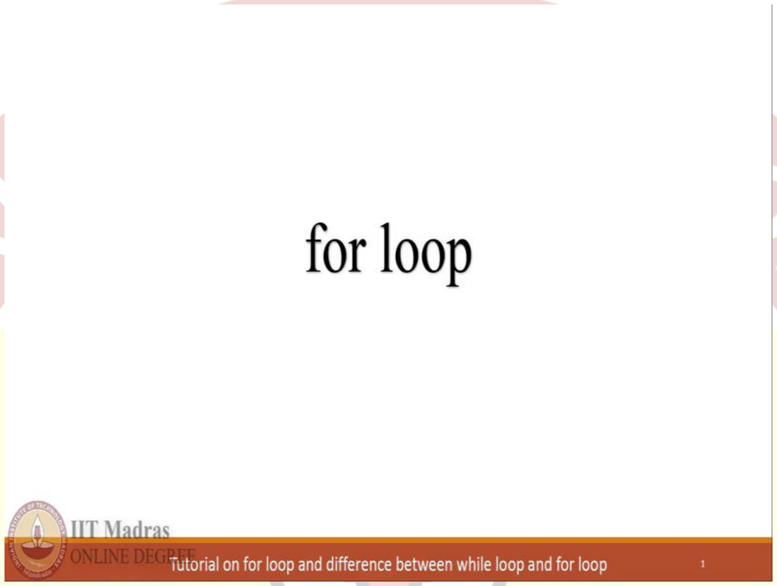
IIT Madras

ONLINE DEGREE

Programming in Python
Professor. Sudarshan Iyengar
Department of Computer Science and Engineering
Indian Institute of Technology, Ropar
Tutorial on for loop and difference between while loop and for loop

(Refer Slide Time: 0:15)

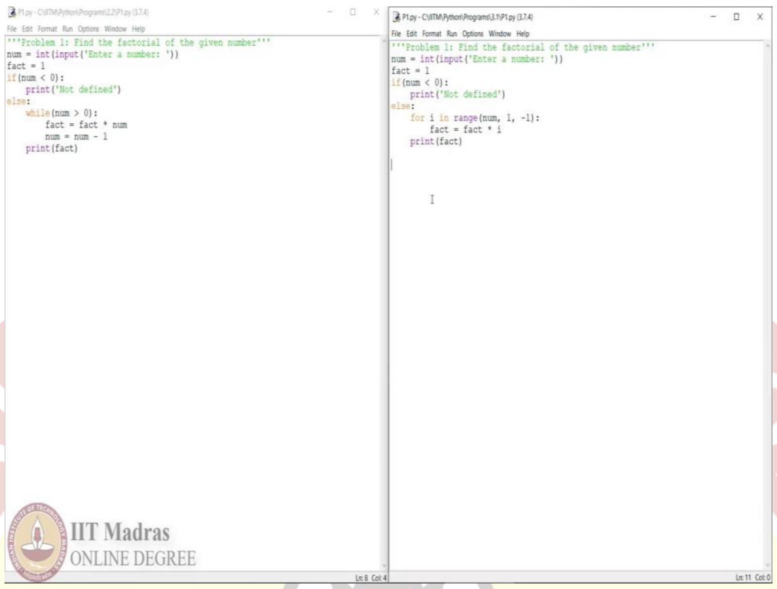


for loop

IIT Madras
ONLINE DEGREE
Tutorial on for loop and difference between while loop and for loop
1

Hello Python students. In last tutorial, we saw while loop and some examples using while loop. In this tutorial, we will see another type of loop called for loop, for loop has its own advantages as compared to while loop, we will use some sample examples and try to elaborate how for loop works using range function and the functionality which is referred as for each from the given sequence, we will use the same examples which we saw with while loop and try to convert them into Python codes using for loop.

