

Class #04: Loops in Python

- ❖ Working with the while loop.
- ❖ Understanding the for loop.
- ❖ Creating infinite and nested loops.
- ❖ Using break, continue, and pass statements. Assignment #4: Loop exercises and control flow challenges.

Python all Data Type:

- Single Types:
 - ✚ int, float, complex, bool, None, dict, set, frozenset
- Sequence Types:
 - ✚ str, list, tuple, range, bytes, bytearray, memoryview

নিচে Python এর operators গুলি সারণী আকারে সাজানো হলো:

ক্ষপ	অপারেটরসমূহ
Arithmetic Operators	+, -, *, /, //, %, **
Assignment Operators	=, +=, -=, *=, /=, //=, %=, **=, &=, `
Comparison Operators	==, !=, >, <, >=, <=
Logical Operators	and, or, not
Identity Operators	is, is not
Membership Operators	in, not in
Bitwise Operators	&, `

Python Membership Operators:

Python-এ Membership Operators দুটি অপারেটর থাকে: **in** এবং **not in**। এগুলি ব্যবহার করা হয় যেকোনো সিকুয়েন্স টাইপ ডেটা (যেমন: List, Tuple, String, Set) এর মধ্যে কোনো মান বা element আছে কিনা তা যাচাই করার জন্য।

Python এ **in** এবং **not in** অপারেটরের মান শুধুমাত্র True অথবা False হতে পারে, উপাদানটি সিকুয়েন্স বা কালেকশনে উপস্থিত থাকলে True, না থাকলে False।

```
# fruits তালিকায় কিছু ফলের নাম
fruits = ['apple', 'banana', 'cherry']
```

1. in Operator:

এই অপারেটরটি চেক করে যে নির্দিষ্ট মানটি সিকুয়েন্সে রয়েছে কি না। যদি মানটি সিকুয়েন্সে থাকে, তাহলে এটি True রিটার্ন করে, অন্যথায় False রিটার্ন করে।

```
# চেক করছি যে 'apple' ফলটি fruits তালিকায় রয়েছে কি না (IN)
print('apple' in fruits) # True, কারণ 'apple' fruits তালিকায় রয়েছে
# চেক করছি যে 'orange' ফলটি fruits তালিকায় রয়েছে কি না (IN)
print('orange' in fruits) # False, কারণ 'orange' fruits তালিকায় নেই
```

2. not in Operator:

এই অপারেটরটি চেক করে যে নির্দিষ্ট মানটি সিকোয়েন্সে নেই কি না। যদি মানটি সিকোয়েন্সে না থাকে, তাহলে এটি True রিটার্ন করে, অন্যথায় False রিটার্ন করে।

```
# চেক করছি যে 'apple' ফলটি fruits তালিকায় নেই কি না (NOT IN)
print('apple' not in fruits) # False, কারণ 'apple' fruits তালিকায় রয়েছে
# চেক করছি যে 'orange' ফলটি fruits তালিকায় নেই কি না (NOT IN)
print('orange' not in fruits) # True, কারণ 'orange' fruits তালিকায় নেই
```

এভাবে, **Membership Operators** আপনাকে নির্দিষ্ট মানটি কোনো সিকোয়েন্সে (যেমন: List, String, Set) আছে কিনা তা দ্রুত জানার সুযোগ দেয়।

```
# fruits তালিকায় কিছু ফলের নাম
fruits = ['apple', 'banana', 'cherry']

# চেক করছি যে 'apple' ফলটি fruits তালিকায় রয়েছে কি না (IN)
print('apple' in fruits) # True, কারণ 'apple' fruits তালিকায় রয়েছে
# চেক করছি যে 'orange' ফলটি fruits তালিকায় রয়েছে কি না (IN)
print('orange' in fruits) # False, কারণ 'orange' fruits তালিকায় নেই

# চেক করছি যে 'apple' ফলটি fruits তালিকায় নেই কি না (NOT IN)
print('apple' not in fruits) # False, কারণ 'apple' fruits তালিকায় রয়েছে
# চেক করছি যে 'orange' ফলটি fruits তালিকায় নেই কি না (NOT IN)
print('orange' not in fruits) # True, কারণ 'orange' fruits তালিকায় নেই
```

