Introduction to Python

Lesson 1: Introduction

Ratthaprom PROMKAM, Dr.rer.nat

Department of Mathematics
Faculty of Science and Technology, RMUTT

Lesson Outline

Python Prompt

Verification of Python Installation:

```
>_ python --version
Python 3.10.5
```

Activating Python Prompt:

```
>_ python
Python 3.10.5
Type "help", "copyright", "credits" or "license" for more ...
>>>
```

Python Prompt

```
>>> 334 + 3.1415
337.1415
>>> 12345679 * 81
999999999
>>> 22 / 7
3.142857142857143
>>> (4 - 2) ** 10
1024
>>> 87 // 4
21
>>> 87 % 4
3
>>> exit()
```

Printing to Console

```
>>> print('Hello, World!')
Hello, World!
>>> print('This year is', 2022)
This year is 2022
>>> print('I am', 20, 'years old')
I am 20 years old
>>> print("Nice to meet you")
Nice to meet you
>>> print('12345', '6789')
12345 6789
```

Variables

Arithmetic Statements

```
3.1415 * (3.45 ** 2), 'Hello', ...
```

↓

Variable Declaration

```
variable = data
```

↓

Printing to Console

```
print('The variable is', variable)
```

Variable Declaration

Number

```
3.1415 * (3.45 ** 2)
```

```
variable = data
```

Variable Name

Text

```
'Hello'
```

Using Variables

```
>>> width = 30
>>> height = 40
>>> area = width * height
>>> print('Area is', area)
Area is 1200
```

Python Scripts

area_rectangle.py

```python
width = 30
height = 40
area = width * height
print('Area is', area)
```

Activate Python

Python Script (*.py)

```
>_ python area_rectangle.py
Area is 1200
```

Commenting Python Code

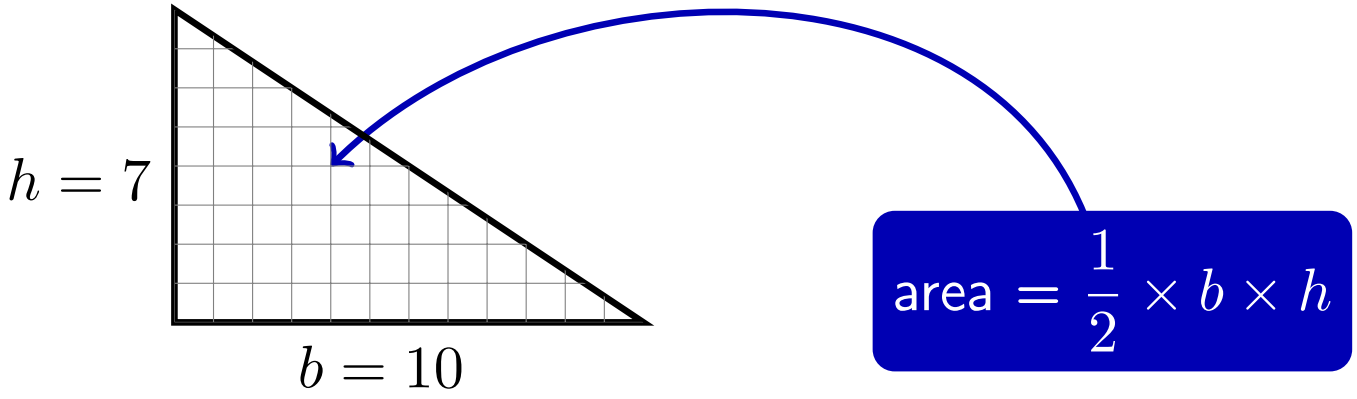
hello_world.py

```python
print('Hello, World!')
# print('This is printed with Python')
year = 2022
print('This year is', year)
# print('It is awesome')
```

This is comments!

```
>_ python hello_world.py
Hello, World!
This year is 2022
```

Example



area_triangle.py

```python
# Calculate area of a right triangle
base = 10
height = 7
area = 0.5 * base * height
print('Area of this triangle is', area)
```

```
>_ python area_triangle.py
Area of this triangle is 35
```

Basic Data Types

Class	Examples	Data Type
`int`	`4, -290, 65_000_000, ...`	Integer Numbers
`float`	`4.0, -0.29, 15.458, 3.145E2, 1.22e-3 ...`	Floating Point Numbers
`complex`	`3+2j, -2+0.5j, -3.14j, -5.1+3.114e2j, ...`	Complex Numbers
`str`	`'Hello', "World", '2022', 's', '*!4aP(&^4)', ...`	Strings
`bool`	`True, False`	Boolean

Basic Data Types

The type of a Python object determines what kind of object it is; every object has a type.

```
>>> type(-255)
<class 'int'>
>>> type(3.1415E5)
<class 'float'>
>>> type(3.0-16.5j)
<class 'complex'>
>>> type(True)
<class 'bool'>
```

Data Operations: Numbers

Operation	Operator	Example	Result
Addition	`+`	`123 + 45.5`	`168.5`
Subtraction	`-`	`123 - 45.5`	`77.5`
Multiplication	`*`	`123 * 45.5`	`5571.9`
Exponentiation	`**`	`123 ** 4`	`228886641`
Division	`/`	`123 / 45.5`	`2.7032967`
Floor Division	`//`	`123 / 45`	`2`
Modulus	`%`	`123 % 45`	`32`

Data Operations: Strings

Operation	Operator	Example	Result
Addition	`+`	`'Hello' + 'World'`	`'HelloWorld'`
		`'Hello' + 2022`	☹ `TypeError`
Multiplication	`*`	`3 * 'Hello'`	`'HelloHelloHello'`
		`'*' * 10`	`'**********'`
		`'World' * 10.45`	☹ `TypeError`
		`'Hello' * 'World'`	☹ `TypeError`

