. Checks if two sets are equal
B. Checks if there are no common elements between two sets
C. Adds a new element to the set
D. Removes duplicates from the set
✅ **Answer:** B. Checks if there are no common elements between two sets

MCQs on Modules in Python

1. What is a module in Python?
A. A function defined inside a script
B. A file containing Python definitions and statements
C. A type of variable
D. A class in Python
✅ **Answer:** B. A file containing Python definitions and statements
(☐*A module is a Python file that contains functions, variables, and classes*)

---

2. How do you import an entire module in Python?
A. import module_name
B. from module_name import *
C. import * from module_name
D. include module_name
✅ **Answer:** A. import module_name
(*You can import the whole module using import module_name*)

---

3. Which of the following is the correct syntax to import a specific function from a module?
A. import module_name.function_name
B. from module_name import function_name
C. from function_name import module_name
D. import function_name from module_name
✅ **Answer:** B. from module_name import function_name
(*This imports a specific function from the module into the current namespace*)

---

4. What is the result of importing a module in Python?
A. It loads the module and runs all the code inside it
B. It only loads the function names in the module
C. It adds a new class to the current script
D. It creates a copy of the module in memory
✅ **Answer:** A. It loads the module and runs all the code inside it
(*When a module is imported, Python executes all the code in the module*)

---

5. How do you rename a module during import?
A. import module_name as new_name
B. from module_name import new_name
C. import new_name from module_name
D. module_name.rename(new_name)
✅ **Answer:** A. import module_name as new_name
(*You can rename the module using the as keyword*)

---

6. What is the purpose of the dir() function in Python?
A. It lists the functions in a module
B. It lists all attributes of a module
C. It runs a script
D. It imports a module
✅ **Answer:** B. It lists all attributes of a module
(*The dir() function returns a list of valid attributes in the specified object or module*)

---

7. Which function is used to check if a module is already imported in Python?
A. is_imported()
B. is_module()
C. sys.modules
D. check_module()
✅ **Answer:** C. sys.modules
(*You can use sys.modules to check if a module is already imported in Python*)

---

8. How can you reload a module in Python after modifying it?
A. reload(module_name)
B. import module_name again
C. reimport module_name
D. import module_name.reload()
✅ **Answer:** A. reload(module_name)
(□*The reload() function from the importlib module can be used to reload a module*)

---

9. What is the __name__ variable used for in Python modules?
A. It stores the module's file path
B. It represents the module's name
C. It holds the list of all functions in the module
D. It holds the version of the module
✅ **Answer:** B. It represents the module's name
(□*The __name__ variable in a module holds its name when it's imported. If the module is run directly, it is set to "__main__"*)

---

10. How do you import a module that is located in another directory?
A. Use sys.path.append(directory_path) to add the directory to the Python path
B. Use import directory_path.module_name
C. Use from directory_path import module_name
D. Python can only import modules from the current directory
✅ **Answer:** A. Use sys.path.append(directory_path) to add the directory to the Python path
(□*You can add the directory containing the module to sys.path to import it*)

---

11. What is the purpose of the importlib module in Python?
A. It is used to import Python files into the current script
B. It is used to manage modules and provide functionalities like reloading them
C. It is used to define modules
D. It is used to compile Python files
✅ **Answer:** B. It is used to manage modules and provide functionalities like reloading them
(□*Importlib helps with dynamic module loading and reloading*)

---

12. Which of the following statements correctly imports the sqrt function from the math module?
A. import math.sqrt
B. import sqrt from math
C. from math import sqrt
D. from sqrt import math
✅ **Answer:** C. from math import sqrt
(□*This imports only the sqrt function from the math module*)

---

13. Which Python module allows you to interact with the operating system?
A. sys
B. os
C. time
D. datetime
✅ **Answer:** B. os
(□*The os module provides a way of interacting with the operating system*)

---

14. How can you find the file path of the currently executing script in Python?
A. os.path()
B. sys.path()
C. __file__
D. path()
✅ **Answer:** C. __file__
(□*The __file__ variable stores the path of the current script*)

---

15. How can you prevent a module from being run when imported?
A. Use the __main__ block
B. Use import as
C. Define the module's functions only
D. Use if __name__ == "__main__":
✅ **Answer:** D. Use if __name__ == "__main__":
(□*This condition ensures that code inside the block is only executed if the module is run directly, not when it is imported*)

MCQs on Types of Modules in Python

1. Which of the following is a built-in module in Python?
A. numpy
B. sys
C. requests
D. pandas
✅ **Answer:** B. sys
(□*sys is a built-in module in Python that provides access to some variables used or maintained by the Python interpreter*)

---

2. What is the correct way to import a module from Python's standard library?
A. import module_name
B. import module_name from library
C. import library.module_name
D. import module_name as library
✅ **Answer:** A. import module_name
(□*Modules from the Python standard library are imported by their module name directly*)

---

3. Which of the following is a custom user-defined module?
A. os
B. math
C. mymodule.py
D. sys
✅ **Answer:** C. mymodule.py
(□*A custom user-defined module is typically a .py file created by the user*)

---

4. How can you organize Python files into a module group?
A. By placing files in a folder and naming it with .py extension
B. By creating a __init__.py file inside a folder
C. By naming files with _
D. By using import for each file individually
✅ **Answer:** B. By creating a __init__.py file inside a folder
(□*A folder with an __init__.py file is treated as a package, which can contain multiple modules*)

---

5. What is the difference between a built-in module and a standard library module?
A. Built-in modules are part of the Python interpreter; standard library modules require installation
B. Standard library modules are part of the Python interpreter; built-in modules require installation
C. There is no difference; both are the same
D. Built-in modules are written in C, while standard library modules are written in Python
✅ **Answer:** A. Built-in modules are part of the Python interpreter; standard library modules require installation
(□*Built-in modules come bundled with the Python interpreter, while standard library modules are also part of Python but might need to be explicitly installed in some cases*)

