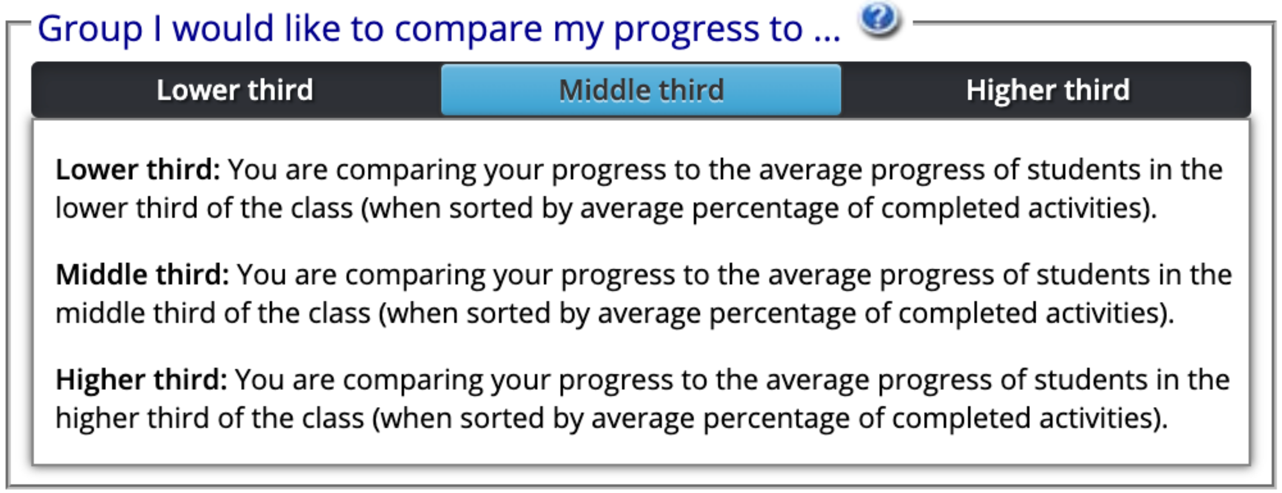


Group I would like to compare my progress to ...

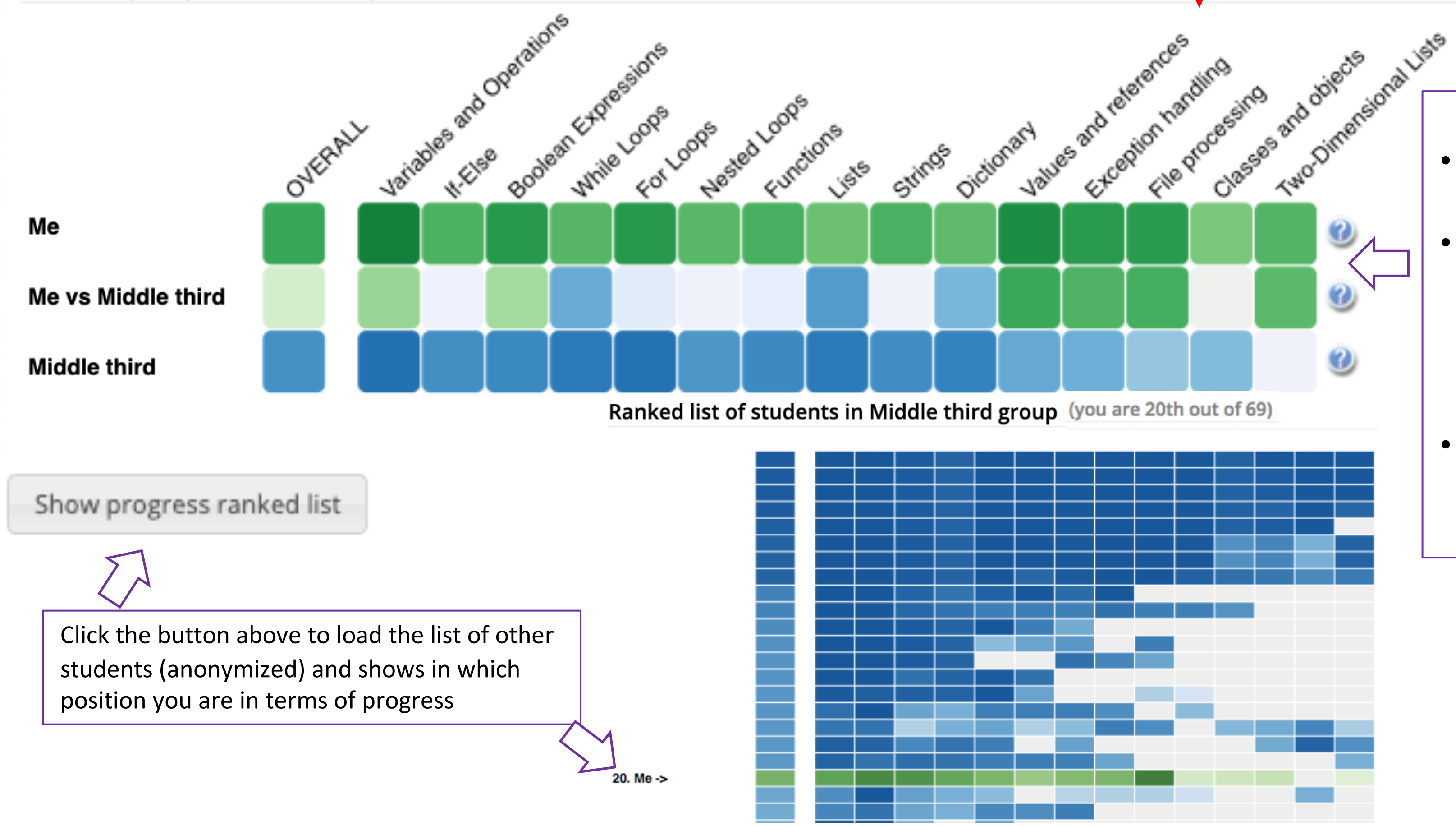
Lower third: You are comparing your progress to the average progress of students in the lower third of the class (when sorted by average percentage of completed activities).

