CSC108 FSG

Wed. 2-3pm MN2266

Intro & Ice Breaker

Introduce yourself to others…

- ❏ Name
- ❏ One fun fact about yourself
- ❏ What you want to get out of the FSGs

Activity 1 - For Loops

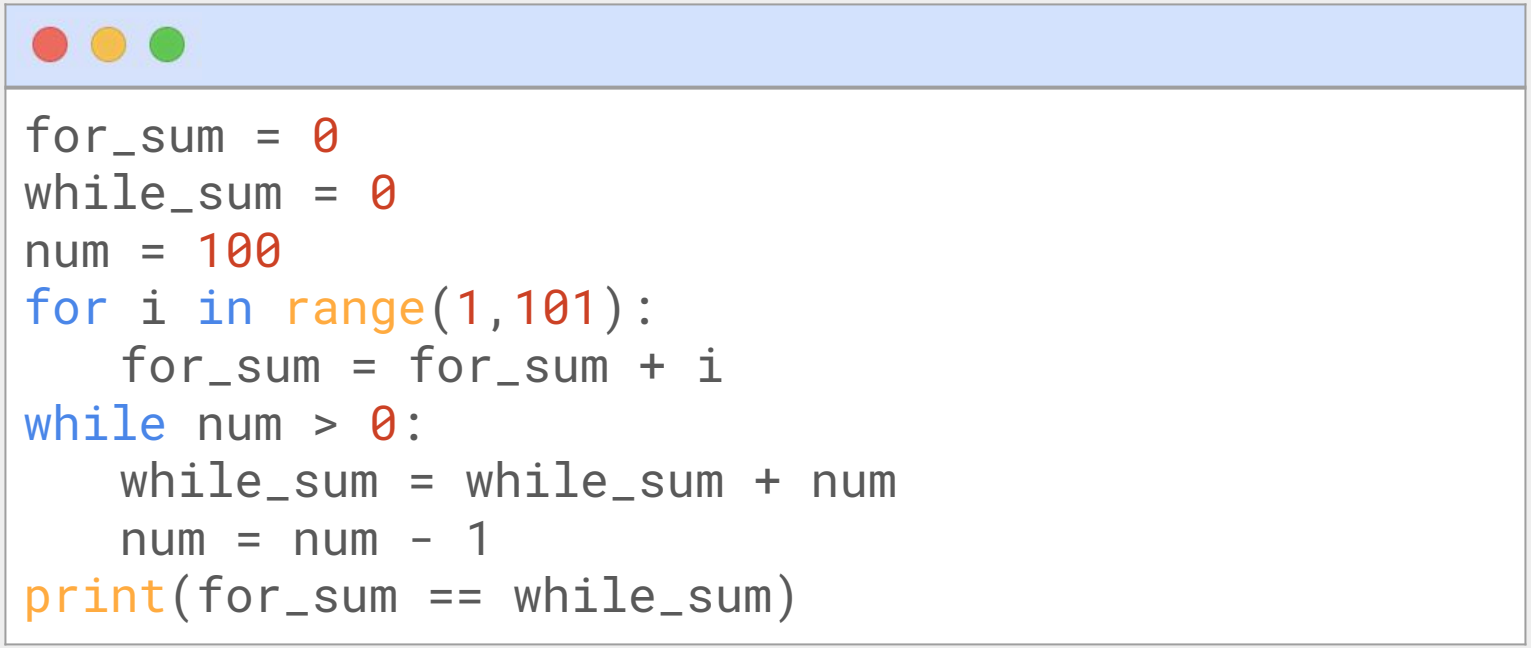
```
sum = 0

for i in range(1,101):
    sum = sum + i

print(sum)
```

Question : What will be the output of the following program?

Activity 1 - While Loops



```
for_sum = 0
while_sum = 0
num = 100
for i in range(1,101):
    for_sum = for_sum + i
while num > 0:
    while_sum = while_sum + num
    num = num - 1
print(for_sum == while_sum)
```

Question : What will be the output `True` or `False`?

Activity 1 - For/While Loops

Question : When should you choose `for` loop over `while` loop?