---

6. Which of the following is NOT a type of module in Python?
A. Built-in modules
B. Standard library modules
C. Custom modules
D. External modules
E. Hidden modules
✅ **Answer:** E. Hidden modules
(□*There is no concept of "hidden modules" in Python; modules can be built-in, standard library, or custom*)

---

7. What type of module is the math module in Python?
A. Built-in module
B. Standard library module
C. External module
D. User-defined module
✅ **Answer:** B. Standard library module
(□*math is a standard library module, meaning it comes pre-installed with Python*)

---

8. Which of the following external modules must be installed separately using pip?
A. os
B. sys
C. requests
D. math
✅ **Answer:** C. requests
(☐requests is an external module that is not included in the standard library and must be installed separately using pip)

---

9. How can you check if a specific module is available in Python's standard library?
A. By using import module_name and checking for errors
B. By using the dir() function
C. By visiting Python's official documentation
D. By using sys.modules
✅ **Answer:** A. By using import module_name and checking for errors
(☐If a module is not available, Python will raise an ImportError when you try to import it)

---

10. Which of the following is a typical user-defined module?
A. my_script.py
B. math.py
C. sys.py
D. os.py
✅ **Answer:** A. my_script.py
(☐User-defined modules are typically Python files created by the user for specific purposes, like my_script.py)

---

11. How can you import a module from a subfolder (package) in Python?
A. import folder.module_name
B. from folder import module_name
C. import module_name from folder
D. Both A and B
✅ **Answer:** D. Both A and B
(☐Both methods work if the folder contains an __init__.py file and you wish to access the module inside)

---

12. Which of the following Python modules are considered "external" modules?
A. os
B. requests
C. time
D. math
✅ **Answer:** B. requests
(☐requests is an external module that is not bundled with Python and requires installation via pip)

---

13. What is the role of the __init__.py file in a folder?
A. It indicates that the folder is a regular folder, not a package
B. It makes the folder a Python package
C. It stores the package’s metadata
D. It stores functions and classes for the package
✅ **Answer:** B. It makes the folder a Python package
(□*The presence of __init__.py in a folder tells Python that the folder should be treated as a package*)

---

14. How do you import all functions and variables from a module?
A. import module_name*
B. import * from module_name
C. from module_name import *
D. include * from module_name
✅ **Answer:** C. from module_name import *
(□*This imports everything from the specified module into the current namespace*)

---

15. What type of module is datetime in Python?
A. Built-in module
B. Standard library module
C. External module
D. User-defined module
✅ **Answer:** B. Standard library module
(□*datetime is a standard library module, available by default in Python*)

MCQs on Arguments in Python

1. What are the types of arguments in Python?
A. Positional, Keyword, Default
B. Positional, Keyword, Variable-length
C. Positional, Keyword, Default, Variable-length
D. None of the above
✅ **Answer:** C. Positional, Keyword, Default, Variable-length
(□*Python supports different types of arguments like positional, keyword, default, and variable-length*)

---

2. Which of the following is a positional argument in Python?
A. def func(a, b):
B. def func(a=5, b=10):
C. def func(*args):
D. def func(a, *args):
✅ **Answer:** A. def func(a, b):
(□*Positional arguments are passed based on their position in the function call*)

---

3. What does the following function signature represent?

python

CopyEdit

```
def func(a, b, c=10):
```

A. a and b are required, c is an optional argument with a default value

B. a, b, and c are all optional arguments

C. a and c are optional arguments, b is required

D. b and c are required arguments, a is optional

✅ **Answer:** A. a and b are required, c is an optional argument with a default value

(□*c has a default value of 10, making it optional*)

---

4. Which of the following is an example of keyword arguments?

A. func(2, 4)

B. func(a=2, b=4)

C. func(2, b=4)

D. func(a=2, 4)

✅ **Answer:** B. func(a=2, b=4)

(□*Keyword arguments are passed by explicitly naming the parameter and its value*)

---

5. What will be the output of the following code?

python

CopyEdit

```
def greet(name, age=25):
    print(f"Hello, {name}! You are {age} years old.")

greet("Alice", 30)
```

A. Hello, Alice! You are 25 years old.

B. Hello, Alice! You are 30 years old.

C. Hello, Alice!

D. Error

✅ **Answer:** B. Hello, Alice! You are 30 years old.

(□*Since the age argument is provided in the function call, it overrides the default value*)

---

6. Which type of argument allows you to pass a variable number of arguments to a function?

A. Positional argument

B. Keyword argument

C. Default argument

D. Variable-length argument

✅ **Answer:** D. Variable-length argument

(□*Variable-length arguments allow you to pass a dynamic number of arguments, using *args for positional or **kwargs for keyword arguments*)

---

7. What is the purpose of *args in Python?
A. To pass a fixed number of arguments to a function
B. To pass a list of arguments as one argument
C. To accept a variable number of positional arguments
D. To accept a variable number of keyword arguments
✅ **Answer:** C. To accept a variable number of positional arguments
(□*args allows you to pass a variable number of positional arguments as a tuple*)

---

8. How do you pass a dictionary of keyword arguments to a function?
A. *args
B. **kwargs
C. args[]
D. kwargs[]
✅ **Answer:** B. **kwargs
(□***kwargs allows you to pass a dictionary of keyword arguments to a function*)

---

9. What is the correct way to define a function that accepts both positional arguments and variable-length keyword arguments?
A. def func(a, *args, **kwargs):
B. def func(*args, **kwargs, a):
C. def func(a, **args, *kwargs):
D. def func(a, *kwargs):
✅ **Answer:** A. def func(a, *args, **kwargs):
(□*You can use *args for variable-length positional arguments and **kwargs for variable-length keyword arguments*)

