Skip to main content

Side panel

UWA-CSSE

-
- 0

  Notifications

  You have no notifications

  See all

- Zehua Zhu

  Dashboard

  Profile Grades Messages Preferences

  Log out

- CITS1401 2022S2
- Participants
- Grades
- General
- Labs

- Home
- Dashboard
- Calendar
- Private files
- My courses
- CITS1401 2022S2

CITS1401 Computational Thinking with Python (2022S2)

1. Home
2. My courses
3. CITS1401 2022S2
4. Labs
5. Lab 03. Conditions and simple loops

Information

Flag question

Information text

Conditions allow us to create decision processes in our programs. Below are some key properties of conditions.

As in the previous section, type the line or lines given the left column into your Python shell, inspect the output, read the notes, and figure out what's happening!

Input Line	Notes

```
a = 1
b = 2
a < b #will print out True
```

Various operators will check the equality of values, such as <, >, <=, >=, ==, != etc.

```
if True:
    print("yes!")
```

You can use the boolean variable values True or False directly (make sure to capitalise them).

```
if False:
    print("yes!")
else:
    print("no!")
```

Obviously you need to satisfy the condition (i.e. True) to go into the if-block, otherwise, you will be taken into the else-block.

```
if a < b:
    print("b is larger")
else:
    print("a is larger")
```

we can put boolean expressions directly into if statements.

```
if 0 < a < 10:
    print("a is between 0 and 10")
```

Python allows you to combine some of the conditional expressions, which you cannot do in other languages (most of the time). However, you are suggested to focus on the methods which is more clear to understand.

```
if a > 0 and a < 10:
    print("a is between 0 and 10")
```

Of course, you don't have to stick with Python-based syntax, you can separate those expressions as you wish.

```
if a < 0 and a > 10:
    print("a is not between 0 and 10")
```

But make sure your logic is correct because Python will not check those for you.

```
if a < 0 or a > 10:
    print("a is not between 0 and 10")
```

A fix to the above illogical expression (replace and with 'or').

```
if 5 < 0 or 9 > 10:
    print("a is not between 0 and 10")
```

the operands take higher precedence over ('or', 'and', 'not'). To differentiate, make sure to use brackets where necessary.

```
if 'he' in 'hello':
    print("he is in hello!")
```

In addition the operator 'in' is also useful to quickly check and iterative items and their membership (i.e. something belongs to the list or not).

```
if 2 in [1, 2, 3]:
    print("found 2!")
else:
    print("nope")
```

Similarly for list.

```
if 'apple' > 'banana':
    print("apple is larger than banana?!")
```

Not just numbers, you can compare strings too! Basically, a string gets converted to a number from letter by letter (e.g. try using ord('a') and see what you get), and Python then compares the value associated with each letter at the same index. For example, 'a' from 'apple' will be compared with 'b' from 'banana'. If the letters are the same, then Python moves onto the next index and compares the letters there, and so on.

Question 1

Not complete
Marked out of 1.00
Flag question

Question text

In the following code, only one block of code will be executed. However, one code block must be executed.

```
if (test 1):
    # code block 1
elif (test 2):
    # code block 2
elif (test 3):
    # code block 3
else:
    # code block 4
```

Select one:
● True
○ False
Check

Question 2

Not complete
Marked out of 1.00
Flag question

Question text

Which of the following are syntactically INCORRECT (ie their syntax is INCORRECT for the Python programming language)? Select **ALL** answers. You may assume all variables have previously been declared.

Select one or more:

☑

```
if a is > b:
   print('correct')
```

□

```
if a > b:
   print('correct')
elif a > b:
   print('correct 2')
```

☑

```
if a > b:
print('correct')
elif a > b:
print('correct 2')
```

☑

```
if a > b:
    print('correct')
else:
    print('uh oh')
elif a > b:
    print('correct 2')
```

☐

```
if a > b:
    print('correct')
```

Check

Question 3

Not complete
Marked out of 1.00
Flag question

Question text

What is printed after executing the code below? Try to answer the question without trying it out!

```
a = 10
b = 15
if a > b:
    a += b
else:
    a += b * 2

if a < b:
    print("awesome!")
elif a > 20:
    print("great!")
else:
    print("nice!")
```

Answer: great!