A. When the number of iteration is know in advance

B. When the number of iteration is unknown

C. When the condition for termination depends on dynamic input

D. When we want to terminate loop based on condition rather than the number of times it runs

Activity 1 - For/While Loops

Question : Which scenario is most appropriate to use a `while` instead of `for` loop?

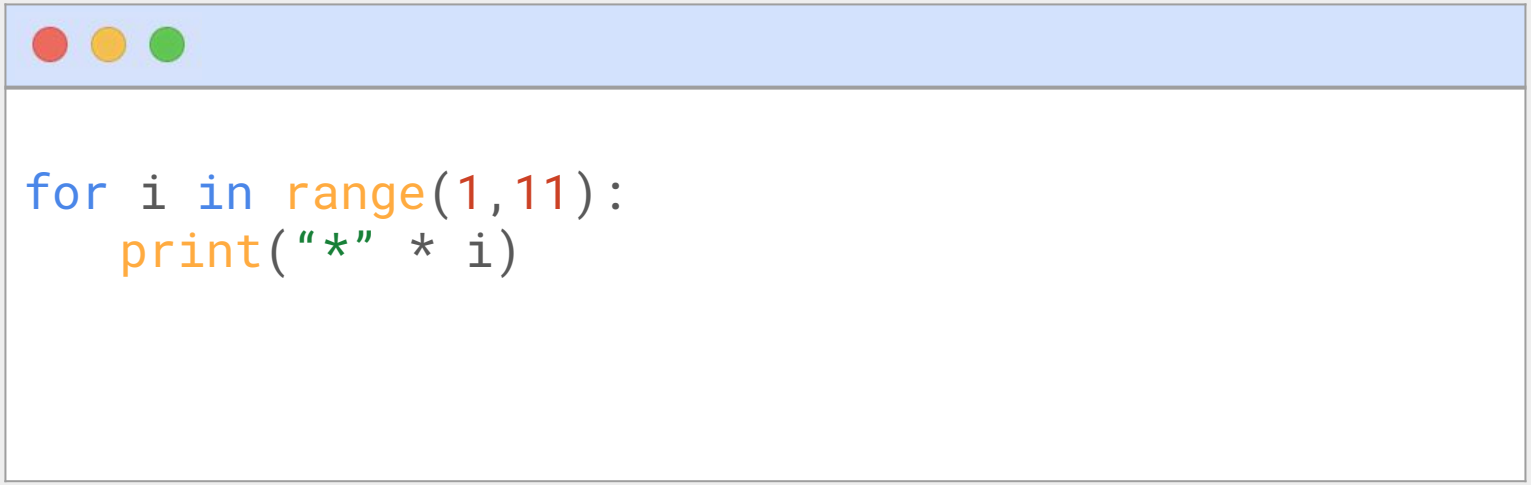
A. Iterating through a list of scores to calculate the average