---

10. What will be the output of the following code?
python
CopyEdit

```
def func(a, b=10, c=20):
    print(a, b, c)

func(5, c=30)
```

A. 5 10 20
B. 5 20 30
C. 5 30 20
D. Error
✅ **Answer:** B. 5 20 30
(□*b takes the default value, while c is overridden by the value passed in the function call*)

---

11. Which of the following functions is an example of using both positional and keyword arguments?
A. def func(*args, a, b):
B. def func(a, b=10, *args):
C. def func(a, *args, b):
D. def func(a, b, c=10):
✅ **Answer:** D. def func(a, b, c=10):
(□*a and b are positional, and c is a default argument with a value*)

12. What is the output of the following code?
python
CopyEdit
```
def func(a, *args, b=10):
    print(a, args, b)

func(5, 6, 7, b=20)
```
A. 5 (6, 7) 10
B. 5 (6, 7) 20
C. 5 6 7 20
D. Error
✅ **Answer:** B. 5 (6, 7) 20
(☐*args captures the positional arguments (6, 7) as a tuple, and b is passed as a keyword argument with value `20*)

---

13. Which type of argument is used when you want to provide a value that can be overridden?
A. Positional argument
B. Keyword argument
C. Default argument
D. Variable-length argument
✅ **Answer:** C. Default argument
(☐*Default arguments are assigned a value but can be overridden when explicitly passed in a function call*)

---

14. What is the correct way to call a function with both default and non-default arguments?
A. Pass non-default arguments first, then default arguments
B. Pass default arguments first, then non-default arguments
C. Pass default arguments only
D. Both can be passed in any order
✅ **Answer:** A. Pass non-default arguments first, then default arguments
(☐*When calling a function, non-default arguments must be passed before default ones*)

---

15. Which of the following will cause an error in Python?
A. func(5, 10) when def func(a, b=10):
B. func(a=5, 10) when def func(a, b):
C. func(a=5) when def func(a, b=10):
D. func(5, b=10) when def func(a, b=10):
✅ **Answer:** B. func(a=5, 10) when def func(a, b):
(☐*You cannot mix positional and keyword arguments in this order. Keyword arguments must appear after positional arguments*).

MCQs on Exception Handling in Python

1. Which of the following keywords is used to catch exceptions in Python?
A. throw
B. except
C. catch
D. raise
✅ **Answer:** B. except
(☐*The except keyword is used to catch and handle exceptions in Python*)

---

2. What is the purpose of the finally block in Python?
A. To handle exceptions
B. To execute code whether or not an exception occurred
C. To define the exception type
D. To raise exceptions
✅ **Answer:** B. To execute code whether or not an exception occurred
(☐*The finally block is executed no matter what, whether an exception occurs or not*)

---

3. What will the following code output?
python
CopyEdit

```
try:
    x = 1 / 0
except ZeroDivisionError:
    print("Cannot divide by zero")
finally:
    print("This will always execute")
```

A. Cannot divide by zero
B. This will always execute
C. Cannot divide by zero
This will always execute
D. Error
✅ **Answer:** C. Cannot divide by zero
This will always execute
(☐*The except block will handle the ZeroDivisionError, and the finally block will always execute*)

---

4. Which of the following is the correct syntax to raise an exception manually in Python?
A. raise Exception("An error occurred")
B. throw Exception("An error occurred")
C. raise("An error occurred")
D. raise("Exception")
✅ **Answer:** A. raise Exception("An error occurred")
(☐*The raise keyword is used to manually raise an exception in Python*)

---

5. Which of the following exceptions is raised when a variable or function is used before it is assigned a value?
A. TypeError
B. IndexError
C. NameError
D. ValueError
✅ **Answer:** C. NameError
(☐*A NameError occurs when a local or global name is not found*)

---

6. What will be the output of the following code?
python
CopyEdit
```
try:
   x = int("abc")
except ValueError:
   print("ValueError occurred")
except Exception:
   print("General Exception occurred")
```
A. ValueError occurred
B. General Exception occurred
C. Error
D. None
✅ **Answer:** A. ValueError occurred
(☐*The ValueError occurs because the string "abc" cannot be converted to an integer*)

---

7. How can you catch multiple exceptions in a single except block?
A. except (ValueError, ZeroDivisionError):
B. except ValueError or ZeroDivisionError:
C. except ValueError, ZeroDivisionError:
D. except (ValueError, ZeroDivisionError) as e:
✅ **Answer:** A. except (ValueError, ZeroDivisionError):
(☐*You can catch multiple exceptions by specifying them in parentheses, separated by commas*)

---

8. Which of the following statements is correct about the else block in exception handling?
A. It runs if no exception occurs in the try block
B. It runs when any exception occurs in the try block
C. It is always executed
D. It must always follow the finally block
✅ **Answer:** A. It runs if no exception occurs in the try block
(☐*The else block is executed if no exception is raised in the try block*)

---

9. What will the following code output?

python
CopyEdit

```
try:
   x = 10 / 0
except ZeroDivisionError:
   print("Zero Division Error")
else:
   print("No errors occurred")
```

A. Zero Division Error
B. No errors occurred
C. Zero Division Error
No errors occurred
D. Error

✅ **Answer:** A. Zero Division Error
(*The ZeroDivisionError will be raised, so the else block will not execute*)

---

10. Which of the following exceptions is raised when a non-existent file is opened in Python?

A. FileNotFoundError
B. IOError
C. ValueError
D. NameError

✅ **Answer:** A. FileNotFoundError
(*FileNotFoundError occurs when trying to open a file that does not exist*)

---

11. Which of the following statements is true about custom exceptions in Python?

A. Custom exceptions must inherit from the BaseException class
B. Custom exceptions can inherit from any class
C. Custom exceptions must inherit from the Exception class
D. Custom exceptions cannot be created