Input Function

Python `input()` function takes user keyboard input. It returns the user input in form of a `string` data type.

```
>>> name = input('Enter your name: ')
Enter your name: John Wick
>>> print(name)
John Wick
>>> type(name)
<class 'str'>
```

Example

ask_name.py

```python
print('This is how input function works')
name = input('Enter your name: ')
age = input('Enter your age: ')
print('Your name is', name, ', and you are', age, 'years old.')
```

```
>_ python ask_name.py
This is how input function works
Enter your name: John Wick
Enter your age: 42
Your name is John Wick , and you are 42 years old.
```

Type Casting

```
>>> x = '123456'
>>> type(x)
<class 'str'>
>>> y = int(x)
>>> type(y)
<class 'int'>
>>> print(y + 1)
123457
```

String Literal

Integer Casting

Type Casting

```
>>> x = '123.456'
>>> type(x)
<class 'str'>
>>> y = float(x)
>>> type(y)
<class 'float'>
>>> print(y + 1)
124.456
```

String Literal

Floating Point Casting

Type Casting

```
>>> x = 3.1414
>>> type(x)
<class 'float'>
>>> y = str(x)
>>> type(y)
<class 'str'>
>>> print(y + '55555')
3.141455555
```

Floating Point Literal

String Casting

Boolean Operators

Operation	Operator	Example	Result
Conjunction	`and`	`True and True`	`True`
		`True and False`	`False`
		`False and True`	`False`
		`False and False`	`False`
Disjunction	`or`	`True or True`	`True`
		`True or False`	`True`
		`False and True`	`True`
		`False and False`	`False`
Negation	`not`	`not True`	`False`
		`not False`	`True`

Comparison Operators

Operation	Operator	Example	Result
Equality	`==`	`5 == 5.0`	`True`
		`-1.255 == -1.25`	`False`
		`'Hello' == 'Hello '`	`False`
		`'A' == 'a'`	`False`
Inequality	`!=`	`5 != 5.0`	`False`
		`-1.255 != -1.25`	`True`
		`'Hello' != 'Hello '`	`True`
		`'A' != 'a'`	`True`

Comparison Operators

Operation	Operator	Example	Result
Equality	`==`	`5 == 5.0`	`True`
		`-1.255 == -1.25`	`False`
		`'Hello' == 'Hello '`	`False`
		`'A' == 'a'`	`False`
Inequality	`!=`	`5 != 5.0`	`False`
		`-1.255 != -1.25`	`True`
		`'Hello' != 'Hello '`	`True`
		`'A' != 'a'`	`True`

```
not (5 == 5.0)
```

Comparison Operators

Operation	Operator	Example	Result
Strict Ordinalities	`<`	`5 < 5.14`	`True`
(Less than)		`-3.50 < -4.59`	`False`
		`1 < 1`	`False`
		`-1 < 0 < 1`	`True`
		`'a' < 'b'`	`True`
		`'BNK' < 'BNA'`	`False`

Comparison Operators

Operation	Operator	Example	Result
Strict Ordinalities	`<`	`5 < 5.14`	`True`
(Less than)		`-3.50 < -4.59`	`False`
		`1 < 1`	`False`
		`-1 < 0 < 1`	`True`
		`'a' < 'b'`	`True`
		`'BNK' < 'BNA'`	`False`

```
(-1 < 0) and (0 < 1)
```

Comparison Operators

Operation	Operator	Example	Result
Strict Ordinalities	`>`	`5 > 5.14`	`False`
(Greater than)		`-3.50 > -4.59`	`True`
		`1 > 1`	`False`
		`999 > 99 > 9`	`True`
		`'a' > 'b'`	`False`
		`'BNK' > 'BNA'`	`True`

Comparison Operators

Operation	Operator	Example	Result
Ordinalities	`<=`	`5 <= 5.14`	`True`
(Less than or equal to)		`-3.50 <= -4.59`	`False`
		`1 <= 1`	`True`
		`-1 <= 0 <= 1`	`True`
		`'a' <= 'b'`	`True`
		`'BNK' <= 'BNA'`	`False`

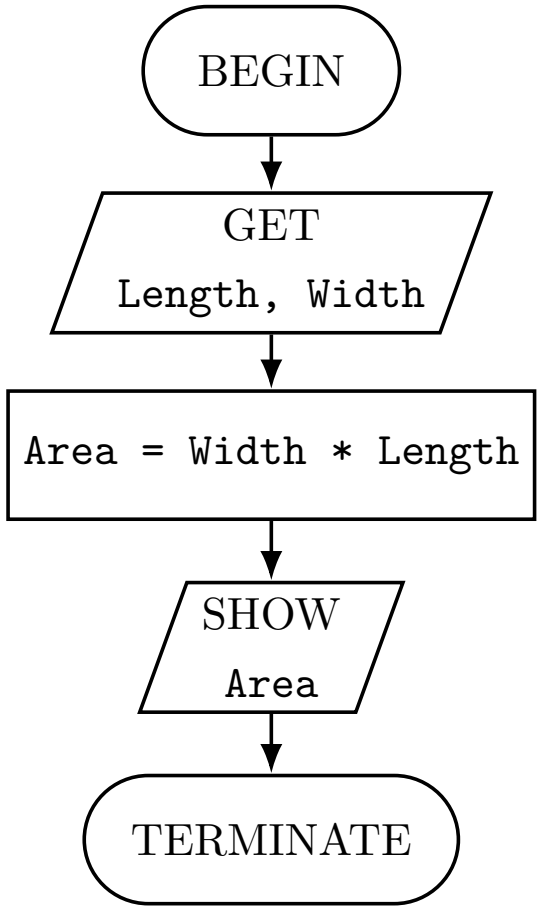
Comparison Operators

Operation	Operator	Example	Result
Ordinalities	`>=`	`5 >= 5.14`	`False`
(Greater than or equal to)		`-3.50 >= -4.59`	`True`
		`1 >= 1`	`True`
		`999 >= 99 >= 9`	`True`
		`'a' >= 'b'`	`False`
		`'BNK' >= 'BNA'`	`True`