(Refer Slide Time: 0:59)


P1.py - C:\IITM\Python\Programs\2.2\P1.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 1: Find the factorial of the given number'''
num = int(input('Enter a number: '))
fact = 1
if(num < 0):
    print('Not defined')
else:
    while(num > 0):
        fact = fact * num
        num = num - 1
    print(fact)
```

P1.py - C:\IITM\Python\Programs\3.1\P1.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 1: Find the factorial of the given number'''
num = int(input('Enter a number: '))
fact = 1
if(num < 0):
    print('Not defined')
else:
    for i in range(num, 1, -1):
        fact = fact * i
    print(fact)
```

IIT Madras
ONLINE DEGREE
Ln: 8 Col: 4
Ln: 11 Col: 0

This is the first example, which we saw, which finds the factorial of the given number using while loop. Now, we will try to convert this particular while loop into its equivalent for loop in order to achieve seen output, which is factorial of the given number. Let us try to convert it. Now, as you can see, here, the left side program is using while loop whereas, the right hand side program is using for, loop the rest of the code is same.

Now, the difference over here is using while loop we are reducing the value of number by 1 in every iteration, as we do the factorial into number. Same thing we are doing here as well, but over here in for loop, we do not have to write this particular statement number is equal to number minus 1, because that is been taken care of range function.

Here, the range function has three parameters first, the number that is the starting point, then this one which is the ending point and the step count, that means the value of number will start from the given input and then it will do minus 1 for every iteration till it reaches to 1 and the updated value will be stored in variable i which we are using to multiply with variable factorial.

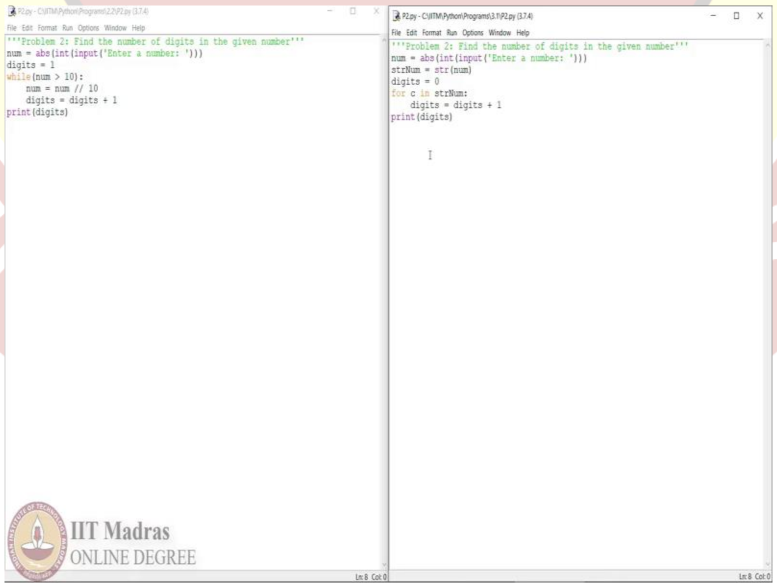
Once the entire for loop execution is finished, we are printing the result which is stored in variable fact. As you can see, both programs using while loop as well as for loop looks identical, but they have a little bit of difference, as a programmer we have to decide which loop is more ideal for this particular example.

Now, the question is how to decide that what are the factors which we should consider before deciding whether to use while loop or for loop? So, the most important factor we should consider is whether do we know the range of the loop are not, as in are we aware in advance that the loop is going to run for some n number of times, if we are certain that the given loop is going to execute for let us say 10 times or 12 times then we should opt for for loop.

Whereas, if we do not know how many number of times the particular loop is going to execute, we should opt for while loop. This is important because whenever we use for loop we have to provide this range function and inside that range function, we should also mention the starting point of the loop and the ending point of the loop. In order to provide these counts, we should know how many number of times the loop will execute.

Hence, for is more suitable when we know the total number of iterations required in that particular program, whereas while is more generic way of execution. As we can see, this is a problem statement where we are trying to find the factorial of a number and in such a case, the number of iterations are known to us. Hence, for loop is more suited option than a while loop. Similarly, we will try to convert the other problem statements from previous tutorial. Let us see the second problem statement.