✅ **Answer:** C. Custom exceptions must inherit from the Exception class
(*Custom exceptions are typically created by subclassing the built-in Exception class*)

---

12. What will the following code output?

python
CopyEdit

```
try:
   raise ValueError("This is a custom error")
except ValueError as e:
   print(e)
```

A. ValueError
B. This is a custom error
C. This is a custom error ValueError
D. Error

✅ **Answer:** B. This is a custom error
(*The custom message passed to ValueError will be printed*)

---

13. What is the default behavior when an exception is not caught in Python?
A. Python terminates the program and displays an error message
B. Python ignores the exception and continues execution
C. Python tries to handle the exception automatically
D. Python raises a RuntimeError
✅ **Answer:** A. Python terminates the program and displays an error message
(☐*If an exception is not caught, Python will terminate the program and display an error traceback*)

---

14. Which of the following exceptions is raised when an invalid index is accessed in a list?
A. TypeError
B. IndexError
C. ValueError
D. KeyError
✅ **Answer:** B. IndexError
(☐*An IndexError occurs when trying to access an index that is out of range in a list*)

---

15. Which of the following statements is true about the raise keyword in Python?
A. raise is used to throw an exception
B. raise is used to handle exceptions
C. raise can only be used to throw built-in exceptions
D. raise automatically catches exceptions
✅ **Answer:** A. raise is used to throw an exception
(☐*The raise keyword is used to manually throw (raise) an exception in Python*)

MCQs on Base Exception Error in Python

1. What is the base class for all exceptions in Python?
A. BaseError
B. BaseException
C. Exception
D. Error
✅ **Answer:** B. BaseException
(☐*BaseException is the base class for all exceptions in Python. All built-in exceptions are derived from it*)

---

2. Which of the following exceptions is directly derived from BaseException?
A. Exception
B. SystemExit
C. TypeError
D. ValueError
✅ **Answer:** B. SystemExit
(☐*SystemExit, KeyboardInterrupt, and GeneratorExit are directly derived from BaseException*)

---

3. What is the purpose of BaseException in Python?
A. To handle all types of exceptions
B. To serve as the base class for all exceptions
C. To define the try-except block behavior
D. To represent system-related errors only
✅ **Answer:** B. To serve as the base class for all exceptions
(□*BaseException is the root class for all exceptions in Python, from which all other exceptions inherit*)

---

4. Which of the following exceptions is NOT derived from BaseException?
A. KeyError
B. TypeError
C. SystemExit
D. MemoryError
✅ **Answer:** D. MemoryError
(□*MemoryError is derived from Exception, not directly from BaseException*)

---

5. What happens if an exception is raised that is not derived from BaseException in Python?
A. Python will automatically handle it
B. Python will terminate the program
C. It will not be considered an exception
D. Python will ignore the error
✅ **Answer:** C. It will not be considered an exception
(□*All exceptions in Python must be derived from BaseException for Python to recognize them as exceptions*)

---

6. Which of the following is a direct subclass of BaseException?
A. StopIteration
B. ArithmeticError
C. OSError
D. ImportError
✅ **Answer:** A. StopIteration
(□*StopIteration is a direct subclass of BaseException, used to signal the end of an iterator*)

---

7. What will happen if you catch BaseException in a try-except block?
A. It will catch all exceptions, including system exit and keyboard interrupts
B. It will catch only runtime errors
C. It will catch only logical errors
D. It will cause an error in the program
✅ **Answer:** A. It will catch all exceptions, including system exit and keyboard interrupts
(□*Since BaseException is the root of all exceptions, catching it will catch every type of exception, including system-related exceptions like SystemExit*)

---

8. Which of the following is true about BaseException in Python?
A. BaseException can be directly raised
B. BaseException is the base for runtime exceptions only
C. BaseException can be inherited to create new exception classes
D. BaseException is used to exit a program
✅ **Answer:** C. BaseException can be inherited to create new exception classes
(☐*You can create custom exceptions by inheriting BaseException, though it's more common to inherit from Exception*)

---

9. What is the relationship between BaseException and Exception?
A. Exception is a subclass of BaseException
B. BaseException is a subclass of Exception
C. BaseException and Exception are independent classes
D. Exception is a class inside BaseException
✅ **Answer:** A. Exception is a subclass of BaseException
(☐*Exception inherits from BaseException and is the base class for most built-in exceptions*)

---

10. What is the result of the following code?
python
CopyEdit
```
try:
   raise BaseException("This is a base exception")
except BaseException as e:
   print(e)
```
A. This is a base exception
B. Error: This is a base exception
C. BaseException raised
D. BaseException is not caught
✅ **Answer:** A. This is a base exception
(☐*The code raises a BaseException, which is caught by the except block and prints the message*)

---

11. Which of the following is an example of using BaseException to create a custom exception class?
A. class MyError(BaseException): pass
B. class MyError(Exception): pass
C. class MyError(SystemExit): pass
D. class MyError(StopIteration): pass
✅ **Answer:** A. class MyError(BaseException): pass
(☐*You can create custom exceptions by inheriting BaseException, but it's more common to inherit Exception*)

---

12. Which of the following exceptions are *not* directly derived from BaseException?
A. ValueError
B. TypeError
C. KeyboardInterrupt
D. SystemExit
✅ **Answer:** A. ValueError
(☐*ValueError is derived from Exception, which is a subclass of BaseException. It is not directly derived from BaseException*)

---

13. In which of the following scenarios would you most likely need to catch BaseException?
A. To catch general exceptions during program execution
B. To handle specific exceptions like TypeError or ValueError
C. To catch system-level exceptions such as SystemExit or KeyboardInterrupt
D. To handle runtime exceptions only
✅ **Answer:** C. To catch system-level exceptions such as SystemExit or KeyboardInterrupt
(☐*BaseException is the root class for all exceptions, including system-level exceptions like SystemExit and KeyboardInterrupt*)