Procedural Programming

rectangle_calculator.py

```python
Length = float(input('Enter the length: '))
Width = float(input('Enter the width: '))
Area = Width * Length
print('Area is', Area)
```

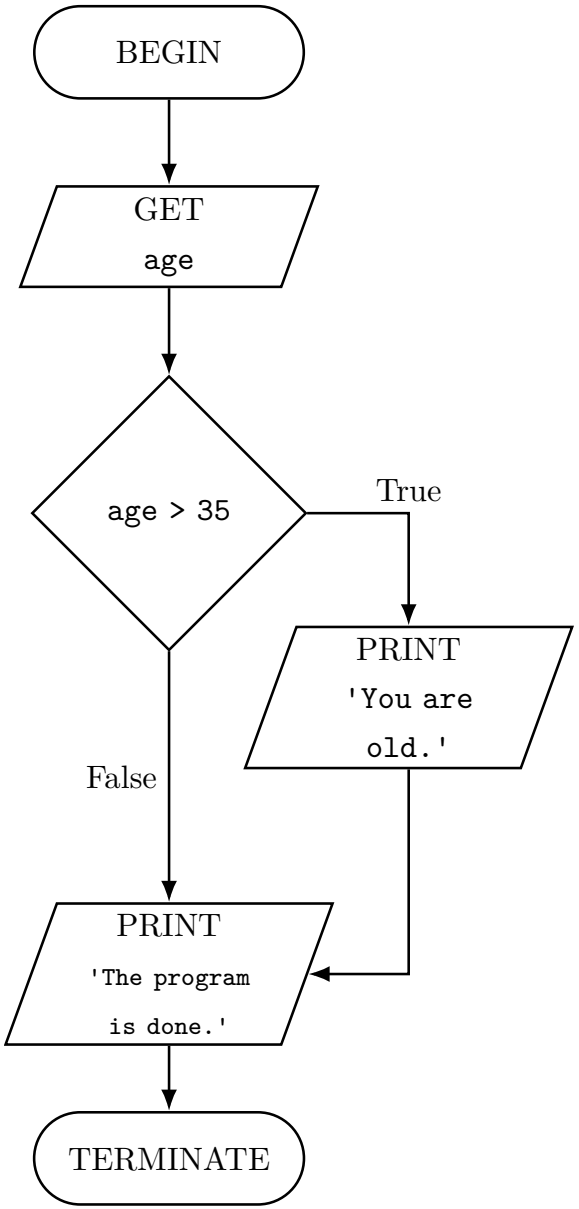


Condition Control Statements

ask_age.py

```
age = float(input('How old are you? '))
if age > 35:
    print('You are old.')
print('The program is done.')
```

```
>_ python ask_age.py
How old are you? 42
You are old.
The program is done.
```

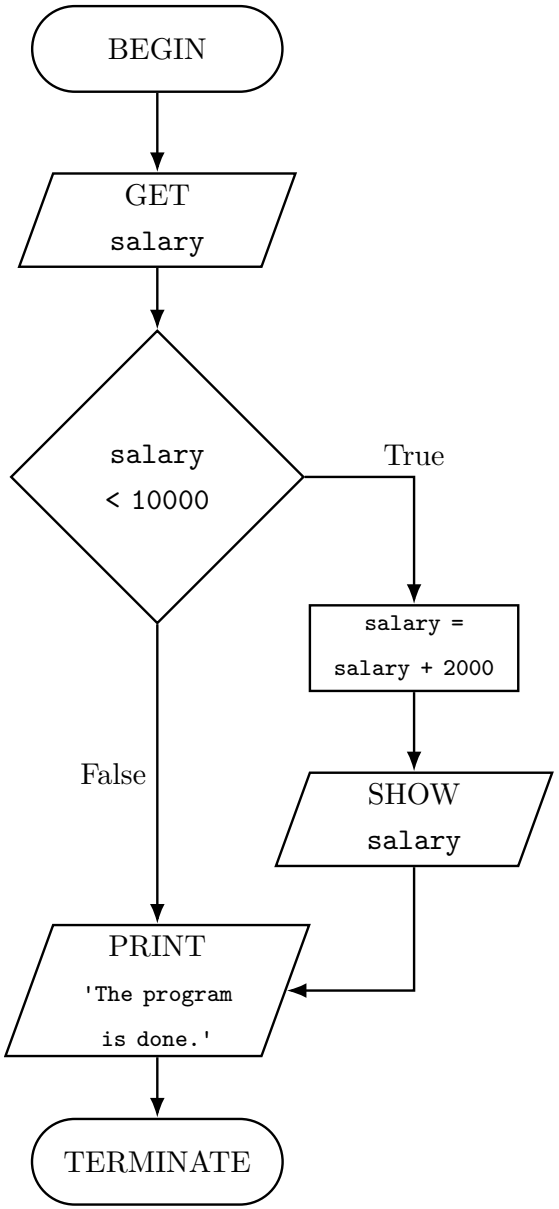


Condition Control Statements

ask_salary.py

```
salary = float(input('Enter the salary: '))
if salary < 10000:
    salary = salary + 2000
    print('Now, your salary is', salary)
print('The program is done.')
```