B. Continuously asking the user for input repeatedly until they type "Stop."

C. Printing each character in a string

D. Summing numbers in a fixed range from 1 to 100

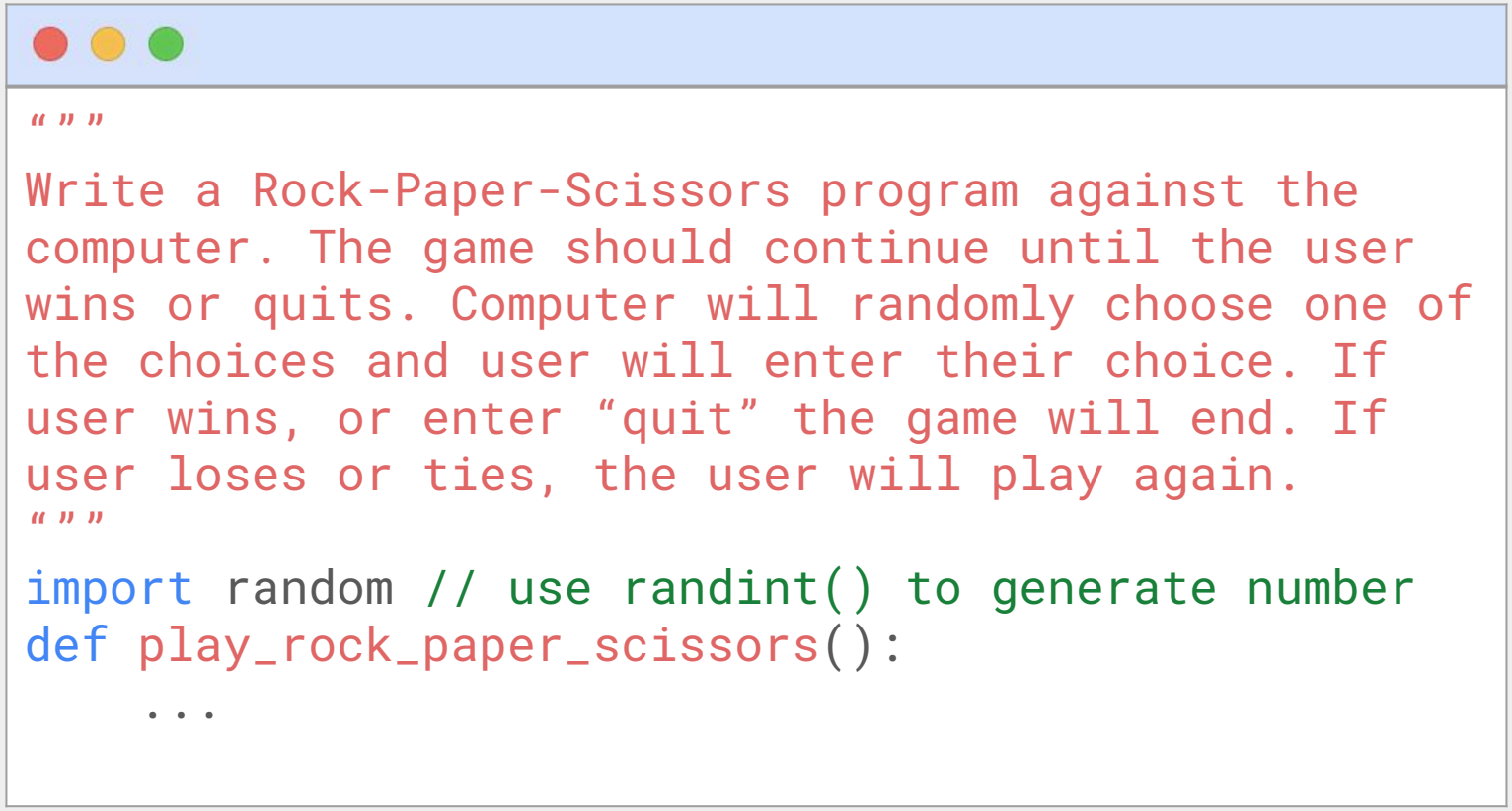
Activity 2 - For/While Loops



```
for i in range(1,11):
    print("*" * i)
```

Question : Write a program that prints the same output using the `while` loop

Activity 2 - For/While Loops



```
“””
Write a Rock-Paper-Scissors program against the
computer. The game should continue until the user
wins or quits. Computer will randomly choose one of
the choices and user will enter their choice. If
user wins, or enter “quit” the game will end. If
user loses or ties, the user will play again.
“””
import random // use randint() to generate number
def play_rock_paper_scissors():
    ...
```

Activity 3 - Creating a Challenge Question

Topics to implement:

- ❏ For loops
- ❏ While loops
- ❏ List
- ❏ Week 1 - 4 materials(if/else, string slicing, .etc)

Try to make a question that you are not confident in solving. Good luck!

Sample Challenge Question

Write a function that, when given a non-empty integer list and a target sum, returns a list of unique pairs that add up to the target sum. Each pair should be returned as a list, and the first element of the pair should be less than or equal to the second element of the pair.

```
find_unique_pairs([1, 2, 3, 4, 5, 6], 6)

>> [[1, 5], [2, 4]] // 1 <= 5, 2 <= 5
```

```
def find_unique_pairs(numbers: list[int], target_sum: int) -> list[list[int,int]]:
```