

List numbers one by one

```

10 # Get input from the user
11 number = int(input("Please enter a number 7 to 18 digits: "))
12 # Determine the number of digits in the integer
13 num_digits = len(str(number))
14 # Ensure the number is between 7 to 18 digits
15 if 7 <= num_digits <= 18:
16     # Loop to print out each digit
17     for i in range(num_digits):
18         # Print the digit
19         print(num_digits - num_digits, "10 ** num_digits")
20 # Calculate the power of 10 for the current position
21 power_of_10 = 10 ** (num_digits - i)
22 # Print the power of 10, the number, the number // power_of_10, number // power_of_10
23 print(power_of_10, "power_of_10", "number", "number // power_of_10", "number // power_of_10")
24 # Extract the leftmost digit
25 digit = number // power_of_10
26 # Print the digit
27 print(digit)
28 # Remove the leftmost digit from the number
29 number %= power_of_10
30 # Decrease the number of digits left to process
31 num_digits -= 1
32 else:
33     print("Number was not 7 to 18 digits.")

```

sentinel loop

```

114 # Initialize total miles and total gallons to zero
115 total_miles = 0
116 total_gallons = 0
117 overall_avg = 0
118 # Start the sentinel-controlled loop, meaning it repeats until it is broken by user input
119 while True:
120     # Get the gallons used from the user, convert to float
121     gallons = float(input("Please enter the gallons used (-1 to stop the loop): "))
122     # Check for the sentinel value and if so escape
123     if gallons == -1:
124         break
125     # Get the miles driven from the user
126     miles = float(input("Please enter the miles driven: "))
127     # Calculate miles per gallon for this tankful
128     mpg = miles / gallons
129     print("The miles/gallon for this tank was {:.1f}.".format(mpg))
130 # Update the total miles and total gallons for avg calculation later
131 total_miles += miles
132 total_gallons += gallons
133 # Calculate the overall average miles per gallon
134 if total_gallons != 0:
135     overall_avg = total_miles / total_gallons
136 print("The overall average miles/gallon was {:.1f}.".format(overall_avg))

```

palindrome

```

149 # 3.12 (Palindromes) A palindrome is a number, word or text phrase that reads the same
150 # Get input from the user
151 number = int(input("Please enter a five-digit integer: "))
152 # Ensure the number is five digits
153 if 10000 <= number <= 99999:
154     # Extract the digits
155     digit1 = number // 10000
156     digit2 = (number // 1000) % 10
157     digit3 = (number // 100) % 10
158     digit4 = (number // 10) % 10
159     digit5 = number % 10
160     # Check if the number is a palindrome
161     if digit1 == digit5 and digit2 == digit4:
162         print("{} is a palindrome.".format(number))
163     else:
164         print("{} is not a palindrome.".format(number))
165 else:
166     print("Please enter a five-digit integer.")

```

consecutives in list

```

190 i = 1
191 total = 0
192 consecutive_3_14 = 0
193 consecutive_3_141 = 0
194 iteration_for_3_14 = 0
195 iteration_for_3_141 = 0
196
197 for denominator in range(1, 6000, 2):
198     if i % 2 == 0:
199         total += 4/denominator
200     else:
201         total -= 4/denominator
202
203 # Check for consecutive 3.14 approximations
204 if str(total)[4] == '3.14':
205     consecutive_3_14 += 1
206     if consecutive_3_14 == 2 and not iteration_for_3_14:
207         iteration_for_3_14 = i
208         print('hit 3.14', i)
209     else:
210         consecutive_3_14 = 0
211
212 # Check for consecutive 3.141 approximations
213 if str(total)[5] == '3.141':
214     consecutive_3_141 += 1
215     if consecutive_3_141 == 2 and not iteration_for_3_141:
216         iteration_for_3_141 = i
217         print('hit 3.141', i)
218     else:
219         consecutive_3_141 = 0
220
221 # Print the results
222 print('i:', i, 'total:', total)
223 i += 1

```

splicing

```

18 print("subset_grades:", subset_grades)
19 # Just the last element
20 subset2 = subset_grades[-1:]
21 print("subset2:", subset2)
22
23 # reversing our list with splicing
24 grades_rev = grades[::-1]
25 print("grades_rev:", grades_rev)
26 # return everything except for the last two elements
27 subset3 = grades[:-2]
28 print("subset3:", subset3)
29
30 # return every other number
31 every_other_grade = grades[::2]

```

moving avg

```

100 # Moving average
101 # Input: list of numbers
102 # Output: list of moving averages
103 # Example: [1, 2, 3, 4, 5] -> [2, 2.5, 3, 3.5, 4]

```

arithmetic

```

100 # Arithmetic
101 # Input: list of numbers
102 # Output: list of arithmetic results
103 # Example: [1, 2, 3, 4, 5] -> [2, 4, 6, 8, 10]

```

pi

```

100 # Pi
101 # Input: list of numbers
102 # Output: list of pi values
103 # Example: [1, 2, 3, 4, 5] -> [3.14, 6.28, 9.42, 12.56, 15.7]

```

odd or even

```

100 # Odd or even
101 # Input: list of numbers
102 # Output: list of odd or even results
103 # Example: [1, 2, 3, 4, 5] -> [True, False, True, False, True]

```

multiples

```

100 # Multiples
101 # Input: list of numbers
102 # Output: list of multiples
103 # Example: [1, 2, 3, 4, 5] -> [2, 4, 6, 8, 10]

```

formatting

```

100 # Formatting
101 # Input: list of numbers
102 # Output: list of formatted strings
103 # Example: [1, 2, 3, 4, 5] -> ['1', '2', '3', '4', '5']

```

By chase2981

Published 10th June, 2025.
Last updated 10th June, 2025.
Page 1 of 1.

Sponsored by [Readable.com](https://readable.com)
Measure your website readability!
<https://readable.com>

cheatography.com/chase2981/