IT5001—*Software Development Fundamentals*

Quiz 0

This quiz is **NOT** graded. Its purpose is for you to personally assess your ability to continue with this course. You will receive a refund if you withdraw from this course within the first two weeks.

You are encouraged to allocate no more than **1 hour** for this assessment. You are also encouraged to abide by the following rules in preparation for upcoming exams.

Rules:

- You are allowed **ONE** A4-sized cheat sheet, double-sided, printed or written.
- You are allowed **ONE** blank sheet of A4 paper (in addition to your cheat sheet) for drafts.
- You **cannot** refer to any another document or search for information online.
- You **cannot** access any files on your computer, including `.py` files.
- You **cannot** use any other electronic devices, including smart watches.
- You **cannot** use other tools or IDE, such as IDLE, pycharm, etc. to help you.
- You **cannot** communicate with anyone throughout the quiz.

There are **17 questions** spanning pages **2 through 4** (inclusive). Solutions for self-marking are appended to the end of this document—starting from **page 6**. This document contains **7 pages**.

Should you find this quiz exceptionally challenging, inform your lecturer.

Expression Evaluation

Without using IDLE, determine the results from evaluating the following Python expressions and select the correct option.

1) `1 + 2 - 3 + 4 - 5`
A. `1`
B. `0`
C. `-1`
D. `-3`
E. `-9`

2) `2 ** 3 ** 2`
A. `12`
B. `16`
C. `64`
D. `512`

3) `13 // 4 / 2 % 2`
A. `1`
B. `1.0`
C. `1.5`
D. `2`
E. Evaluating this expression yields an error

4) `'abc'[1]`
A. `'a'`
B. `'b'`
C. `'c'`
D. `'abc'`
E. Evaluating this expression yields an error

5) `'abc'[int('123'[1])]`
A. `'a'`
B. `'b'`
C. `'c'`
D. `'abc'`
E. Evaluating this expression yields an error

6) `'abc'[5::-2]`
A. `'ca'`
B. `'b'`
C. `'caca'`
D. `''` (empty string)
E. Evaluating this expression yields an error

7) `'abc'[3:4]`
A. `'abc'`
B. `'c'`
C. `'cba'`
D. `''` (empty string)
E. Evaluating this expression yields an error

8) `not False and not False or False`
A. `True`
B. `False`
C. Evaluating this expression yields an error

9) `'very' + 'good' + 'job'[2:3]`
A. `'verygoodb'`
B. `'r'`
C. `'veryo'`
D. `''` (empty string)
E. Evaluating this expression yields an error

10) `(('a' + 'b') * 4)[3:5]`
A. `'abababab'`
B. `'bababababa'`
C. `'ba'`
D. Evaluating this expression yields an error

Program Tracing

In each of the following questions in this section, you are given a complete Python program stored in a `.py` file. Determine the output (if any) of the program upon execution, and choose the correct option.

11)

```
ans = 1
for i in range(0, 10, 5):
    ans += i
print(ans)
```

A. 55
B. 56
C. 5
D. 6
E. Executing this program yields an error
F. The program runs in an infinite loop
G. None of the above

12)

```
ans = 0
while ans < 5:
    ans += ans
print(ans)
```

A. 0
B. 4
C. 5
D. 8
E. Executing this program yields an error
F. The program runs in an infinite loop
G. None of the above

13)

```
ans = 0.0
while ans != 100.0:
    ans += 0.001
print(ans)
```

A. 100.0
B. 99.9
C. 100.1
D. 0
E. 0.1
F. The program runs in an infinite loop
G. None of the above

14)

```
a = 1
b = 2
if a < 1:
    if b > 0:
        print('1')
    else:
        print('2')
else:
    if b < 4:
        print('3')
    else:
        print('4')
```

A. 1
B. 2
C. 3
D. 4
E. The program completes successfully but prints nothing
F. None of the above

15)

```
x = 10
y = 20
z = 30 * (x > y) + 40 * (x <= y)
print(z)
```

A. 40
B. 30
C. 20
D. 10
E. None of the above

Programming

*Reminder: You are **not allowed** to use IDLE or any other IDE to help you with the questions in this section. Note: You **cannot** import any functions/packages. However, you can use all the built-in functions and operators.*

16) People often communicate the time using either a 12h clock or a 24h clock format. The 12h clock format is either `H.MMam` or `H.MMpm` where $1 \leq$ `H` ≤ 12 and $0 \leq$ `MM` ≤ 59, or a four-digit number `HHMM` where $1 \leq$ `HH` ≤ 12 and $0 \leq$ `MM` ≤ 59; the 24h clock format is always written as a four-digit number `HHMM` where $0 \leq$ `HH` ≤ 23 and $0 \leq$ `MM` ≤ 59. Clearly, a time like `1.30am` is written in the 12h clock format, `1300` is written in the 24h clock format, and `0100` can be either in 12h or 24h format.