Middle third: You are comparing your progress to the average progress of students in the middle third of the class (when sorted by average percentage of completed activities).

Higher third: You are comparing your progress to the average progress of students in the higher third of the class (when sorted by average percentage of completed activities).

When you click on **Lower third, Middle third** or **Higher third**, the progress visualization below will be automatically updated to reflect the average progress of the students in that selected group. Students are divided into three equal groups based on their rank

If you want to compare your progress with lower/middle/higher third group of the students in the class, you need to click **“Lower third”**/ **“Middle third” / “Higher Progress”**. The system will remember your choice next time you accessed it.



Click the button above to load the list of other students (anonymized) and shows in which position you are in terms of progress

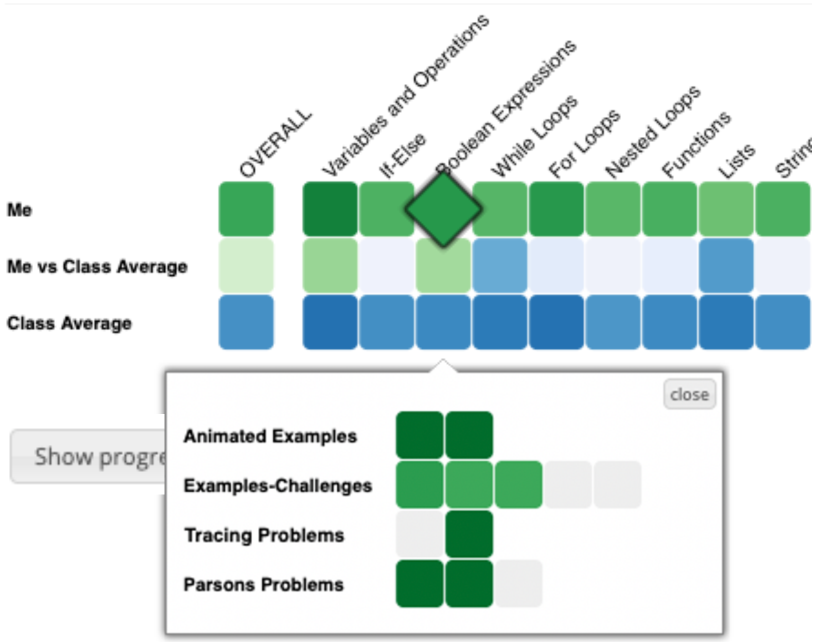
Progress Visualization

- First row (Me) shows **your progress** (Darker green means more progress on that topic)
- Second row(Me vs *group*) **compares your progress** with the **average progress of the students in selected group** (Darker green means you have more progress than the group; darker blue means they have more progress than you; grey means equal progress.
- Third row (*Group*) shows the **average progress of students in the selected group** (Darker blue means more progress on that topic)

How to Increase your Progress?