(Refer Slide Time: 4:57)


P2.py - C:\IITM\Python\Programs\2.2\P2.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 2: Find the number of digits in the given number'''
num = abs(int(input('Enter a number: ')))
digits = 1
while(num > 10):
    num = num // 10
    digits = digits + 1
print(digits)
```

P2.py - C:\IITM\Python\Programs\3.1\P2.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 2: Find the number of digits in the given number'''
num = abs(int(input('Enter a number: ')))
strNum = str(num)
digits = 0
for c in strNum:
    digits = digits + 1
print(digits)
```

IIT Madras
ONLINE DEGREE
Ln: 8 Col: 0
Ln: 8 Col: 0

Find the number of digits in the given number. In this program, we are calculating the number of digits from the entered number. In this particular case, the number of iterations cannot be predicted, because the number entered by user can be a single digit number or it can have n number of digits. Hence, we cannot predict how many number of times this particular while loop will execute.

In such a case, it is not possible to convert this code into its equivalent for loop code. Because, as we mentioned earlier, for loop uses a function called range and that function requires the starting point and the ending point, which is not known at this particular stage in this example. But if you remember, I mentioned for loop has one more advantage for each from the given sequence of elements. If we use that particular feature of for loop, then we might have a workaround solution to this particular problem. It may not be the ideal solution, but it is possible.

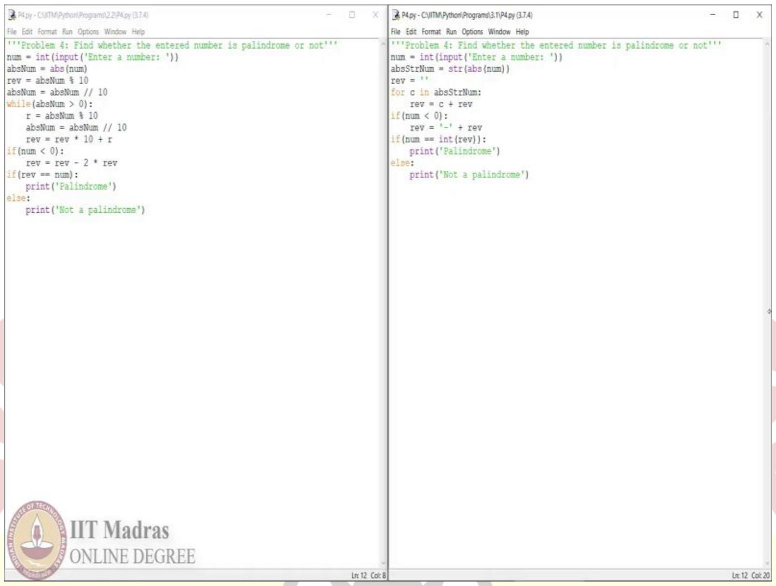
Now, let us see how that can be done. Let us look at this particular Python code using for loop line by line. First, we accepted the number from the user as exactly as we did in the previous code. After that, we have done something new over here, which is conversion of that number into its equivalent string. The function str will convert this particular number into its equivalent string form, and it will get stored in variable str num.

For example, if we have a number 1234, then its string equivalent representation will have four different characters first characters will be 1, second is 2, third is 3, fourth is 4. For loop can be used to iterate over such a string representation using the feature I mentioned earlier called for each. Now, let us see how that works.

For C as in for each character C. In str num, do digits is equal to digits plus 1. As in, for every character in str num, a C will have one value, which means in first iteration, the value for C will be 1, in second iteration value for C will be 2. In third iteration, the value for C will be 3. And for fourth iteration, the value in C will be 4, because that is the string value we are storing in this particular variable.

As the iterations are happening, we will increment the variable digits. And once the loop is finished, we will print the value digits. As I mentioned earlier, this is not the ideal version of writing Python code for this problem statement. But still, it is possible using for loop. Now, let us move on to the next problem statement which we saw earlier.