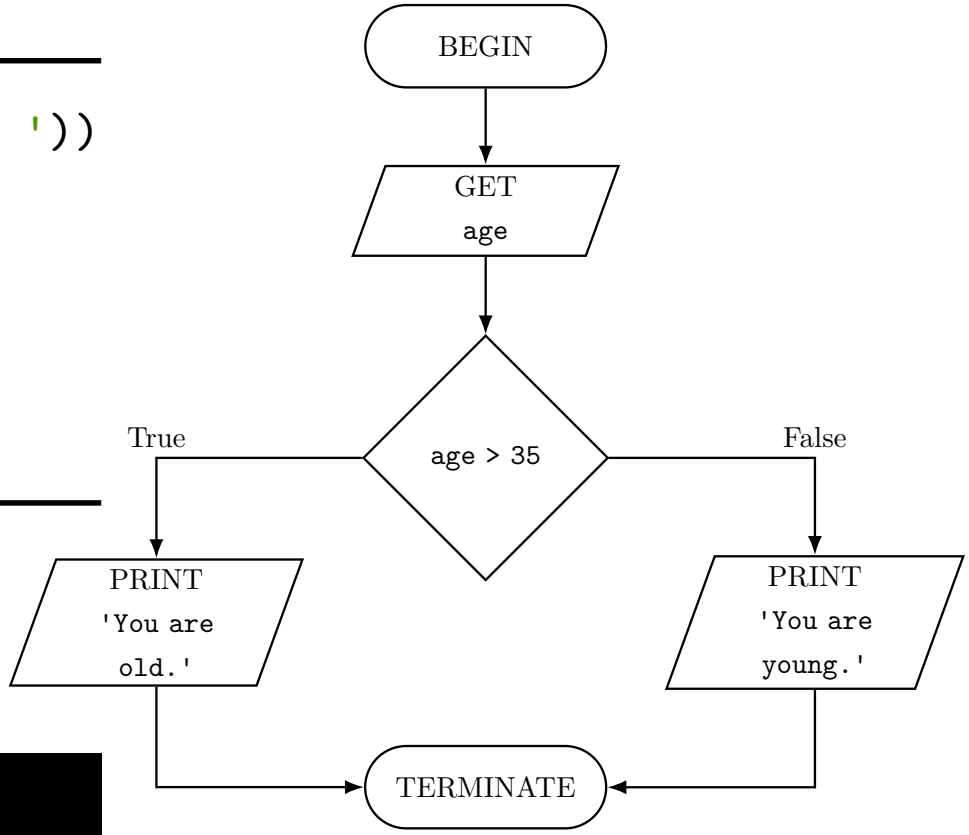
```
>_ python ask_salary.py
Enter the salary: 7500
Now, your salary is 9500
The program is done.
```



Condition Control Statements

ask_age_v2.py

```
age = float(input('How old are you? '))
if age > 35:
    print('You are old.')
else:
    print('You are young.')
```




```
>_ python ask_age_v2.py
How old are you? 35
You are young.
```

Condition Control Statements

ch_size.py

```python
chest = float(input('Chest length? '))
if chest <= 34:
    print('Size = XS')
elif chest <= 36:
    print('Size = S')
elif chest <= 38:
    print('Size = M')
elif chest <= 40:
    print('Size = L')
else:
    print('Size = XL')
```

```
>_ python ch_size.py
Chest length? 24
Size = XS
```

```
>_ python ch_size.py
Chest length? 37.5
Size = M
```

```
>_ python ch_size.py
Chest length? 46
Size = XL
```

Condition Control Statements

ch_size.py

```python
chest = float(input('Chest length? '))
if chest <= 34:
    size = 'XS'
elif chest <= 36:
    size = 'S'
elif chest <= 38:
    size = 'M'
elif chest <= 40:
    size = 'L'
else:
    size = 'XL'
print('Size =', size)
```

```
>_ python ch_size.py
Chest length? 24
Size = XS
```

```
>_ python ch_size.py
Chest length? 37.5
Size = M
```