---

14. How can you define a custom exception that is specifically designed to be used for system-related errors, inheriting from BaseException?
A. class SystemError(BaseException): pass
B. class SystemError(Exception): pass
C. class SystemError(KeyboardInterrupt): pass
D. class SystemError(SystemExit): pass
✅ **Answer:** A. class SystemError(BaseException): pass
(☐*You can create a custom exception inheriting from BaseException to design your own exceptions*)

---

15. Which of the following is NOT a valid use of BaseException in exception handling?
A. Catching all exceptions including system and user exceptions
B. Raising custom exceptions directly derived from BaseException
C. Handling exceptions that require manual termination of a program
D. Automatically catching only runtime errors
✅ **Answer:** D. Automatically catching only runtime errors
(☐*BaseException will catch all exceptions, not just runtime errors. It is too broad for just catching runtime exceptions*)

MCQs on The try Block in Python

1. What is the primary purpose of the try block in Python?
A. To execute code that might raise an exception
B. To define the exception class
C. To raise an exception manually
D. To log errors in the program
✅ **Answer:** A. To execute code that might raise an exception
(☐*The try block is used to enclose code that may potentially raise an exception*)

---

2. What will the following code output?
python
CopyEdit
```
try:
   x = 1 / 0
except ZeroDivisionError:
   print("Cannot divide by zero")
```
A. Cannot divide by zero
B. Error occurred
C. None
D. ZeroDivisionError
✅ **Answer:** A. Cannot divide by zero
(*The ZeroDivisionError is caught in the except block, and the message "Cannot divide by zero" is printed*)

---

3. What happens if an exception is raised inside a try block, but there is no corresponding except block?
A. The program continues to execute normally
B. The program crashes immediately
C. A finally block will always handle the exception
D. A traceback will be displayed, and the program will stop
✅ **Answer:** D. A traceback will be displayed, and the program will stop
(*If there is no except block to catch the exception, Python will display a traceback and terminate the program*)

---

4. Which of the following is the correct way to use the try block?
A. try { code } except { exception }
B. try: code except: exception
C. try: code except exception as e:
D. try exception: code except:
✅ **Answer:** C. try: code except exception as e:
(*This is the correct syntax for using the try block along with the except block to catch exceptions*)

---

5. Can you have multiple except blocks following a single try block?
A. No, you can only have one except block
B. Yes, you can have one except block for each exception type
C. No, the except block must always handle all types of exceptions
D. Yes, but the program will run much slower with multiple except blocks
✅ **Answer:** B. Yes, you can have one except block for each exception type
(*Multiple except blocks can be used to catch different types of exceptions raised within the try block*)

---

6. What is the result of the following code?

python
CopyEdit

```
try:
   print("Hello")
   x = 1 / 0
   print("World")
except ZeroDivisionError:
   print("Error in division")
finally:
   print("Always executed")
```

A. Hello Error in division Always executed
B. Hello World Always executed
C. Hello World Error in division
D. Error in division Always executed

✅ **Answer:** A. Hello Error in division Always executed

(□*The ZeroDivisionError occurs after printing "Hello", but the finally block will always execute, printing "Always executed"*).

---

7. What is the result if there is no except block to handle an exception in the try block?

A. The program continues executing
B. The finally block executes, and the program continues
C. The program crashes, and a traceback is shown
D. Python silently ignores the exception

✅ **Answer:** C. The program crashes, and a traceback is shown

(□*If there is no except block to handle an exception, Python will raise an error and stop execution, displaying a traceback*)

---

8. Which of the following types of code can be placed inside the try block?

A. Only code that raises exceptions
B. Code that might raise exceptions
C. Code for debugging purposes
D. Only function calls

✅ **Answer:** B. Code that might raise exceptions

(□*The try block should enclose code that may potentially raise exceptions during execution*)

---

9. Which of the following is a correct way to handle an exception and print the error message using the try block?

A. try: code except Exception: print(error)
B. try: code except Exception as e: print(e)
C. try: code except Exception as e: print(error)
D. try: code except: print(Exception)

✅ **Answer:** B. try: code except Exception as e: print(e)

(□*The correct way to handle an exception and print its message is to use except Exception as e: and then print e*)

---

10. Which of the following is NOT a valid Python exception type that can be caught in a try block?

A. ValueError
B. ZeroDivisionError
C. TypeError
D. loopError

✅ **Answer:** D. loopError

(☐*loopError is not a built-in exception in Python. All other options are valid Python exception types*)

---

11. In Python, can the try block be used without an except block?

A. Yes, but it will only execute the finally block
B. No, a try block must always have an except block
C. Yes, but you must have a finally block
D. Yes, but it will raise an error if no except block is present

✅ **Answer:** A. Yes, but it will only execute the finally block

(☐*A try block can be used with only a finally block, but if no except block is present, only the finally block will be executed after the try block*)

---

12. What will the following code output?

python
CopyEdit

```
try:
    raise ValueError("An error occurred")
except ValueError as e:
    print("Caught:", e)
```

A. An error occurred
B. Caught: ValueError
C. Caught: An error occurred
D. Error occurred

✅ **Answer:** C. Caught: An error occurred

(☐*The ValueError is raised and caught in the except block, printing the message associated with the exception*)

---

13. Can you use try without an else block in Python?

A. No, else is required when using try
B. Yes, else is optional
C. No, the else block must always be included
D. Yes, but the program will not handle exceptions correctly

✅ **Answer:** B. Yes, else is optional

(☐*The else block is optional in a try-except statement. It is executed only if no exception is raised in the try block*)

---

14. What happens if there is an exception in the finally block?
A. The program will ignore the exception
B. The exception in the finally block will be suppressed
C. The exception will propagate after the finally block is executed
D. The program will stop immediately
✅ **Answer:** C. The exception will propagate after the finally block is executed
(☐*If an exception occurs in the finally block, it will propagate after the finally block finishes executing*)

