Can NAAS go out yet?

Loops in Python

Resource

Secondary

11-14 years

Contents

Noteable Activities for Schools: Loops in Python

These resources are a guide for teachers to demonstrate to the whole class or direct individual students as appropriate. The activities below can be directly distributed to pupils.

For instructions on how to install and use Noteable resources, please look at our guides for teachers in GLOW: GLOW guidance for teachers to start using Noteable.

Content and Curriculum links

Level	Context	Indicators
11-14	Loops in Python	For, While, Range

Knowledge	Using bullet point lists to give instructions
Curriculum links (England) Computing KS2	• design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
Curriculum links Wales)	
Scottish Curriculum for Excellence	**Experiences and Outcomes:** • I understand language constructs for representing structured information. **TCH 3-14a** **Benchmark:** ○ Understands simple compression and encryption techniques used in computing technology.
All: Cross-curricular opportunities	The activities have identified opportunities for Mathematics, Sciences, Language, Arts and Design and Physical Education.

What are loops in coding?

Imagine you have a toy train track with lots of pieces. You want the train to go around the track multiple times. Instead of telling the train to go around each piece one by one, you can use a loop. A loop is like a magical spell that repeats a set of instructions over and over again.

First, let's remember Control Flow Statements (if-else):

- Imagine you're playing a game, and you have to make decisions based on certain conditions. For example:
    - If it's raining outside, you'll take an umbrella.
    - Otherwise, you'll wear your regular shoes.
- In Python, we use if and else statements to make decisions in our code. Here's an example:

Python Copy code

```python
weather = "rainy"

if weather == "rainy":

    print("Take an umbrella!")

else:

    print("Wear regular shoes.")
```

If the weather is rainy, it prints "Take an umbrella!" Otherwise, it prints "Wear regular shoes."

Types of Loops:

1. **For Loop**:

    Think of a conveyor belt that moves items one by one. A for loop in Python helps us go through a list of things (like numbers or words) one at a time.

    Example: Suppose we want to print the numbers from 1 to 5:

Python Copy code

```python
for number in range(1, 6):

    print(number)
```

This loop will print:

```
Python

1
2
3
4
5
```

For Loop with a range() Function:

- The range() function gives us a sequence of numbers. It's like having a bunch of numbered cards.
- Example: If we want to count from 1 to 10:

```
Python                                                    Copy code

for num in range(1, 11):
    print(num)
```

This will print the numbers from 1 to 10.

Another example of For Loop (Our Conveyor Belt):

- Let's say we have a list of names: "Alice," "Bob," "Charlie," and "David."
- We can use a for loop to say hello to each person:

```
Python                                                    Copy code

names = ["Alice", "Bob", "Charlie", "David"]
for name in names:
    print(f"Hello, {name}!")
```

When we run this code, it will print:

```
Python

Hello, Alice!

Hello, Bob!

Hello, Charlie!

Hello, David!
```

2. While Loop:

- Imagine you're playing a game, and you keep doing something until a specific condition is met. A while loop works similarly.
- Example: Let's count down from 10 to 1:

```
Python                                                    Copy code

countdown = 10

while countdown > 0:

    print(countdown)

    countdown -= 1
```

This loop will print:

```
Python

10

9

8

...

1
```

Another example of While Loop (The Game Quest):

- Imagine you're on a treasure hunt. You keep searching for clues until you find the hidden treasure.
- Here's how a while loop works:

Python Copy code

```python
coins_collected = 0

while coins_collected < 5:

    print(f"Found a coin! Total coins: {coins_collected}")

    coins_collected += 1
```

When we run this code, it will print:

Python

```
Found a coin! Total coins: 0

Found a coin! Total coins: 1

Found a coin! Total coins: 2

Found a coin! Total coins: 3

Found a coin! Total coins: 4
```

3. Break and Continue:

- Sometimes you want to stop a loop early or skip some parts. That's where break and continue come in.
- break stops the loop completely, like saying, "I'm done!"
- continue skips the current step and moves to the next one.