Output window:

```
C:\Users\Minhazul Kabir> NumPy.py
True
False
False
True
```

Loop:

এখানে বাস্তবিক উদাহরণ দেখানো হলো, যেখানে, লুপ ব্যবহার করে কাজটি সহজে করা যাবে:

পণ্য দাম যোগ করা:

ধরি, তুমি একটি দোকানে কাজ করছো এবং তোমার কাছে কিছু পণ্যের দাম দেওয়া হয়েছে। তোমার কাজ হলো, সমস্ত পণ্যের দাম যোগ করে মোট খরচ বের করা। প্রতিটি পণ্যের দাম আলাদা আলাদা করে যোগ করতে হলে সময় নষ্ট হতে পারে, কিন্তু লুপ ব্যবহার করলে তুমি পুরো তালিকা একবারে দেখতে পারবে এবং স্বয়ংক্রিয়ভাবে মোট খরচ বের করতে পারবে।

পণ্যের দাম:

1. প্রথম পণ্যের দাম ১০০ টাকা
2. দ্বিতীয় পণ্যের দাম ২০০ টাকা
3. তৃতীয় পণ্যের দাম ৫০ টাকা
4. চতুর্থ পণ্যের দাম ১৫০ টাকা
5. পঞ্চম পণ্যের দাম ১২৫ টাকা
6. ষষ্ঠ পণ্যের দাম ২৭ টাকা

তোমাকে লুপ ব্যবহার করে মোট খরচ বের করতে হবে। লুপের মাধ্যমে প্রতিটি পণ্যের দাম যোগ করা যাবে।

এভাবে: **মোট খরচ: $100 + 200 + 50 + 150 + 125 + 27 = 602$ টাকা।**

আপনি যদি Python-এর লুপ সম্পর্কে শিখতে চান, তবে W3Schools একটি ভালো রিসোর্স হতে পারে। তাদের

দুটি পেজে আপনি পাইথনের While Loop এবং For Loop সম্পর্কে বিস্তারিত জানতে পারবেন:

https://www.w3schools.com/python/python_while_loops.asp ও

https://www.w3schools.com/python/python_for_loops.asp এখানে আপনি পাইথনের লুপের বিভিন্ন

ধরনের ব্যবহার এবং তাদের উদাহরণ পেতে পারেন।

লুপ অর্থ ফাঁস। প্রোগ্রামিং এ loop হচ্ছে একই কাজের পুনরাবৃত্তি।

➤ একটা লুপের ৪টা অংশ:

01. Start (শুরু)
02. Condition (Relational/Boolean Operator) (শর্ত)
03. Statement (বিবৃতি/কাজ)
04. Increment/Decrement (বৃদ্ধি/হ্রাস)

➤ লুপের কার্যপদ্ধতিঃ

- 
- 1) শর্ত যাচাই করবে
 - 2) শর্ত সত্যি হলে কাজ, শর্তের মধ্যে কাজ করবে
 - 3) বৃদ্ধি বা হ্রাস করবে
- 3) নং থেকে আবার 1) নং এ যাবে। এভাবে চলতে থাকবে।

while loop এর গঠনঃ

শুরু

while শর্তঃ

বিবৃতি/কাজ

বৃদ্ধি/হ্রাস

For loop এর গঠনঃ

for ভ্যারিয়েবল in সিকোয়েন্সঃ

বিবৃতি/কাজ

বা

for ভ্যারিয়েবল in range(শুরু, শর্ত, বৃদ্ধি/হ্রাস):

বিবৃতি/কাজ

minhazul_kabir = 1027 এর অর্থ হলো, minhazul_kabir নামক চলক বা variable এর মান 1027 হিসেবে assign করা। '=' এই চিহ্নকে আমরা গণিতে সমান চিহ্ন বলে থাকি। কিন্তু, প্রোগ্রামিং এ '=' কে assignment operator বলা হয়। assign মানে হলো, সাময়িক সংরক্ষণ। যেহেতু, minhazul_kabir এর মান আমরা 1027 যত ইচ্ছা পরিবর্তন করতে পারি, সেজন্য '=' কে assignment operator বলা হয়।