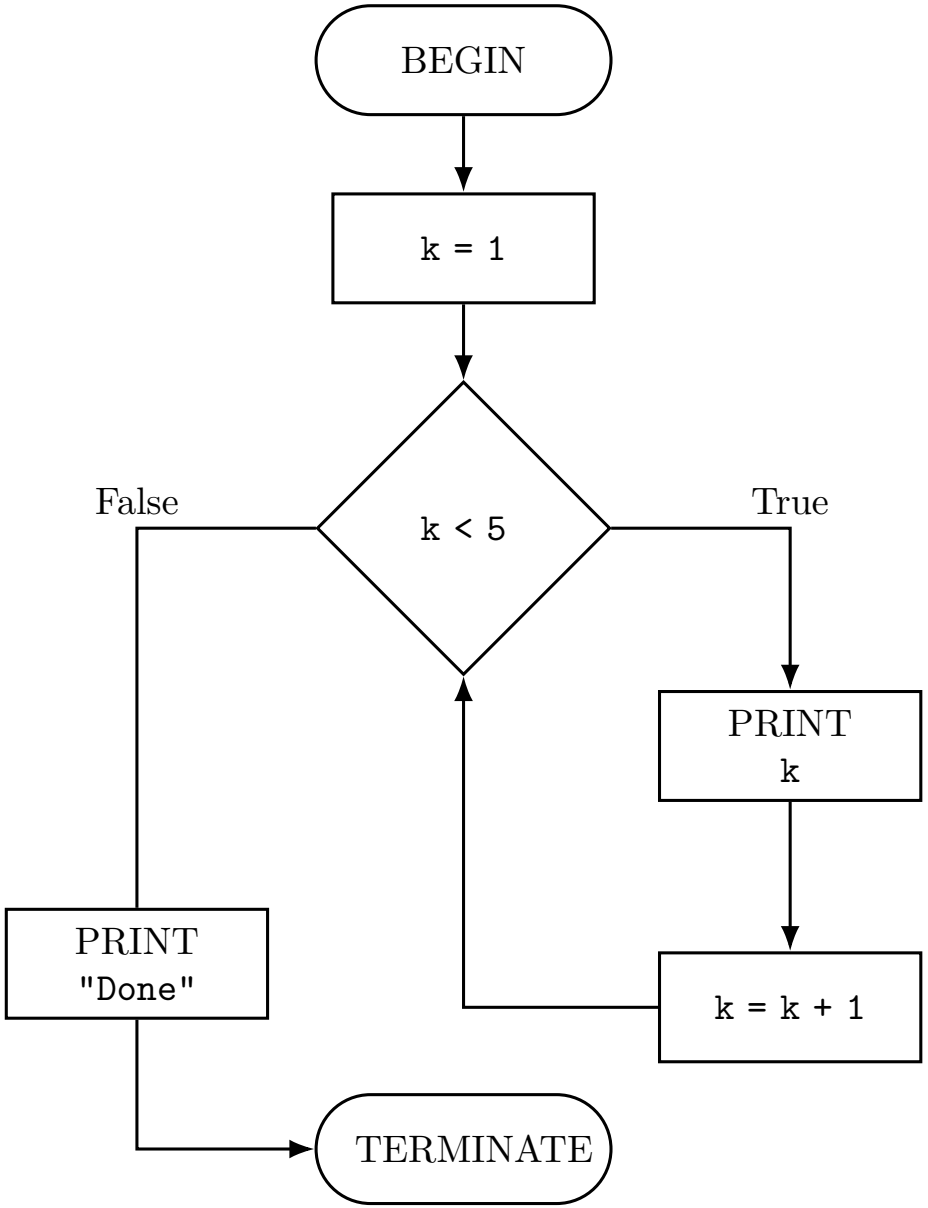
```
>_ python ch_size.py
Chest length? 46
Size = XL
```

Python While Loops

loop_ex1.py

```python
k = 1
while k < 5:
    print(k)
    k = k + 1
print('Done')
```

```
>_ python loop_ex1.py
1
2
3
4
Done
```



Examples

times_table.py

```python
print('Times-table Generator')
n = 1
k = int(input('Enter an integer: '))
while n <= 12:
    x = n * k
    print(k, '*', n, '=', x)
    n = n + 1
print('-' * 25)
```

```
>_ python times_table.py
Times-table Generator
Enter an integer: 9
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
9 * 11 = 99
9 * 12 = 108
-------------------------
```

Example

Write a Python program to find

$$1 + 2 + 3 + ... + 1000.$$

summation_1.py

```python
summation = 0
k = 1
N = 1000
while k <= N:
    summation = summation + k
    k = k + 1
print('Summation is', summation)
```

```
>_ python summation_1.py
Summation is 500500
```

Example

Write a Python program to find

$$1 + 3 + 5 + \ldots + 999.$$

$$\sum_{k=1}^{500} (2k - 1)$$

summation_2.py

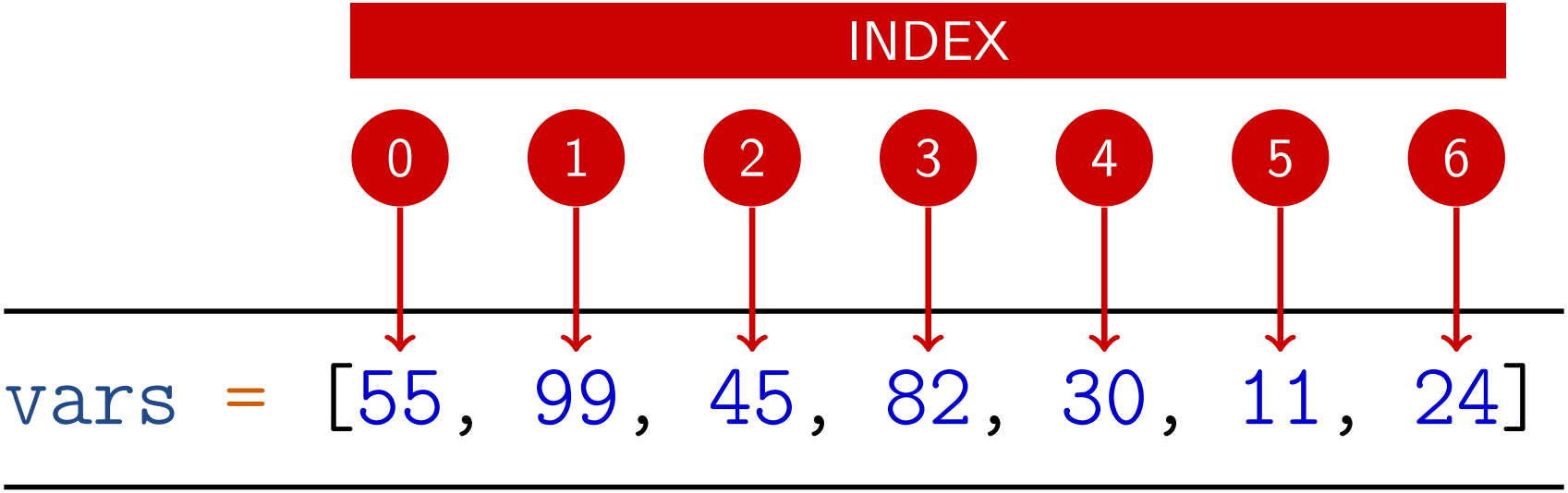
```python
summation = 0
k = 1
N = 500
while k <= N:
    x = 2*k - 1
    summation = summation + x
    k = k + 1
print('Summation is', summation)
```

```
>_ python summation_2.py
Summation is 250000
```