Example:

Python Copy code

```python
for num in range(1, 11):

    if num == 5:

        break  # Stops the loop when num is 5
    print(num)
```

This will print:

```
Python

1

2

3

4
```

4. **Nested Loop:**

- Imagine you have a box of toys, and each toy has smaller parts. A nested loop is like going through those smaller parts inside each toy.
- Example: Printing a multiplication table:

```
Python                                                        Copy code

for i in range(1, 6):

    for j in range(1, 6):

        print(i * j, end=" ")

    print()  # Move to the next line
```

This will print:

```
Python

1 2 3 4 5

2 4 6 8 10

3 6 9 12 15

4 8 12 16 20

5 10 15 20 25
```

Each row represents the result of multiplying two numbers.

Why Use Loops?

- Loops help us avoid repeating the same code over and over. They make our programs shorter, like using a magic spell to do a task many times.
- So, next time you want your toy train to go around the track or collect all the coins in a game, remember Python loops!

Please observe the indentation in all codes! They are important for the correct run of the codes and can't be missed out!

Activity 1

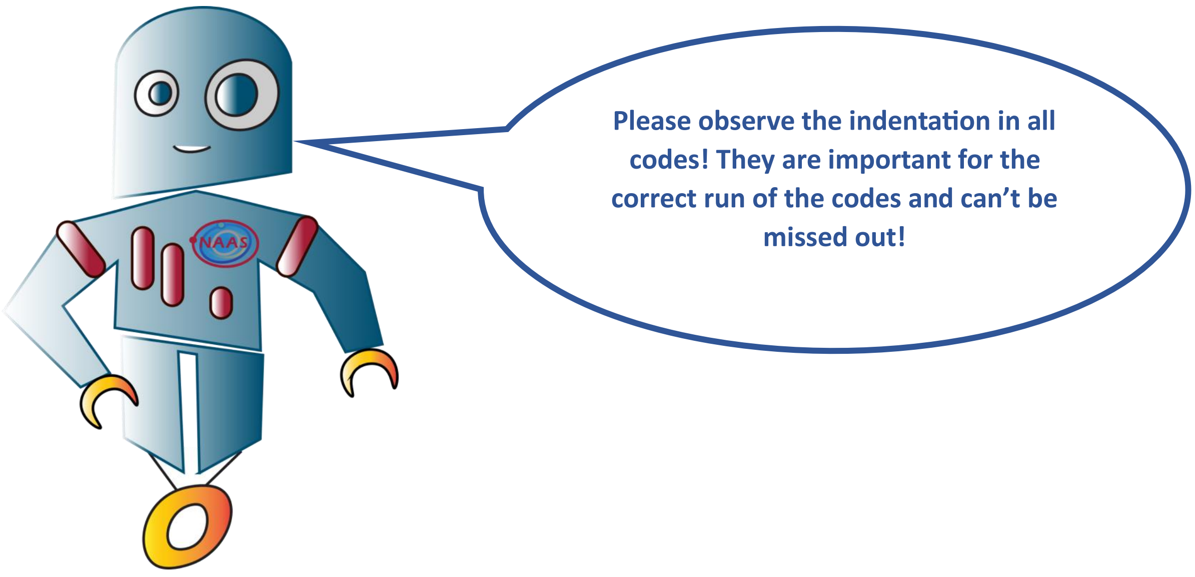
a) NAAS wants to go for a walk, but as you know, robots usually can't get wet! It was raining the whole day yesterday, but today it is sunny. As NAAS is a little worried that the rain may come back, he wants your help in guessing the weather for him.

Write a program that asks the user to input the current weather (e.g., "sunny," "rainy," "cloudy"). Based on their input, your program should give a fun response. For example:

Python Copy code

```python
weather = input("What's the weather like today? ")

if weather == "sunny":

    print("Time for a picnic!")

elif weather == "rainy":

    print("Grab an umbrella!")

else:

    print("Enjoy the day!")
```

b) Oh, No! It started to rain again! As your predictions are part of a guessing game and too funny to be trusted, NAAS looked at the weather website and saw that the rain is supposed to stop in 30 seconds. Use a while loop to count down the seconds left for the rain to stop:


NAAS
Please observe the indentation in all codes! They are important for the correct run of the codes and can't be missed out!