ধরি, আমার কাছে একটা বোতলে 6 (old) ফোঁটা পানি আছে। বোতলের নাম 'i'। এতে আমরা আরও 3 ফোঁটা পানি দিলাম। এখন, বোতলে পানির পরিমাণ 9 (new) ফোঁটা।

$$i = i + 3$$

$$\text{new value} = \text{old value} + 3$$

সাধারণত অংকে, $i = i + 3$ কে কাটাকাটি করে। $0 = 3$ হয়। কিন্তু, '=' প্রোগ্রামিং এ সমান চিহ্ন না হওয়ায় সেক্ষেত্রে এটা কাজ করে না।

প্রোগ্রামাররা অত্যাধিক অলস হওয়ার জন্য, তারা সব কিছু সংক্ষিপ্ত করে লিখেঃ

পূর্ণরূপ	সংক্ষিপ্তরূপ	
$i = i + 1$	$i++$	$i = i + 1$ এর অর্থ হলো i এর পূর্বের মানের সাথে 1 যুক্ত করা।
$i = i + 2$		$i = i + 2$ এর অর্থ হলো i এর পূর্বের মানের সাথে 2 যুক্ত করা।
$i = i + 3$		$i = i + 3$ এর অর্থ হলো i এর পূর্বের মানের সাথে 3 যুক্ত করা।
$i = i + 4$		$i = i + 4$ এর অর্থ হলো i এর পূর্বের মানের সাথে 4 যুক্ত করা।
$i = i + 5$		$i = i + 5$ এর অর্থ হলো i এর পূর্বের মানের সাথে 5 যুক্ত করা।
$i = i + 6$		$i = i + 6$ এর অর্থ হলো i এর পূর্বের মানের সাথে 6 যুক্ত করা।

i = i + 4	i+=4	i = i + 7 এর অর্থ হলো i এর পূর্বের মানের সাথে 7 যুক্ত করা।
i = i + 5	i+=5	i = i + 8 এর অর্থ হলো i এর পূর্বের মানের সাথে 8 যুক্ত করা।
i = i + 6	i+=6	i = i + 9 এর অর্থ হলো i এর পূর্বের মানের সাথে 9 যুক্ত করা।
i = i + 7	i+=7	i = i + 10 এর অর্থ হলো i এর পূর্বের মানের সাথে 10 যুক্ত করা।
i = i + 8	i+=8	i = i + 11 এর অর্থ হলো i এর পূর্বের মানের সাথে 11 যুক্ত করা।

❖ Understanding the while & for loop.

output: 1 2 3 4 5 6 7 8 9 10

<u>While loop</u>	<u>For Loop</u>
<pre>i = 1 while i < 11: print(i) i += 1</pre>	<pre>for i in range(1, 11, 1): print(i)</pre>

শুরুর মান 1,

A.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 1, চেক করা হবে, $1 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 1 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 2</p>
B.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 2, চেক করা হবে, $2 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 2 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 3</p>
C.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 3, চেক করা হবে, $3 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 3 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 4</p>
D.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 4, চেক করা হবে, $4 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 4 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 5</p>
E.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 5, চেক করা হবে, $5 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 5 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 6</p>
F.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 6, চেক করা হবে, $6 < 11$? এটা সত্য, তাই লুপ চলবে।</p> <p>ধাপ ০২ (কাজ করবে): i = 6 প্রিন্ট হবে।</p> <p>ধাপ ০৩ (বৃদ্ধি): i ১ বাড়ানো হবে, এখন i = 7</p>
G.	<p>ধাপ ০১ (শর্ত পরীক্ষা): i = 7, চেক করা হবে, $7 < 11$? এটা সত্য, তাই লুপ চলবে।</p>