---

15. What is the proper syntax for handling exceptions in Python using try, except, and else blocks?
A. try: code except Exception as e: handle exception else: no exception occurred
B. try: code except: handle exception else: code
C. try: code except Exception else: code
D. try: code except Exception: handle exception else: code
✅ **Answer:** D. try: code except Exception: handle exception else: code
(☐*This syntax ensures that exceptions are handled in the except block and code after try executes normally in the else block*)

MCQs on The except Block in Python

1. What is the purpose of the except block in Python?
A. To execute code that might raise an exception
B. To handle exceptions that occur in the try block
C. To define the type of exception
D. To exit the program after an error occurs
✅ **Answer:** B. To handle exceptions that occur in the try block
(☐*The except block is used to catch and handle exceptions that occur within the try block*)

---

2. What will the following code output?
python
CopyEdit

```
try:
    x = 1 / 0
except ZeroDivisionError:
    print("Error: Division by zero")
```

A. Error: Division by zero
B. ZeroDivisionError
C. None
D. Error: Invalid division
✅ **Answer:** A. Error: Division by zero
(☐*The exception ZeroDivisionError is caught by the except block, and the message is printed*)

---

3. Which of the following is a valid way to catch multiple exceptions in a single except block?

A. except (TypeError, ValueError):

B. except TypeError or ValueError:

C. except (TypeError, ValueError) as e:

D. except TypeError or ValueError as e:

✅ **Answer:** A. except (TypeError, ValueError):

(☐*You can catch multiple exceptions by using a tuple of exception types in the except block*)

---

4. What is the result of the following code?

python

CopyEdit

```
try:
   x = 1 / 0
except ValueError:
   print("Caught ValueError")
except ZeroDivisionError:
   print("Caught ZeroDivisionError")
```

A. Caught ValueError

B. Caught ZeroDivisionError

C. ZeroDivisionError

D. The program crashes

✅ **Answer:** B. Caught ZeroDivisionError

(☐*The ZeroDivisionError is raised and caught by the corresponding except block*)

---

5. What is the effect of the else block in the try-except structure?

A. It is executed if an exception occurs in the try block

B. It is executed only if no exception occurs in the try block

C. It is executed no matter what

D. It is used to handle specific exceptions

✅ **Answer:** B. It is executed only if no exception occurs in the try block

(☐*The else block runs only if no exception is raised in the try block*)

---

6. How would you capture the exception object in the except block for later use?

A. except Exception as e:

B. except Exception:

C. except as e:

D. except Exception as e: print(e)

✅ **Answer:** A. except Exception as e:

(☐*You can capture the exception object by using as e in the except block to refer to the exception instance*)

---

7. Which of the following code is used to catch a specific exception and print its message in the except block?

A. except Exception as e: print(e)

B. except e as Exception: print(e)

C. except Exception as e: print(Exception)

D. except Exception: print(e)

✅ **Answer:** A. except Exception as e: print(e)

(*This code catches the exception and prints the message associated with it*)

---

8. Can you catch all exceptions in a generic way without specifying the type in the except block?

A. Yes, by using except:

B. No, you must specify the exception type

C. Yes, by using except AllExceptions:

D. Yes, by using except Exception as e:

✅ **Answer:** A. Yes, by using except:

(*except: catches all exceptions, but it is not recommended because it can make debugging difficult*)

---

9. What happens if you catch an exception in the except block and re-raise it using raise?

A. The program will terminate immediately

B. The exception is caught and the program continues

C. The exception will propagate up to the outer scope

D. The exception will be suppressed and not shown

✅ **Answer:** C. The exception will propagate up to the outer scope

(*Re-raising the exception using raise after it has been caught will propagate the exception to the outer try-except block or terminate the program if not caught*)

---

10. What will the following code output?

python

CopyEdit

```
try:
    print("Before error")
    raise ValueError("Custom error message")
except ValueError as e:
    print("Caught an error:", e)
finally:
    print("This will always execute")
```

A. Before error Caught an error: Custom error message This will always execute

B. Before error This will always execute

C. Caught an error: Custom error message This will always execute

D. Error occurred This will always execute

✅ **Answer:** A. Before error Caught an error: Custom error message This will always execute

(*The ValueError is raised, caught, and the message is printed. The finally block always executes afterward*)

---

11. Which of the following is the correct way to catch multiple different types of exceptions in separate except blocks?
A. except ValueError: except TypeError:
B. except (ValueError, TypeError):
C. except ValueError: except TypeError as e:
D. except ValueError as e: except TypeError as e:
✅ **Answer:** D. except ValueError as e: except TypeError as e:
(□*You can catch different exceptions in separate except blocks by handling each one independently*)

---

12. How would you handle a FileNotFoundError exception when trying to open a file in Python?
A. except FileNotFoundError as e:
B. except IOError:
C. except FileNotFoundError:
D. except:
✅ **Answer:** A. except FileNotFoundError as e:
(□*You can catch a FileNotFoundError and handle it by specifying it in the except block*)

---

13. What happens if the except block does not match the type of exception raised?
A. The else block will execute
B. The exception will be ignored
C. The finally block will execute, but the program will continue running
D. The exception will propagate and terminate the program
✅ **Answer:** D. The exception will propagate and terminate the program
(□*If no except block matches the raised exception, it will propagate and cause the program to terminate*)

---

14. What is the correct way to catch both IndexError and TypeError in the same except block?
A. except (IndexError, TypeError):
B. except IndexError, TypeError:
C. except (IndexError, TypeError) as e:
D. except (IndexError or TypeError):
✅ **Answer:** A. except (IndexError, TypeError):
(□*You can catch multiple exceptions in a single except block by using a tuple*)