Activity 2

Growing Percentages

NAAS is ready for an experiment using loops in Python. Let’s work with him step by step:

1. Execute the following Python code snippet:

Python   Copy code

```
for count in range(1, 31):
    print(count * "%")
```

2. Observations: What happens when you run this code?
3. Discussion**:**
    - What do you observe about the pattern?
    - How does the number of percentage symbols change with each iteration?
    - Can you predict how many symbols will be printed at the end?
4. Challenge: Your teacher will give you a different iteration. Can you create a code to run that iteration and analyse it following the same steps that you used for the previous code?
5. Variations**:**
    - Modify the code to use a different character (e.g., *, #, or @) instead of %.
    - Adjust the range or step size to explore different patterns.
6. Conclusion**:** Discuss the concept of repetition, patterns, and how loops work.

Activity 3

Multiplication Magic:

- Write a program that generates a multiplication table for any number. Ask the a friend for their favorite number, and then use nested loops to create the table:

Python Copy code

```
favorite_number = int(input("Enter your favorite number: "))

for i in range(1, 11):

    print(f"{favorite_number} x {i} = {favorite_number * i}")
```

Activity 4 – Ready for a Challenge?

Secret Code Breaker:

- Create a secret code by shifting the letters of a word. Ask the user for a word and a shift value (e.g., 3). Then, use a for loop to encode the word:
- You can use the example below to create your own secrete code.

Python Copy code

```
word = input("Enter a word: ")

shift = int(input("Enter the shift value: "))

encoded_word = ""

for letter in word:

    encoded_letter = chr(ord(letter) + shift)

    encoded_word += encoded_letter

print(f"Encoded word: {encoded_word}")
```

Answers – for teachers

Guess the weather:

Activity 1 a):

In this activity, students create a simple weather decision program. They ask the user for the current weather (e.g., “sunny,” “rainy,” “cloudy”) and provide a fun response based on the input.

Example Code:

Python Copy code

```python
weather = input("What's the weather like today? ")

if weather == "sunny":

    print("Time for a picnic!")

elif weather == "rainy":

    print("Grab an umbrella!")

else:

    print("Enjoy the day!")
```

Explanation: The program checks the user’s input and responds accordingly. If it’s sunny, it suggests a picnic; if rainy, it advises grabbing an umbrella; otherwise, it encourages enjoying the day.

Activity 1 b): Students create a countdown timer for 30 seconds. This activity can be extended to working in pairs. Students can create a countdown timer for a special event (e.g., birthday, New Year’s Eve). They ask their partner for the number of seconds and then use a while loop to count down.

You may need to show the code below to the students for the first time, then they can create their own adaptations afterwards.

Python Copy code

```python
import time



seconds = 30

while seconds > 0:
    print(f"Time left: {seconds} seconds")

    time.sleep(1)  # Pause for 1 second

    seconds -= 1


print("Hooray! NAAS can go out!")
```

Explanation: The program keeps printing the remaining time until it reaches zero, creating a countdown effect.

Activity: Growing Percentages

In this activity students can visualize the growing sequence of percentage symbols.

1. **Introduction:**

- Explain that we'll be printing a pattern of percentage symbols.
- Invite students to follow along and observe how the pattern evolves.

2. **Code Implementation:**

- Display the initial pattern (just one percentage symbol).
- Execute the following Python code snippet:

Python   Copy code

```python
for count in range(1, 31):

    print(count * "%")
```

3. **Observations:**

- As the loop progresses, the number of percentage symbols increases.
- Discuss how the pattern grows and what students notice.

4. **Discussion:**

Ask students:

- What do you observe about the pattern?
- How does the number of percentage symbols change with each iteration?
- Can you predict how many symbols will be printed at the end?

5. **Visualize the Output:**

- Display the output of the loop step by step:

- %
- %%
- %%%%
- %%%%%%
- ...

6. Challenge:

- Encourage students to predict the output for a specific iteration (e.g., when count is 10 or 20).
- Run the code to verify their predictions.

7. Variations:

- Modify the code to use a different character (e.g., *, #, or @) instead of %.
- Adjust the range or step size to explore different patterns.

8. Conclusion:

- Discuss the concept of repetition, patterns, and how loops work.
- Highlight that programming allows us to create interesting visual effects and explore mathematical sequences.

Activity 3: Multiplication Magic:

- **What it is**: Students generate a multiplication table for any given number. They ask the user for their favorite number and use nested loops to create the table.