Check

Question 4

Not complete
Marked out of 1.00
Flag question

Question text

A block of code is given below. Which of the subsequent code blocks are equivalent? Select **ALL** answer(s), and try to solve without checking your answer. You can assume all variables have already been declared as integers, and they are all unique numbers (i.e., no two variables will have the same value).

```
if (a > b) and (b < c):
    if a > c:
        print("largest is a")
    else:
        print("largest is c")
elif a < b:
    if c < b:
        print("largest is b")
    else:
        print("largest is c")
else:
    print("largest is a")
```

Select one or more:

☐

```
if a > b > c:
    print("largest is a")
elif b > a > c:
    print("largest is b")
else:
    print("largest is c")
```

☐

```
if a > b:
    if c > a:
        print("largest is c")
    else:
        print("largest is a")
else:
    print("largest is b")
```

☑

```
if a > b:
    if a > c:
        print("largest is a")
    else:
        print("largest is c")
elif c > b:
    if a < b:
        print("largest is c")
else:
    print("largest is b")
```

☐

```
if a > b or a > c:
    print("largest is a")
elif b > a or b > c:
    print("largest is b")
else:
    print("largest is c")
```

☑

```
if a > b:
    if c > a:
        print("largest is c")
    else:
        print("largest is a")
elif b > a and b > c:
    print("largest is b")
else:
    print("largest is c")
```

☑

```
if a > b and a > c:
    print("largest is a")
elif b > a and b > c:
    print("largest is b")
else:
    print("largest is c")
```

Check

Question 5

Not complete
Marked out of 1.00
Flag question

Question text

Write a function `my_abs(value)` that returns the string mentioning whether the `value` is positive, negative or zero. The three possible outputs are "positive", "negative" or "zero". *Remember the outputs are case sensitive.*

You are *required* to use an *if* statement for this question.

For example:

Test	Result
`print(my_abs(3.5))`	`positive`
`print(my_abs(-7.0))`	`negative`
`print(my_abs(0))`	`zero`

Answer:(penalty regime: 0, 0, 10, 20, ... %)

Check

Question 6

Not complete
Marked out of 1.00
Flag question

Question text

Write a boolean function `is_odd(number)` that takes an integer parameter *number* and returns `True` if and only if `number` is odd. (Hint: consider the mod operator `%`).

For example:

Test	Result
`print(is_odd(1))`	`True`
`print(is_odd(2))`	`False`
`print(is_odd(-321))`	`True`

Answer:(penalty regime: 0, 10, ... %)

Check

Question 7

Not complete
Marked out of 1.00
Flag question

Question text

Write a function `bmi_risk(bmi, age)` that takes two positive numeric arguments as parameters `bmi` and `age` and returns a string `Low`, `Medium`, or `High` according to the following table:

	Under 45	45 or over

BMI less than 22	Low	Medium
BMI 22 or more	Medium	High

For example:

Test	Result
`print(bmi_risk(21.5, 44))`	`Low`
`print(bmi_risk(30, 27))`	`Medium`
`print(bmi_risk(30, 45))`	`High`

Answer:(penalty regime: 0, 10, ... %)

Check

Question 8

Not complete
Marked out of 1.00
Flag question

Question text

Write a function `buy_goods(cost, savings)` that takes two positive numeric arguments as parameters cost and savings and returns a Boolean type True only if the cost of the item is less than 5% of the savings, otherwise false.

For example:

Test	Result
`print(buy_goods(15, 200))`	`False`
`print(buy_goods(2, 100))`	`True`

Test	Result
`print(buy_goods(35, 1000))`	`True`

Answer:(penalty regime: 0, 10, ... %)

Check

Question 9

Not complete
Marked out of 1.00
Flag question

Question text

Rewrite the function `record_check(age, gender, location)` below so that only a single if-else statement is used. Your function does not need to return anything.

```
def record_check(age, gender, location):
    """ Function to find a male person depending on their age, gender and location """
    if age > 18:
        if gender == "M":
            if location == "Perth":
                print("Found him!")
            elif location == "Sydney":
                print("Found him!")
            else:
                print("Did not find him.")
        else:
            print("Did not find him.")
    else:
        print("Did not find him.")
```

For example:

Test	Result
`record_check(16, 'F', 'Brisbane')`	`Did not find him.`
`record_check(19, 'M', 'Perth')`	`Found him!`
`record_check(19, 'F', 'Melbourne')`	`Did not find him.`

Answer:(penalty regime: 0, 10, ... %)

```
1
```

Check

Next page

Network connection lost. (Autosave failed).

Make a note of any responses entered on this page in the last few minutes, then try to re-connect.

Once connection has been re-established, your responses should be saved and this message will disappear.

◀ Lab 02. Functions in Python (for late submission or approved special considerations)
Jump to... Jump to...
Lab 03. Conditions and simple loops (for late submission or approved special considerations) ▶
Skip <span id="mod_quiz_navblock_title">Quiz navigation</span>

Quiz navigation

Information i This page Question 1 This page Question 2 This page Question 3 This page Question 4 This page Question 5 This page Question 6 This page Question 7 This page Question 8 This page Question 9 This page Question 10 Question 11 Question 12 Question 13 Question 14 Question 15 Question 16 Question 17 Question 18 Question 19 Question 20 Question 21
Finish attempt ...

Time left

You are logged in as Zehua Zhu (Log out)

CITS1401 2022S2

Data retention summary
Get the mobile app