(Refer Slide Time: 8:44)


P4.py - C:\IITM\Python\Programs\2.2\P4.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 4: Find whether the entered number is palindrome or not'''
num = int(input('Enter a number: '))
absNum = abs(num)
rev = absNum % 10
absNum = absNum // 10
while(absNum > 0):
    r = absNum % 10
    absNum = absNum // 10
    rev = rev * 10 + r
if(num < 0):
    rev = rev - 2 * rev
if(rev == num):
    print('Palindrome')
else:
    print('Not a palindrome')
```

P4.py - C:\IITM\Python\Programs\3.1\P4.py (3.7.4)
File Edit Format Run Options Window Help

```
'''Problem 4: Find whether the entered number is palindrome or not'''
num = int(input('Enter a number: '))
absStrNum = str(abs(num))
rev = ''
for c in absStrNum:
    rev = c + rev
if(num < 0):
    rev = '-' + rev
if(num == int(rev)):
    print('Palindrome')
else:
    print('Not a palindrome')
```

Third problem statement reverse the digits in the given number. In this example, as well, we cannot predict how many number of times this particular while loop will execute. Hence, the range function cannot be defined. Hence, we have to opt for second feature of for, which is for each. In this case as well, we have to convert the given number into its equivalent string representation as we did in the previous example.

Now, let us look at this particular Python code using for loop. We accepted the number from the user then we converted the absolute value of that number into its string form. We defined once reverse variable which if you see over here it is a empty string. As we are doing all operations using strings. The output which is stored in a reverse variable will also be a string.

Then we will iterate over this particular variable which stores the string using characters C, every time we get a new character, we are going to append that character along with a reverse. For example, if our input number is 1234, then this absolute string will hold a string with four characters 1234. And in first iteration, the variable C will hold the first character from that string, which is 1, then we will do reverse is equal to C plus reverse this particular operation will concatenate C, which is 1 and reverse which is empty.

Hence, the value which is stored in variable reverse will be 1. In second iteration value of variable C will be two, then we will do 2 plus reverse which is 1, hence, to concatenation, 1 will become 2 1, next value for C will be 3. So, we will do 3 plus 2 1, which will result into 3 2 1 and then in last iteration C will hold value 4, and we will do 4 plus 3 2 1, which will result into 4321, which is the expected output.

Now, in order to preserve the sign of the original number, we will check whether the number is greater than equal to 0 or not. If the original entered number is greater than or equal to 0, we will simply print the variable reverse or else if it is a negative number, then we will append a negative sign, and then we will print the string representation of the reversed number. Once again, this is not the ideal way of writing the Python code. The most suitable program for this particular problem statement is using while loop which is on the left side of the screen.

Now, let us look at the fourth example which we saw earlier. Problem 4, find whether the entered number is palindrome or not, in case of palindrome program, we actually use the previous program, which was reversing the digits in the number and then we compared those two numbers and if they are equal, we call it a palindrome and if not, then we called it not a palindrome.

Hence, in this case as well, we cannot predict how many number of iterations are required in order to reverse that number. Hence, it is not possible to define the function for for loop. Therefore, again we will convert the input numbers into its equivalent string representation and then try to execute using for each feature of the for loop.

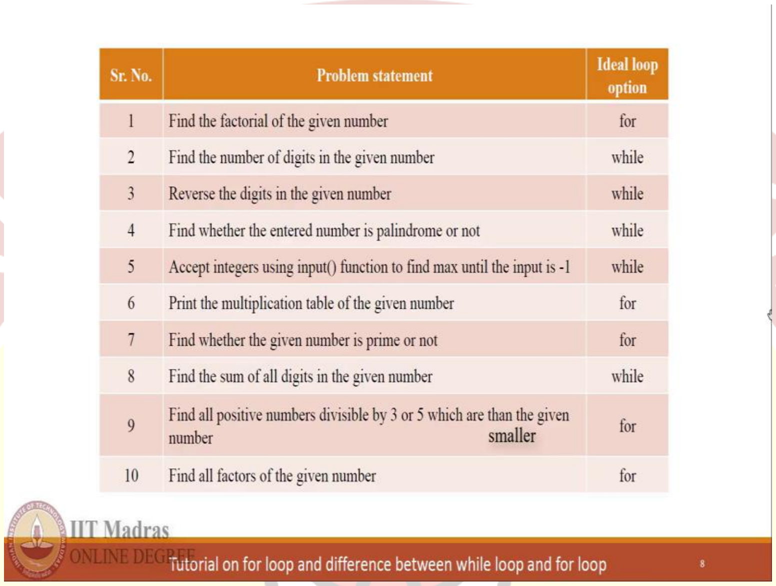
The right side program is doing that, it is converting the input number into a string representation, then using for each functionality of for loop, we are reversing the characters in the given representation of the string number, then we will check whether the original number is less than 0 or not. If that is the case, we will append a negative sign before the number, then we will compare the original number with the reverse number and if they are equal, then we will print palindrome else not a palindrome.