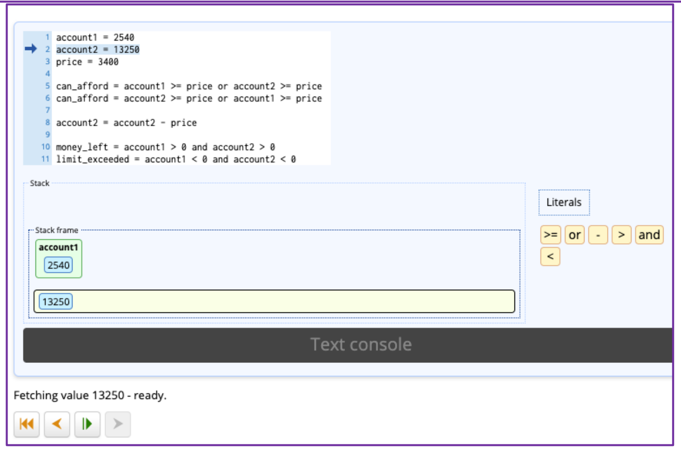
To have more greener cells on ***Me*** row, you need to interact with the learning activities inside each topic.

Click on a topic cell as shown below and access the contents. Viewing animation steps, clicking on example lines or solving challenges, questions and parsons problems to increase your progress.



Animated Examples

Play animation steps to how the program executed

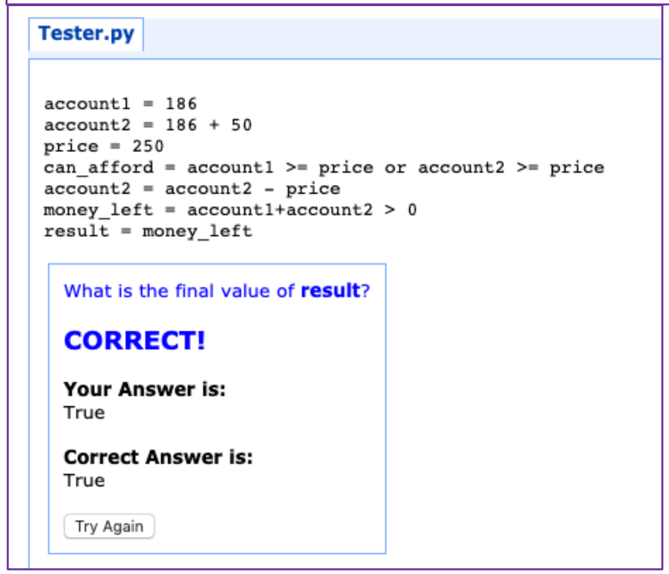


```
account1 = 2540
account2 = 13250
price = 3400

can_afford = account1 >= price or account2 >= price
can_afford = account2 >= price or account1 >= price

account2 = account2 - price

money_left = account1 > 0 and account2 > 0
limit_exceeded = account1 < 0 and account2 < 0
```

Fetching value 13250 - ready.

Tracing Problems

Predict the output of the program. It is either the console output or the value of ***result*** variable.



Tester.py

```
account1 = 186
account2 = 186 + 50
price = 250
can_afford = account1 >= price or account2 >= price
account2 = account2 - price
money_left = account1+account2 > 0
result = money_left
```

What is the final value of **result**?

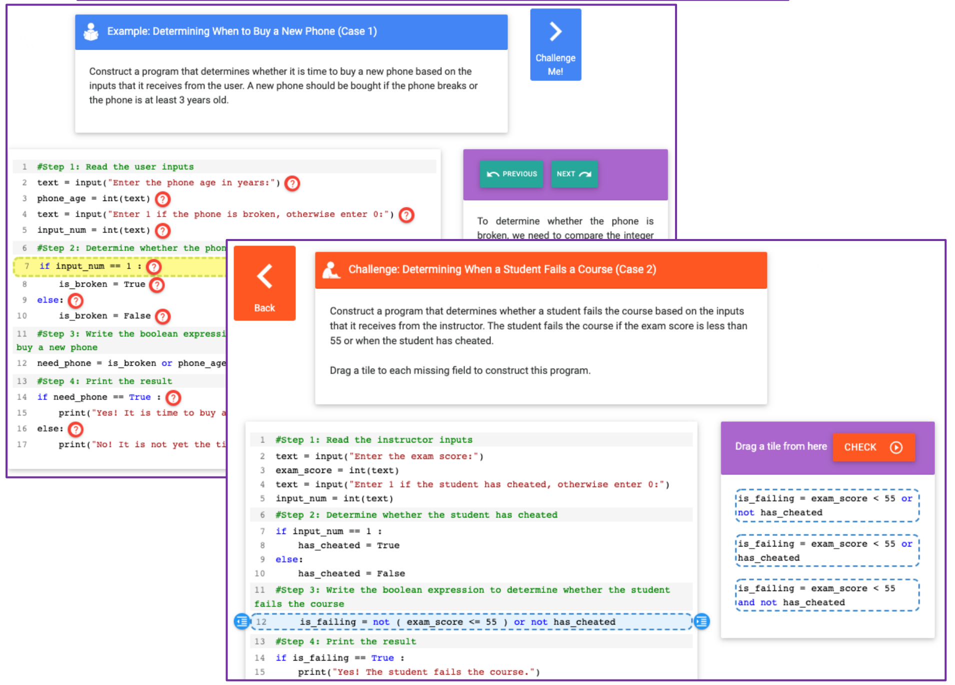
CORRECT!