---

15. Can you handle exceptions of a specific type in the except block and also capture the exception message?
A. Yes, by using except Exception as e:
B. No, the exception message cannot be captured
C. Yes, but only for system exceptions
D. Yes, by using except without specifying the exception
✅ **Answer:** A. Yes, by using except Exception as e:
(□*You can capture the exception message by using as e in the except block, allowing you to access the exception object*)

MCQs on The else Block in Python

1. What is the purpose of the else block in Python?
A. To handle exceptions raised in the try block
B. To define the exception type
C. To execute code only if no exception occurs in the try block
D. To exit the program after handling an exception
✅ **Answer:** C. To execute code only if no exception occurs in the try block
(□*The else block is executed when no exception occurs in the try block*)

---

2. Which block is executed when an exception is not raised in the try block?
A. finally block
B. else block
C. except block
D. The program will stop execution
✅ **Answer:** B. else block
(□*The else block runs only if no exception is raised in the try block*)

---

3. What will the following code output?
python
CopyEdit

```
try:
   x = 5 / 1
except ZeroDivisionError:
   print("Cannot divide by zero")
else:
   print("Division was successful")
```

A. Cannot divide by zero
B. Division by zero was successful
C. Division was successful
D. Error occurred
✅ **Answer:** C. Division was successful
(□*Since no exception occurs in the try block, the else block is executed and prints "Division was successful"*)

---

4. What happens if an exception occurs inside the try block, but there is an else block?
A. The else block will be executed
B. The else block will be skipped
C. The program will stop execution
D. The else block will execute after the exception is handled in the except block
✅ **Answer:** B. The else block will be skipped
(□*If an exception is raised in the try block, the else block is not executed*)

---

5. Which of the following is the correct syntax for using the else block in exception handling?
A. try: code else: code
B. try: code except: code else: code
C. try: code except: code finally: code else: code
D. try: code except exception else: code
✅ **Answer:** B. try: code except: code else: code
(☐*This is the correct syntax for a try-except-else structure in Python*)

---

6. In which scenario will the else block not execute?
A. When an exception is raised in the try block
B. When there is no finally block
C. When the exception is caught in the except block
D. When the code in the try block is commented out
✅ **Answer:** A. When an exception is raised in the try block
(☐*The else block only executes if no exception is raised in the try block*)

---

7. What will the following code output?
python
CopyEdit
```
try:
    x = 1 / 0
except ZeroDivisionError:
    print("Caught ZeroDivisionError")
else:
    print("No exception occurred")
```
A. Caught ZeroDivisionError
B. No exception occurred
C. Error occurred
D. ZeroDivisionError
✅ **Answer:** A. Caught ZeroDivisionError
(☐*Since a ZeroDivisionError occurs in the try block, the except block is executed and "Caught ZeroDivisionError" is printed. The else block is skipped*)

---

8. Which of the following is true about the else block in Python?
A. It is executed if an exception occurs in the try block
B. It is executed if the try block has no exceptions and the except block is not executed
C. It is executed no matter what happens in the try block
D. It is used to raise exceptions manually
✅ **Answer:** B. It is executed if the try block has no exceptions and the except block is not executed
(☐*The else block executes only if no exception is raised in the try block and the except block is skipped*)

---

9. Can the else block be used without a try block?
A. Yes, it can be used independently
B. No, it must follow a try block
C. Yes, but it will only work with finally blocks
D. No, it is not valid in Python
✅ **Answer:** B. No, it must follow a try block
(*The else block must be used in conjunction with a try-except block. It cannot be used independently*)

---

10. What happens if there is an exception in the try block and there is no except block to catch it, but there is an else block?
A. The else block will execute
B. The program will crash, and the else block will not execute
C. The else block will catch the exception
D. The program will continue, skipping the exception
✅ **Answer:** B. The program will crash, and the else block will not execute
(*If an exception occurs in the try block and there is no except block to handle it, the else block will be skipped, and the program will crash*)

---

11. How many times is the else block executed in a try-except-else structure?
A. Always once, even if an exception occurs
B. Never, since it's an optional block
C. Once, only if no exception occurs in the try block
D. It's executed multiple times, depending on the exception
✅ **Answer:** C. Once, only if no exception occurs in the try block
(*The else block runs only once if the try block executes without raising an exception*)

---

12. Can the else block raise an exception?
A. No, the else block cannot raise an exception
B. Yes, the else block can raise exceptions like any other block
C. Yes, but only in the case of a custom exception
D. No, but the finally block can raise an exception
✅ **Answer:** B. Yes, the else block can raise exceptions like any other block
(*The else block can raise an exception, just like any other block of code in Python*)

---

13. What will be the output of the following code?
python
CopyEdit
```
try:
   result = 10 / 2
except ZeroDivisionError:
   print("Division by zero")
else:
   print("The result is:", result)
```
A. Division by zero
B. The result is: 5.0
C. Error occurred
D. ZeroDivisionError
✅ **Answer:** B. The result is: 5.0
(□*Since no exception occurs, the else block is executed, and the result of the division is printed*)

---

14. What happens if an exception occurs in the try block, but there is no except block to handle it and an else block is present?
A. The else block will be executed
B. The else block will execute after the exception is raised
C. The else block will be skipped, and the exception will terminate the program
D. The program will continue execution, and the exception will be ignored
✅ **Answer:** C. The else block will be skipped, and the exception will terminate the program
(□*If an exception occurs and there is no except block, the else block is skipped, and the exception will terminate the program*)

---

15. How does the else block improve the readability of exception handling in Python?
A. It allows all exceptions to be caught in one place
B. It provides a clear structure for when there are no exceptions
C. It is used to handle specific exceptions
D. It prevents the program from crashing
✅ **Answer:** B. It provides a clear structure for when there are no exceptions
(□*The else block clearly separates the code that runs when there are no exceptions from the code that handles exceptions, improving readability and organization*)

MCQs on The finally Block in Python

1. What is the purpose of the finally block in Python?
A. To handle exceptions raised in the try block
B. To execute code after the try and except blocks, regardless of whether an exception occurred
C. To define the type of exception
D. To exit the program after an error occurs
✅ **Answer:** B. To execute code after the try and except blocks, regardless of whether an exception occurred
(□*The finally block is executed no matter what happens in the try and except blocks*)

---

2. What will happen if there is an exception in the try block and there is no except block to catch it, but there is a finally block?