	ধাপ ০২ (কাজ করবে): $i = 7$ প্রিন্ট হবে। ধাপ ০৩ (বৃদ্ধি): $i + 1$ বাড়ানো হবে, এখন $i = 8$
H.	ধাপ ০১ (শর্ত পরীক্ষা): $i = 8$, চেক করা হবে, $8 < 11$? এটা সত্য, তাই লুপ চলবে। ধাপ ০২ (কাজ করবে): $i = 8$ প্রিন্ট হবে। ধাপ ০৩ (বৃদ্ধি): $i + 1$ বাড়ানো হবে, এখন $i = 9$
I.	ধাপ ০১ (শর্ত পরীক্ষা): $i = 9$, চেক করা হবে, $9 < 11$? এটা সত্য, তাই লুপ চলবে। ধাপ ০২ (কাজ করবে): $i = 9$ প্রিন্ট হবে। ধাপ ০৩ (বৃদ্ধি): $i + 1$ বাড়ানো হবে, এখন $i = 10$
J.	ধাপ ০১ (শর্ত পরীক্ষা): $i = 10$, চেক করা হবে, $10 < 11$? এটা সত্য, তাই লুপ চলবে। ধাপ ০২ (কাজ করবে): $i = 10$ প্রিন্ট হবে। ধাপ ০৩ (বৃদ্ধি): $i + 1$ বাড়ানো হবে, এখন $i = 11$
K.	ধাপ ০১ (শর্ত পরীক্ষা): $i = 11$, চেক করা হবে, $11 < 11$? এটা মিথ্যা, তাই লুপ বন্ধ হয়ে যাবে।

range(start, stop, step):

Python-এ `range()` একটি built-in ফাংশন যা একটি সিকুয়েন্স (সংখ্যার সিরিজ) তৈরি করতে ব্যবহৃত হয়।
সাধারণত এটি `for loop`-এর সাথে ব্যবহার করা হয় যাতে নির্দিষ্ট সংখ্যক পুনরাবৃত্তি বা ইটারেশন সম্পাদন করা যায়।

range() ফাংশনের গঠন:

`range()` ফাংশন সাধারণত তিনটি আঙ্গমেন্ট গ্রহণ করে:

1. **start (ঐচ্ছিক):** সিকুয়েন্সের প্রথম সংখ্যা (অথবা মান)। এটিতে কোনো মান না লিখলে অর্থাৎ ঐচ্ছিক মান (default: 0)।
2. **stop/condition (আবশ্যিক):** সিকুয়েন্সের শেষ সংখ্যা (এটি অন্তর্ভুক্ত হয় না)। এটি অবশ্যই দিতে হয়।
3. **step/increment,decrement (ঐচ্ছিক):** প্রতি ধাপে(পুনরাবৃত্তিতে) কতটা বাড়ানো হবে অথবা কমানো হবে। এটিতে কোনো মান না লিখলে অর্থাৎ ঐচ্ছিক মান (default: 1)। অর্থাৎ পরবর্তী প্রতি ধাপে মান 1 করে বাড়ানো হবে।

Argument (আঙ্গমেন্ট) কী?

Argument হলো সেই মান বা তথ্য যা একটি ফাংশন বা মেথডে পাস করা হয় যখন ফাংশনটি কল করা হয়। এটি ফাংশনের ভিতরে ব্যবহৃত হয় এবং ফাংশনটির কার্যকারিতা নির্ধারণ করতে সাহায্য করে। সহজ ভাষায়, আঙ্গমেন্ট হলো ফাংশনের ইনপুট যা ফাংশনটি সম্পাদন করতে প্রয়োজনীয়। উদাহরণস্বরূপ, `print()` এবং `input()` ফাংশনে যে মান বা তথ্য আমরা () ব্রাকেটের মধ্যে দেই, তা হলো আঙ্গমেন্ট। () ব্রাকেটের মধ্যে যা লিখবো তা হচ্ছে, আঙ্গমেন্ট।