Example Code:

Python Copy code

```python
favorite_number = int(input("Enter your favorite number: "))

for i in range(1, 11):

    print(f"{favorite_number} x {i} = {favorite_number * i}")
```

Explanation: The program calculates the product of the favorite number with each value from 1 to 10, creating a multiplication table.

Activity 4: Secret Code Breaker:

- What it is: Students create a simple encoding program. They ask the user for a word and a shift value (e.g., 3). Then, they use a for loop to encode the word by shifting its letters.
- This activity is an optional challenge to advanced students. It adds new vocabulary to the code which should be explained before introduced to students.

Example Code:

Python Copy code

```
word = input("Enter a word: ")

shift = int(input("Enter the shift value: "))

encoded_word = ""

for letter in word:

    encoded_letter = chr(ord(letter) + shift)

    encoded_word += encoded_letter

print(f"Encoded word: {encoded_word}")
```

Explanation: The program shifts each letter in the word by the specified value (e.g., shifting ‘a’ by 3 becomes ‘d’). It then prints the encoded word.

Cross-Curricular Opportunities

1. **Mathematics**:

- **Multiplication Tables**: Use loops to generate multiplication tables. For instance, create a program that prints the multiplication table for a given number.

Code

Python Copy code

```
def multiplication_table(number):
    for i in range(1, 11):
        print(f"{number} x {i} = {number * i}")


multiplication_table(5)  # Example for the 5 times table
```

- **Geometry:** You can create a square using loops.

Code:

Python Copy code

```
import turtle

# Set up the turtle
t = turtle.Turtle()

# Draw a square
for _ in range(4):
    t.forward(100)
    t.right(90)

# Hide the turtle
t.hideturtle()
turtle.done()
```

2. **Science**:

- **Simulating Natural Phenomena**: Teach concepts like growth, decay, or population dynamics using loops. For example, simulate the growth of bacteria over time.

Code

Python Copy code

```
def bacterial_growth(initial_population, growth_rate, days):
    population = initial_population
    for day in range(1, days + 1):
        population *= (1 + growth_rate)
        print(f"Day {day}: Population = {population:.2f}")

bacterial_growth(initial_population=100, growth_rate=0.05, days=10)
```

3. **Language Arts**:

- **Word Patterns**: Explore loops to analyze word patterns. For instance, count the occurrences of vowels in a given text.
    - Code

Python Copy code

```
def bacterial_growth(initial_population, growth_rate, days): def count_vowels(text):
    vowels = "aeiou"
    vowel_count = 0
    for char in text.lower():
        if char in vowels:
            vowel_count += 1
    return vowel_count

sample_text = "The quick brown fox jumps over the lazy dog."
print(f"Number of vowels: {count_vowels(sample_text)}")
```

- **Word** Frequency **Analysis:** Count word **frequencies** in a text.

Code:

Python Copy code

```python
text = "The quick brown fox jumps over the lazy dog."

word_counts = {}

for word in text.lower().split():

    word_counts[word] = word_counts.get(word, 0) + 1

print(word_counts)
```

4. **Art and Design:**

- **Turtle Graphics:** Create a colourful spiral using loops.

Code:

Python Copy code

```python
import turtle


t = turtle.Turtle()

colors = ["red", "orange", "yellow", "green", "blue", "purple"]


for i in range(360):

    t.color(colors[i % len(colors)])

    t.forward(i)

    t.right(59)


t.hideturtle()

turtle.done()
```

5. **Physical Education:**

- **Fitness Tracking:** Track daily exercise repetitions using loops.

Code:

Python Copy code

```
push_ups = 0

for day in range(1, 8):

    reps = int(input(f"Day {day}: How many push-ups did you do? "))

    push_ups += reps



print(f"Total push-ups this week: {push_ups}")
```

Please note that the codes above are generalised examples of what you can do using Python on Noteable. To be used, each code will need to be adapted to the complexity of these examples based on the students' proficiency level. Encourage creativity and curiosity as they explore Python loops in various contexts!

References:

1. Cross Curricular Computing - iCompute[1]
2. Barefoot Computing - Primary Computing Resources[2]
3. Python Exercises for Kids[3]

Remember, the key is to encourage students to think creatively and apply Python loops across different contexts. By doing so, they'll deepen their understanding of both programming and other subjects!

Copyrights