If you look at this particular if condition, there is a small change as compared to previous code, which we did using while loop we are comparing the original number with its reverse number directly. Whereas, over here we are converting the reverse number into its equivalent integer form. Then we are comparing because, as we can see the reversed variable is defined as string and you cannot compare a number with string.

Hence, we have to convert that string back to its equivalent integer form, which is the opposite operation of what we did over here. Then the equality check operation will execute correctly.

And we will get the accurate result, whether it is palindrome or not, similar like previous to problem statements, this particular code using for loop is also not a ideal representation of this particular problem statement. Hence, it is better to write this program using a while loop.

(Refer Slide Time: 15:29)



Sr. No.	Problem statement	Ideal loop option
1	Find the factorial of the given number	for
2	Find the number of digits in the given number	while
3	Reverse the digits in the given number	while
4	Find whether the entered number is palindrome or not	while
5	Accept integers using input() function to find max until the input is -1	while
6	Print the multiplication table of the given number	for
7	Find whether the given number is prime or not	for
8	Find the sum of all digits in the given number	while
9	Find all positive numbers divisible by 3 or 5 which are than the given number smaller	for
10	Find all factors of the given number	for

IIT Madras
ONLINE DEGREE
Tutorial on for loop and difference between while loop and for loop
8

Based on these examples, and the analysis what we did just now, we can say, finding the factorial of the given number can be done using for loop, whereas it is recommended to use a while loop in order to solve the remaining three problem statements. Now, let us consider few more problem statements. And without writing a Python code, we will try to analyze which loop is more suitable for that particular Python program.

Fifth problem statement accept integers using input function to find max until the input is minus 1. In this case, we are accepting inputs from the user until we get minus 1, here we cannot predict how many numbers a user will enter before minus 1. Hence, it is not possible to define the range function of the for loop. Hence, the ideal loop for this particular problem statement is while.

Next, print the multiplication table of the given number. For example, if the user enters number 8, and we have to print the multiplication table of 8, then we simply have to multiply the number 8 with 1, then 2, then 3 till 10. That means, a range function can be defined over here from 1 to 10. Therefore, for loop is more ideal option.

Seventh problem statement find whether the given number is prime or not. In order to find whether the number is prime or not, we have to divide the number n with every other number smaller than n. And if the modulo operator returns remainder 0 that means there exists at least one number which is dividing the number n perfectly, which means the number is not prime. If we cannot find even a single number less than n, which can divide n perfectly, then we can term that number as prime. Hence, the range function for this particular problem statement can be defined from 1 to n. Therefore, once again for loop is more ideal option.

Next problem statement, find the sum of all digits in the given number. In this problem statement, we have to add each digit from the given number to find that sum. As we have seen earlier, we cannot predict how many number of digits will be there in the entered number. Therefore, it is not possible to predict the range function. Hence, the suitable option is while loop.

Find all positive numbers divisible by 3 or 5, which are the given number. In this case, we have to check all numbers from 1 to n, n being the given number from the user. Hence, it is easy to define the range function. Therefore, we will opt for for loop. Last problem statement, find all factors of the given number. In order to find the factors of any given number, we have to divide that particular number using all the numbers less than the number itself.

In such a case, once again it is easy to define the range function, hence we will go with for loop. Based on this analysis, one can choose between while and for loop depending on how the problem statement is getting executed. Thank you for watching this tutorial. Happy learning.