Python List

```
vars = [55, 99, 45, 82, 30, 11, 24]
```

Python List



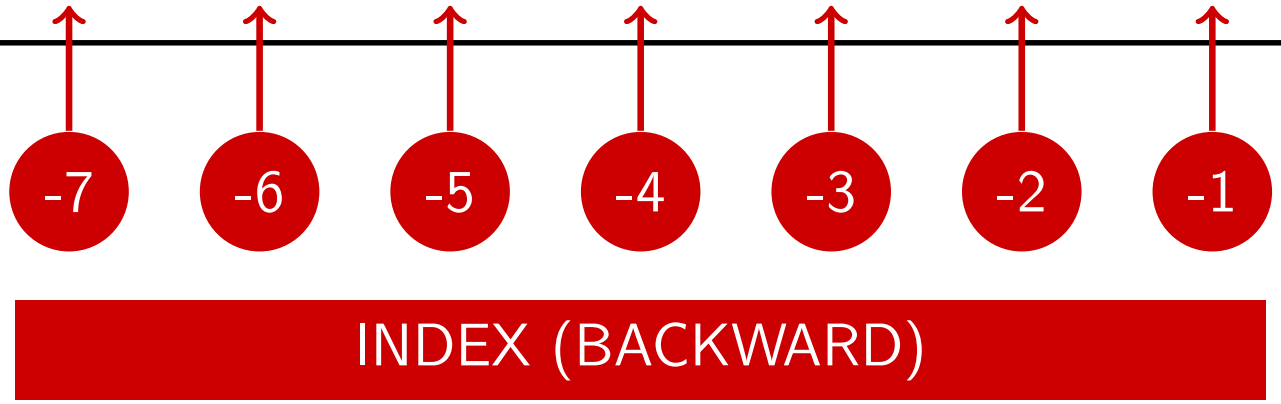

```
>>> vars[0]
55
>>> vars[1]
99
>>> vars[2]
45
```

```
>>> vars[3]
82
>>> vars[4]
30
>>> vars[5]
11
```

```
>>> vars[6]
24
>>> vars[7]
IndexError:
list index
out of range
```

Python List

```
vars = [55, 99, 45, 82, 30, 11, 24]
```

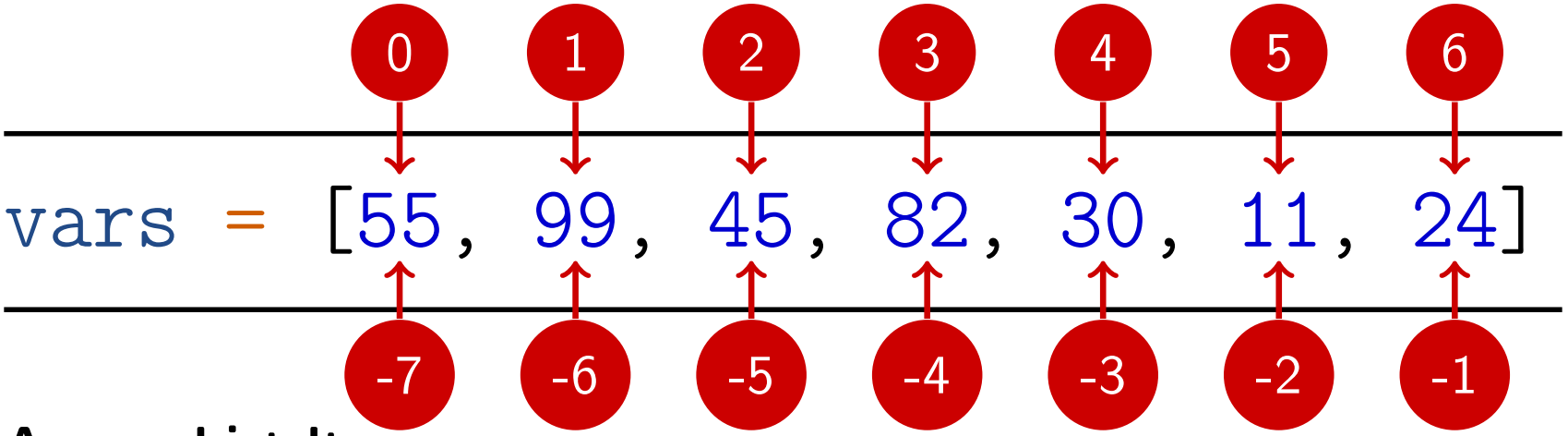



```
>>> vars[-1]
24
>>> vars[-2]
11
>>> vars[-3]
30
```

```
>>> vars[-4]
82
>>> vars[-5]
45
>>> vars[-6]
99
```

```
>>> vars[-7]
55
>>> vars[-8]
IndexError:
list index
out of range
```