Argument (আঙ্গমেন্ট) এর বাস্তব উদাহরণ: ধরা যাক, আপনি একটি বইয়ের দোকানে গিয়ে একজন দোকানদারকে বললেন, "দয়া করে আমাকে একটি সায়েন্স বই দিন।"

এখানে, **বইয়ের ধরন** (যেমন: সায়েন্স, রূপকথা, ইতিহাস) হলো আপনার argument, যেটি দোকানদারকে বলে দিচ্ছে আপনি কোন ধরনের বই চান। দোকানদার সেই অনুযায়ী নির্দিষ্ট বইটি আপনার হাতে তুলে দেবে।

এভাবে, argument হলো সেই তথ্য বা মান যা ফাংশনে পাস করা হয়, যাতে ফাংশনটি নির্দিষ্টভাবে কাজ করতে পারে।

Default Value (ডিফল্ট মান) কী?

Default Value হলো এমন একটি মান যা কোনো আর্গুমেন্টের জন্য পূর্বনির্ধারিত থাকে, যদি ব্যবহারকারী কোনো মান প্রদান না করেন। এটি ফাংশনের ডিফল্ট মান হিসেবে কাজ করে। অর্থাৎ, যখন আর্গুমেন্টের জন্য ব্যবহারকারী কিছু নির্দিষ্ট মান দেন না, তখন ফাংশনটি ঐ ডিফল্ট মান ব্যবহার করে। এর মাধ্যমে ফাংশনটি নির্দিষ্ট আর্গুমেন্ট ছাড়া কাজ করতে পারে।

Default Value (ডিফল্ট মান) এর আরেকটি বাস্তব উদাহরণ: ধরা যাক, আপনি একটি হেয়ার সেলুনে গিয়ে বললেন, "আমি হেয়ার কাটিং করতে চাই, কিন্তু যদি আপনি Amla Shampoo ব্যবহার করেন, তাতে কোনো সমস্যা নেই।"

এখানে, Shampoo হলো একটি প্রোডাক্ট যদি আপনি কিছু না বলেন, তবে তারা default Shampoo ব্যবহার করবে। উদাহরণস্বরূপ, সেলুনটি "sunsilk shampoo" ব্যবহারের সিদ্ধান্ত নিয়ে থাকতে পারে।

এভাবে, default value হলো এমন একটি মান, যা ফাংশন বা সিস্টেমে কোনোকিছু লিখা না হলে তার কাছে গচ্ছিত পূর্বনির্ধারিত মান প্রদান করে।

output: 3 5 7 9 11 13 15 17 19 21	
<u>While loop</u>	<u>For Loop</u>
<pre>i = 3 while i < 22: print(i) i += 2</pre>	<pre>for i in range(3, 22, 2): print(i)</pre>

❖ Working with the for & while loop.

আমাকে একবার এক বড় ভাই (হাজি দানেশ CSE) ফেসবুকে messenger এ Hmm লিখে, আমি উনাকে পাল্টা Hmm2 লিখি। পরে উনি আমাকে Hmm3 লিখে। পরে আমি নিম্নোক্ত কোড লিখি এবং output উনাকে পাঠাই।

for loop code:	while loop code:
<pre>for i in range(1, 501, 1): print(f" Hmm{i}")</pre>	<pre>i = 1 while i<501: print(f" Hmm{i}") i+=1</pre>

তুমি এই কোডগুলা চালু করিয়ে তোমার ফ্রেন্ডের দিয়ে মজা করতে পারো। 😊

for loop code:	for loop code:
n = 'bangLadEsh' for x in n: print(x)	word_tuple = "apple", "banana", "cherry", "date", "elderberry", "fig", "grape", "honeydew", "kiwi", "lemon" for x in word_tuple: print(x)
output:	output:
b a n g L a d E s h	apple banana cherry date elderberry fig grape honeydew kiwi lemon

❖ Creating infinite and nested loops.