Your Answer is:
True

Correct Answer is:
True

Examples-Challenges

Check how a program is constructed line by line in examples and challenge yourself with challenges and complete the missing lines.



Example: Determining When to Buy a New Phone (Case 1)

Construct a program that determines whether it is time to buy a new phone based on the inputs that it receives from the user. A new phone should be bought if the phone breaks or the phone is at least 3 years old.

```
#Step 1: Read the user inputs
text = input("Enter the phone age in years:")
phone_age = int(text)
text = input("Enter 1 if the phone is broken, otherwise enter 0:")
input_num = int(text)
#Step 2: Determine whether the phon
if input_num == 1 :
    is_broken = True
else:
    is_broken = False
#Step 3: Write the boolean expressi
buy a new phone
need_phone = is_broken or phone_age
#Step 4: Print the result
if need_phone == True :
    print("Yes! It is time to buy a
else:
    print("No! It is not yet the ti
```

To determine whether the phone is broken, we need to compare the integer

Challenge: Determining When a Student Fails a Course (Case 2)

Construct a program that determines whether a student fails the course based on the inputs that it receives from the instructor. The student fails the course if the exam score is less than 55 or when the student has cheated.

Drag a tile to each missing field to construct this program.

```
#Step 1: Read the instructor inputs
text = input("Enter the exam score:")
exam_score = int(text)
text = input("Enter 1 if the student has cheated, otherwise enter 0:")
input_num = int(text)
#Step 2: Determine whether the student has cheated
if input_num == 1 :
    has_cheated = True
else:
    has_cheated = False
#Step 3: Write the boolean expression to determine whether the student
fails the course
    is_failing = not ( exam_score <= 55 ) or not has_cheated
#Step 4: Print the result
if is_failing == True :
    print("Yes! The student fails the course.")
```

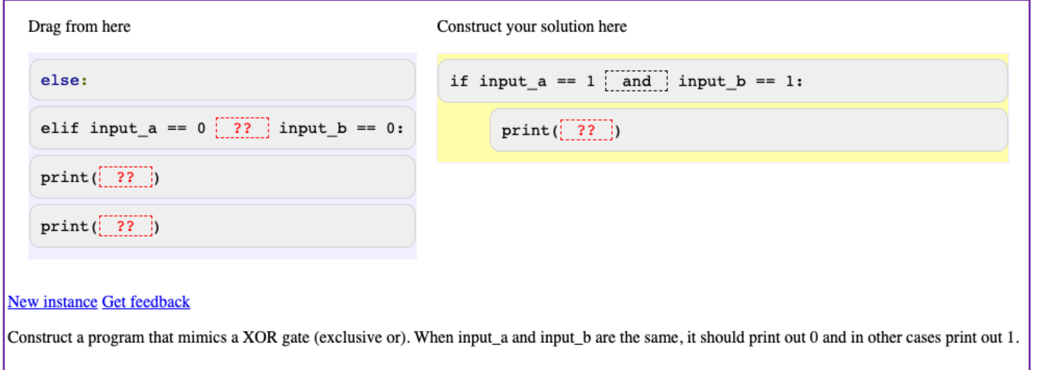
```
is_failing = exam_score < 55 or
not has_cheated
```

```
is_failing = exam_score < 55 or
has_cheated
```

```
is_failing = exam_score < 55
and not has_cheated
```

Parsons Problem

Reorder the program lines to solve the given task at the bottom of the screen. Pay attention to indentation.



Drag from here

```
else:
```

```
elif input_a == 0  ??  input_b == 0:
```

```
print( ?? )
```

```
print( ?? )
```

Construct your solution here

```
if input_a == 1  and  input_b == 1:
    print( ?? )
```

New instance Get feedback

Construct a program that mimics a XOR gate (exclusive or). When input_a and input_b are the same, it should print out 0 and in other cases print out 1.