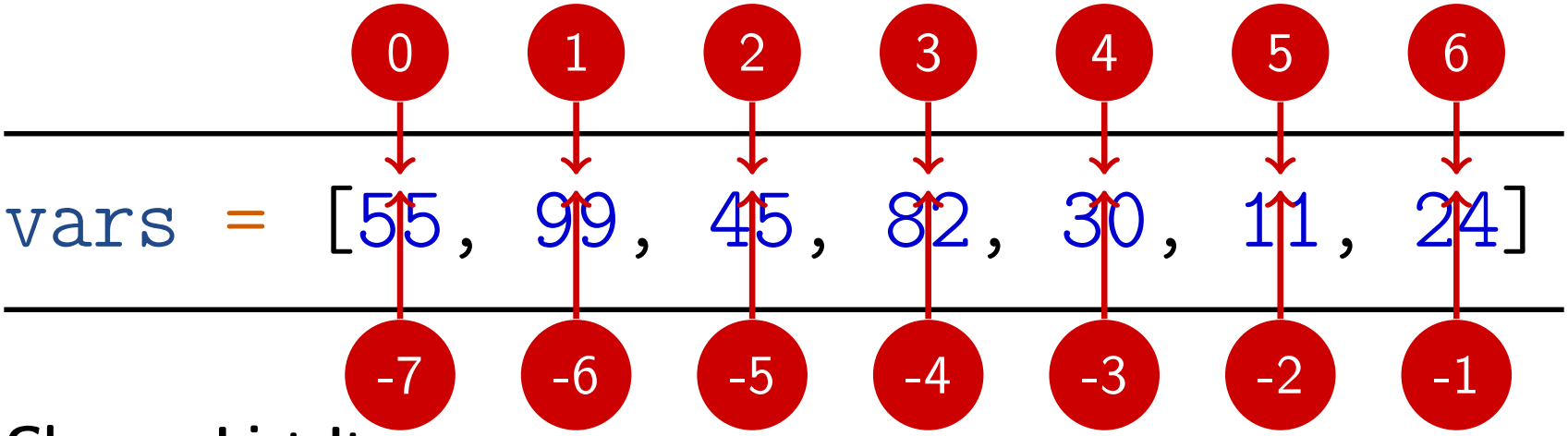
Python List



Access List Items

```
>>> vars[1]
99
>>> vars[0:3]
[55, 99, 45]
>>> vars[2:6]
[45, 82, 30, 11]
```

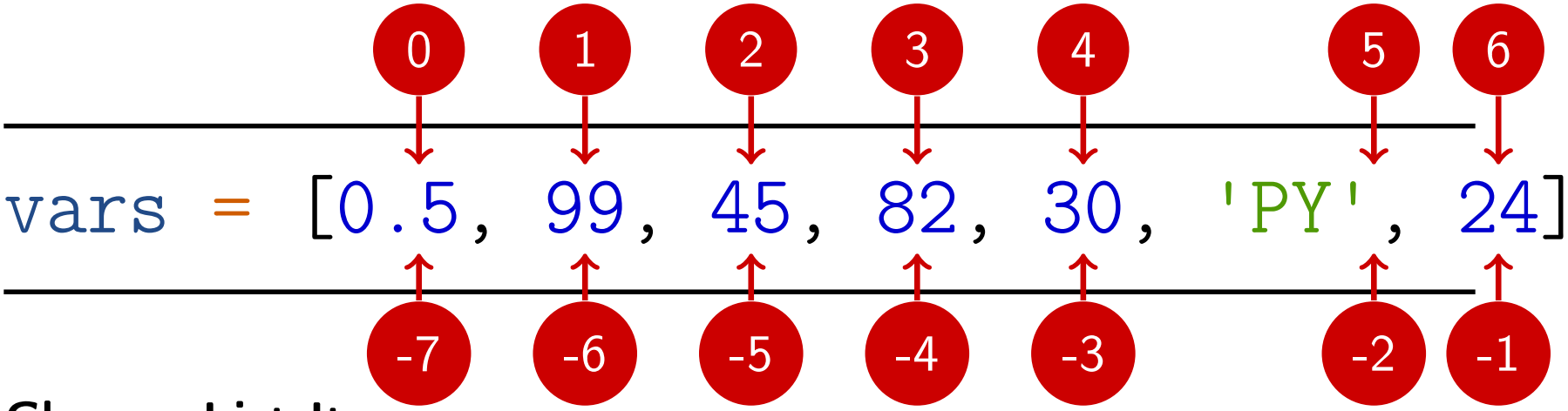
```
>>> vars[-2]
11
>>> vars[-5:-2]
[45, 82, 30]
>>> vars[-7:-1]
[55, 99, 45, 82, 30, 11]
```



Change List Items

```
>>> vars[0] = 0.5
>>> vars[-2] = 'PY'
```

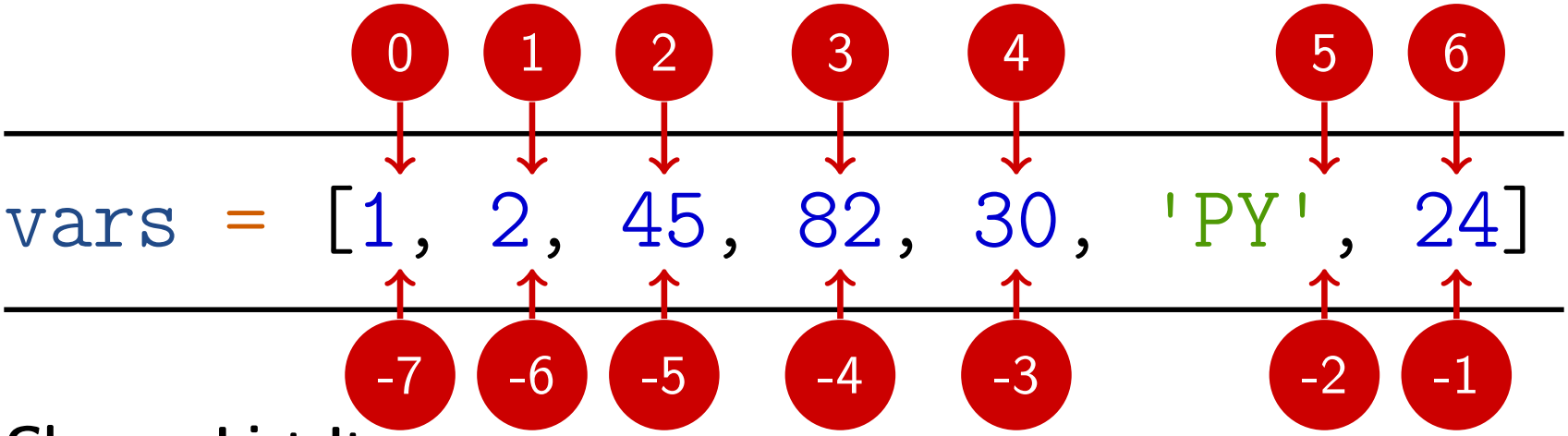
```
>>> vars
[0.5, 99, 45, 82, 30, 'PY', 24]
```