A. The finally block will be skipped
B. The finally block will be executed after the exception terminates the program
C. The finally block will catch the exception
D. The exception will not occur

✅ **Answer:** B. The finally block will be executed after the exception terminates the program

(□*Even if an exception is not caught by the except block, the finally block will always be executed before the program terminates*)

---

3. What is guaranteed about the finally block in Python?

A. It runs only if an exception occurs
B. It runs only if no exception occurs
C. It runs whether an exception occurs or not
D. It runs only if the except block is executed

✅ **Answer:** C. It runs whether an exception occurs or not

(□*The finally block always executes, regardless of whether an exception is raised or not*)

---

4. What will the following code output?

python
CopyEdit

```
try:
    print("Before exception")
    x = 1 / 0
except ZeroDivisionError:
    print("Caught exception")
finally:
    print("Finally block executed")
```

A. Before exception Caught exception Finally block executed
B. Before exception Finally block executed
C. Caught exception Finally block executed
D. Before exception Error occurred

✅ **Answer:** A. Before exception Caught exception Finally block executed

(□*The exception is caught in the except block, but the finally block is executed afterward*)

---

5. Can the finally block be skipped in Python?

A. Yes, if an exception occurs
B. No, the finally block always executes
C. Yes, if the program exits before reaching the finally block
D. No, unless there is no try block

✅ **Answer:** B. No, the finally block always executes

(□*The finally block always runs, even if the program exits or a return statement is executed in the try or except block*)

---

6. What happens if there is a return statement in the try block and a finally block?
A. The finally block will not execute
B. The finally block will execute before the return statement is executed
C. The return statement will be ignored
D. The program will crash
✅ **Answer:** B. The finally block will execute before the return statement is executed
(☐*The finally block will execute before any return statement in the try block is executed*)

---

7. What will the following code output?
python
CopyEdit
```
try:
    x = 1 / 0
except ZeroDivisionError:
    print("Caught exception")
finally:
    print("Finally block executed")
```
A. Caught exception Finally block executed
B. ZeroDivisionError Finally block executed
C. Caught exception
D. Finally block executed
✅ **Answer:** A. Caught exception Finally block executed
(☐*The exception is caught by the except block, and the finally block is executed afterward*)

---

8. Which of the following is true about the finally block in Python?
A. It is used to catch exceptions
B. It executes when an exception is raised but not handled
C. It is used for cleanup actions such as closing files or releasing resources
D. It cannot raise exceptions
✅ **Answer:** C. It is used for cleanup actions such as closing files or releasing resources
(☐*The finally block is typically used for cleanup actions, like closing files, releasing resources, or performing final steps regardless of whether an exception occurred*)

---

9. If there is a try block followed by both an except and finally block, what will happen if the finally block raises an exception?
A. The exception in the finally block will be ignored
B. The exception in the finally block will be propagated, and the except block will not be executed
C. The program will terminate after the exception in the finally block
D. The program will crash after executing the finally block
✅ **Answer:** C. The program will terminate after the exception in the finally block
(☐*If the finally block raises an exception, it will propagate, and the program may terminate unless caught elsewhere*)

---

10. Can you use finally without except in Python?

A. Yes, finally can be used alone

B. No, finally must always be paired with except

C. finally must be used before try

D. Yes, but it requires a try block

✅ **Answer:** A. Yes, finally can be used alone

(☐*The finally block can be used without an except block, in which case it will execute after the try block*)

---

11. What will the following code output?

python

CopyEdit

```
def foo():
    try:
        return 1
    finally:
        return 2

print(foo())
```

A. 1

B. 2

C. None

D. Error

✅ **Answer:** B. 2

(☐*The finally block is executed after the return statement in the try block, so the value returned from the finally block is used*)

---

12. Can you use a finally block with a for loop in Python?

A. No, finally cannot be used with loops

B. Yes, finally can be used to handle any exceptions in the loop

C. Yes, finally can be used only for terminating loops

D. No, finally is used only in exception handling

✅ **Answer:** B. Yes, finally can be used to handle any exceptions in the loop

(☐*You can use a finally block with a loop inside a try block to handle cleanup operations, regardless of whether an exception occurs in the loop*)

---

13. What happens if you put a return statement in both the try block and the finally block?

A. The return statement in the try block is ignored

B. The return statement in the finally block is ignored

C. The function will return the value from the finally block

D. The function will raise an exception

✅ **Answer:** C. The function will return the value from the finally block

(☐*If both try and finally blocks contain return statements, the value from the finally block will be returned, overriding the try block's return value*)

---

14. What will the following code output?

python
CopyEdit
```
try:
    x = 1 / 0
finally:
    print("Finally block executed")
```

A. ZeroDivisionError
B. Finally block executed
C. ZeroDivisionError Finally block executed
D. None

✅ **Answer:** C. ZeroDivisionError Finally block executed
(☐*The exception is raised, but the finally block executes afterward*)

---

15. Can you handle exceptions in the finally block?

A. No, exceptions cannot be caught in the finally block
B. Yes, but the finally block should only be used for cleanup
C. Yes, exceptions in the finally block can be caught using another try-except block
D. No, exceptions will cause the program to terminate immediately

✅ **Answer:** C. Yes, exceptions in the finally block can be caught using another try-except block
(☐*If an exception occurs in the finally block, it can be caught by another try-except structure around the finally block*)