Write a function `which_clock(t)` that accepts as input a **string** `t` and returns `12` if `t` is written in the 12h format, `24` if `t` is written in the 24h format, or `-1` if it is a valid time in either formats.

Assume that the argument to `t` is always a string that represents a valid time in either the 12h or 24h clock format. Example runs follow:

```
>>> which_clock('1.45pm')
12
>>> which_clock('0000')
24
>>> which_clock('1159')
-1
```

17) A *cubic number* is the cube of a (strictly) positive integer $i > 0$. Write a function `largest_cube_LE(k)` that returns the largest cubic number n as an **integer** that is less than or equal to `k`, i.e. it is the largest n satisfying the following equation:

$$i^3 = n \leq k, \text{ for some positive integer } i$$

Assume that the argument to `k` is always a strictly positive integer.

Sample output follows:

```
>>> largest_cube_LE(16)
8
>>> largest_cube_LE(27)
27
>>> largest_cube_LE(10 ** 5 + 1)
97336
```

– End of Quiz –

This page was intentionally left blank.

Quiz Solutions

Expression Evaluation. Finding out the solutions to these is quite simple—just write the expressions in IDLE to see the results after evaluating them.

Here are the explanations for each expression.

(1) Answer: C. Evaluate the expression from left to right.

(2) Answer: D. The exponentiation operator (`**`) is right-associative, i.e. the expression can be re-written as `2 ** (3 ** 2)`. This represents the quantity $2^{(3^2)}$ which is 512.

(3) Answer: C. Evaluate the expression from left to right; `13 // 4` gives `3`, `3 / 2` gives `1.5`, `1.5 % 2` gives `1.5`.

(4) Answer: B. Everything starts from 0 in Computer Science, therefore the character at index `1` is the second character from the left, which is `'b'`.

(5) Answer: C. Evaluate the expression from the innermost brackets. `'123'[1]` gives `'2'`, `int('2')` gives `2`, `'abc'[2]` gives `'c'`.

(6) Answer: A. The slice starts from `5` which exceeds the length of the string, so it defaults to the rightmost character. The step is `-2` therefore we take two steps backwards along the string. The stop is empty, so we allow the slice to end (and include) the first character.

(7) Answer: D. Both the start and stop of the slice exceed the string, so the slice must be empty.

(8) Answer: A. Based on the order of precedence of operators, evaluate the `not`'s first, then the `and`, then the `or`.

(9) Answer: A. String slicing takes higher precedence, so `'job'[2:3]` gives `'b'`, and then concatenate from left to right.

(10) Answer: C. Evaluate the parenthesized expressions first, then the slice. `'a' + 'b'` gives `'ab'`, `'ab' * 4` gives `'abababab'`, `'abababab'[3:5]` gives `'ba'`.

Program Tracing. For this section you may likewise paste the code into IDLE and execute them to see their output. Note to press Ctrl + C to terminate an infinitely-running program.

(11) Answer: D. `range(0, 10, 5)` produces the sequence of numbers `0, 5`. Therefore, after the first iteration, we add `0` to `ans` (making it `1`), then in the second iteration we add `5` to `ans` making it `6`. This completes the loop, therefore `6` is printed.

(12) Answer: F. The `while` loop is entered because `ans < 5` is true. Notice that in each iteration of the loop we add `ans` to itself, therefore it remains as `0`. Therefore, the program never exits the loop.

(13) Answer: F. `ans` will never be **exactly** equal to `100.0` because in each iteration we are adding `0.001` to it, where `0.001` cannot be accurately represented by our computers as a floating point number.

(14) Answer: C. `a < 1` is a false statement, therefore the `else` branch is entered. `b < 4` is a true statement.

(15) Answer: A. `True` is equivalent to `1`, and `False` is equivalent to `0`. Therefore `z` is assigned the result of the expression `30 * (0) + 40 * (1)`.

Programming. The following are only ideas for solving these questions; better solutions exist, and if your code follows a different approach but still returns the correct output, you may consider your solution correct.

(16) If `t` ends with `'m'` then it must be in the 12h format. Otherwise if the first two digits are between 1 and 12, then it might be in either format. Otherwise, it must be in the 24h format.

```
def which_clock(t):
    if t[-1] == 'm': return 12
    if 1 <= int(t[:2]) <= 12: return -1
    return 24
```

(17) Take the cube root of `k` and convert it into an integer `i`. Note that with floating point imprecision, `i ** 3` might not be the right solution. Check if `(i + 1) ** 3` is less than equal to `k`, which if so, return it, otherwise, return `i ** 3`.

```
def largest_cube_LE(k):
    i = int(k ** (1 / 3))
    if (i + 1) ** 3 <= k:
        return (i + 1) ** 3
    return i ** 3
```