এই পথ যদি না শেষ হয়, তাহলে কেমন হবে তুমি বলত ? <https://youtu.be/kCB90Rsxn4s> এই গানটা থেকে যদিও loop তৈরি হয় নাই। আমরা চাইলে, এমন প্রোগ্রাম (লুপ) বানাতে পারি, যা সারাজীবন চলবে। কোনোদিন শেষ হবে না।

নিচের দুইটা কোড অবশ্যই তোমার ল্যাপটপ/কম্পিউটারে চালিয়ে কি পেলে জানাবে।

<u>Infinity looooooooooooooop</u>	<u>No Loop</u>
i = 1 while True: print(f" Hmm{i} ") i+=1	i = 1 while False: print(f" Hmm{i} ") i+=1

একটা জিনিস বুবতে পারলে ? while এর পরে Relational/Boolean Operator হয়। Relational/Boolean Operator এর মান শুধুমাত্র হ্টো। True/ False

Nested Loop:

"Nest" মানে হলো বাসা, বিশেষ করে পাথির বাসা। প্রোগ্রামিংয়ে, নেস্টিং হল এক ধরনের কাঠামো যেখানে একটি কোড এর মধ্যে একই ধরনের কিছু লিখা ।

Nested loop সর্বপ্রথম পেটের ভিতরের কাজ শুরু হবে ।

$5 \times 3 = 15$	$3 \times 3 \times 3 = 27$
<pre>qty = ["one", "two", "three", "four", "five", "six"] fruits = ["apple", "banana", "cherry"] for x in qty: for y in fruits: print(x, y)</pre>	<pre>adj = ["one", "two", "three"] fruits = ["apple", "banana", "cherry"] colors = ["red", "green", "blue"] for x in adj: for y in fruits: for z in colors: print(x, y, z)</pre>
one apple one banana one cherry two apple two banana two cherry three apple three banana three cherry four apple four banana four cherry five apple five banana five cherry six apple six banana six cherry	one apple red one apple green one apple blue one banana red one banana green one banana blue one cherry red one cherry green one cherry blue two apple red two apple green two apple blue two banana red two banana green two banana blue two cherry red two cherry green two cherry blue three apple red three apple green three apple blue three banana red three banana green

	three banana blue
	three cherry red
	three cherry green
	three cherry blue

এই কোডটি নিজের Laptop/Desktop এ চালিয়ে দেখো, কি output পেলে।

```
for i in range(1,11):
    for j in range(1,11):
        print(f"{i} * {j} = {i*j}")
print()
```

❖ Using break, continue, and pass statements. Assignment #4: Loop exercises and control flow challenges.

লুপকে ২ ভাবে নিয়ন্ত্রণ করা যায়

- continue (এড়িয়ে যাওয়া)
- break (থামা)

continue:

এই কোডটি নিজের Laptop/Desktop এ চালিয়ে দেখো, কি output পেলে।

```
for i in range(1, 11): # লুপটি 1 থেকে 10 পর্যন্ত চলবে
    if i == 5: # যদি i এর মান 5 হয়
        continue # লুপের বর্তমান ইটারেশন বাদ দিয়ে পরবর্তী ইটারেশনে চলে যাবে
        print(i) # যদি i 5 না হয়, তবে সেই সংখ্যাটি প্রিন্ট করবে
    else: # যদি লুপটি কোন কারণে "break" ছাড়া শেষ হয়, তবে এই অংশটি কার্যকর হবে
        print("You can use else in Loop") # লুপটি কোন কারণে break ছাড়া শেষ হলে এই বার্তা প্রিন্ট হবে
```

1	> এই লুপটি ১ থেকে ১০ পর্যন্ত সংখ্যা পরীক্ষা করবে।
2	> শুধুমাত্র i এর মান ৫ হবে, তখন continue স্টেটমেন্টটি কার্যকর হবে, অর্থাৎ ৫টি সংখ্যা প্রিন্ট করা হবে না এবং লুপ পরবর্তী সংখ্যার দিকে চলে যাবে।
3	> অন্য সব সংখ্যাগুলি প্রিন্ট হবে।
4	> else: — এই else অংশটি তখন কার্যকর হবে যদি লুপটি break ছাড়াই সম্পূর্ণ হয়। অর্থাৎ, লুপটি শেষ হলে (এবং break হয়নি) তখন else এর কোড ব্লক চলবে।
5	এখানে, ৫ বাদ দেয়া হয়েছে, কারণ যখন i == 5 তখন continue স্টেটমেন্ট কার্যকর হয় এবং ৫টি সংখ্যা প্রিন্ট হতে দেয় না।
6	
7	
8	
9	
10	
You can use else in Loop	

break:

এই কোডটি নিজের Laptop/Desktop এ চালিয়ে দেখো, কি output পেলে। (আগে)

```
for i in range(1, 11): # লুপটি 1 থেকে 10 পর্যন্ত চলবে
    if i == 5: # যদি i এর মান 5 হয়
        break # লুপটি তৎক্ষণাত্ম থামিয়ে দেবে
    print(i) # যদি i 5 না হয়, তবে সেই সংখ্যাটি প্রিন্ট করবে
else: # যদি লুপটি কোন কারণে "break" ছাড়া শেষ হয়, তবে এই অংশটি কার্যকর হবে
    print("You can use else in Loop") # লুপটি কোন কারণে break ছাড়া শেষ হলে এই বার্তা প্রিন্ট হবে
```

- | | |
|---|---------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | A. লুপের চলাচল: for i in range(1, 11): লুপটি 1 থেকে 10 পর্যন্ত চলবে। |
| 2 | B. শর্ত পরীক্ষা: যদি i == 5, তাহলে break কমান্ড কার্যকর হবে, যা লুপটিকে তৎক্ষণাত্ম বন্ধ করে দেবে। |
| 3 | C. প্রিন্ট: i এর মান যদি 5 না হয়, তাহলে print(i) কমান্ড দ্বারা সেই সংখ্যা প্রিন্ট হবে। |
| 4 | D. else ব্লক: else ব্লকটি শুধুমাত্র তখন কার্যকর হয় যদি লুপটি break দিয়ে থামে না। এখানে, break ব্যবহার হওয়ায় else ব্লকটি কার্যকর হবে না। |

এই কোডটি নিজের Laptop/Desktop এ চালিয়ে দেখো, কি output পেলে। (পরে)

```
for i in range(1, 11): # লুপটি 1 থেকে 10 পর্যন্ত চলবে
    print(i) # প্রতিটি সংখ্যাকে প্রিন্ট করবে
    if i == 5: # যদি i এর মান 5 হয়
        break # লুপটি তৎক্ষণাত্ম থামিয়ে দেবে
else: # যদি লুপটি কোন কারণে "break" ছাড়া শেষ হয়, তবে এই অংশটি কার্যকর হবে
    print("You can use else in Loop") # লুপটি কোন কারণে break ছাড়া শেষ হলে এই বার্তা প্রিন্ট হবে
```

- | | |
|---|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | A. লুপের চলাচল: for i in range(1, 11): লুপটি 1 থেকে 10 পর্যন্ত চলবে। |
| 2 | B. প্রিন্ট: প্রতিটি i এর মান print(i) দ্বারা প্রিন্ট হবে। |
| 3 | C. শর্ত পরীক্ষা: if i == 5: যখন i এর মান 5 হয়, তখন break কমান্ড কার্যকর হয় এবং লুপটি তৎক্ষণাত্ম থামে যায়। |
| 4 | |
| 5 | D. else ব্লক: else ব্লকটি শুধুমাত্র তখন কার্যকর হয়, যদি লুপটি break ছাড়া শেষ হয়। কিন্তু এখানে break ব্যবহার হয়েছে, তাই else ব্লকটি কার্যকর হবে না। |

break আগে পেলে আগে থামবে। (1,2,3,4) break পরে পেলে পরে থামবে। (1,2,3,4,5)

One Drive এর নতুন URL: https://studentsust-my.sharepoint.com/:f/g/personal/kabir_student_sust_edu/EkUh22MA2sBGIWISf-K6PJgBRbqv8ThqQJcqapHmYIaKHw?e=6fthQD