Change List Items

```
>>> vars[0:2] = [1, 2]
```

```
>>> vars
[1, 2, 45, 82, 30, 'PY', 24]
```

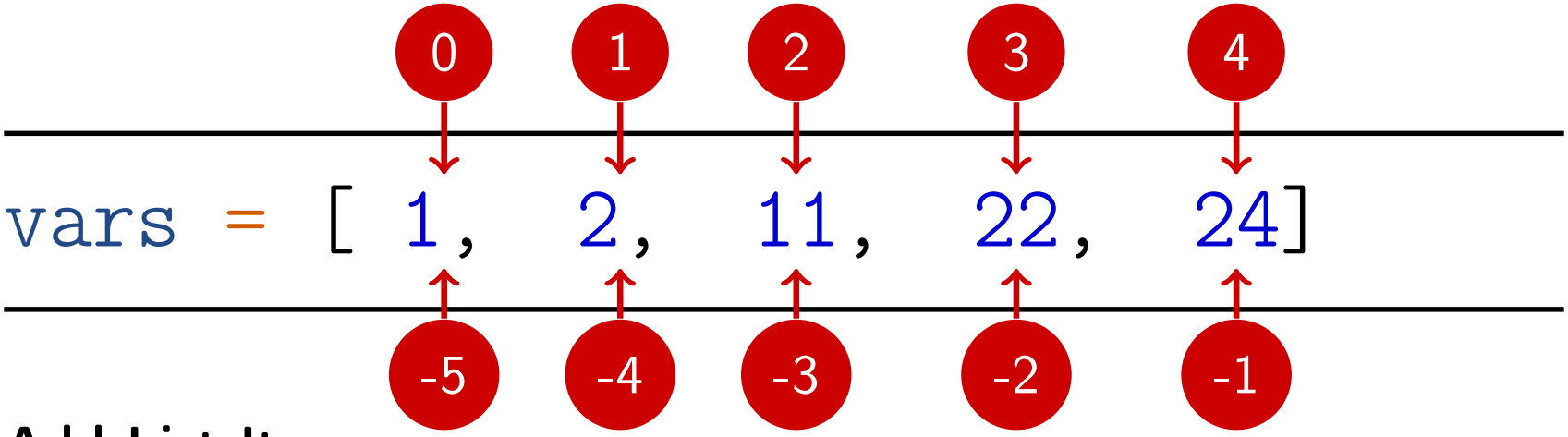


Change List Items

```
>>> vars[2:6] = [11, 22]
```

```
>>> vars
[1, 2, 11, 22, 24]
```

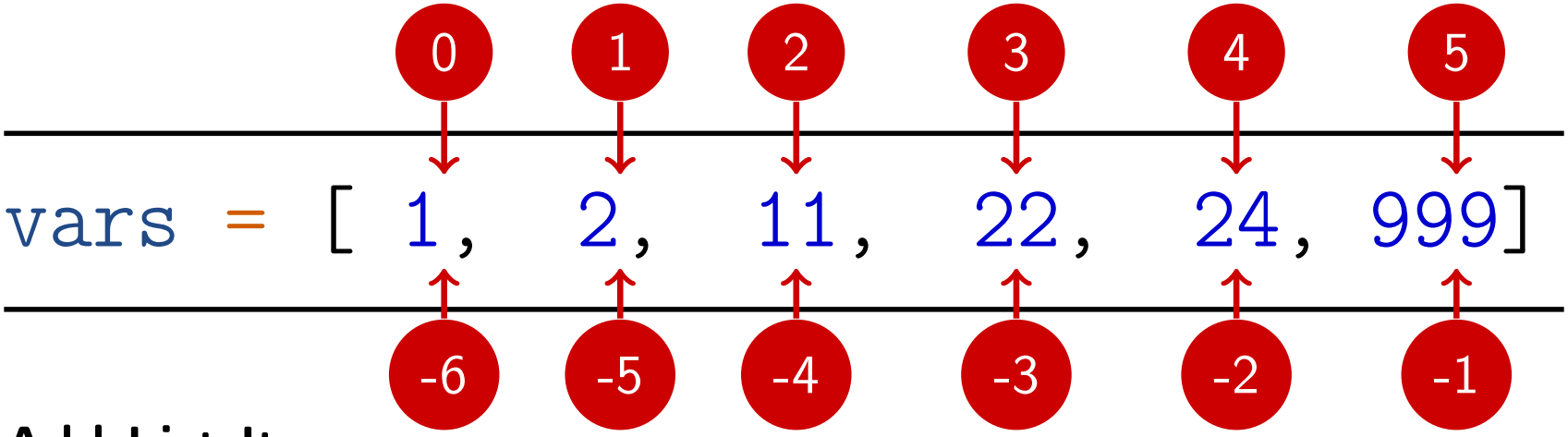
Python List



Add List Items

```
>>> vars.append(999)
```

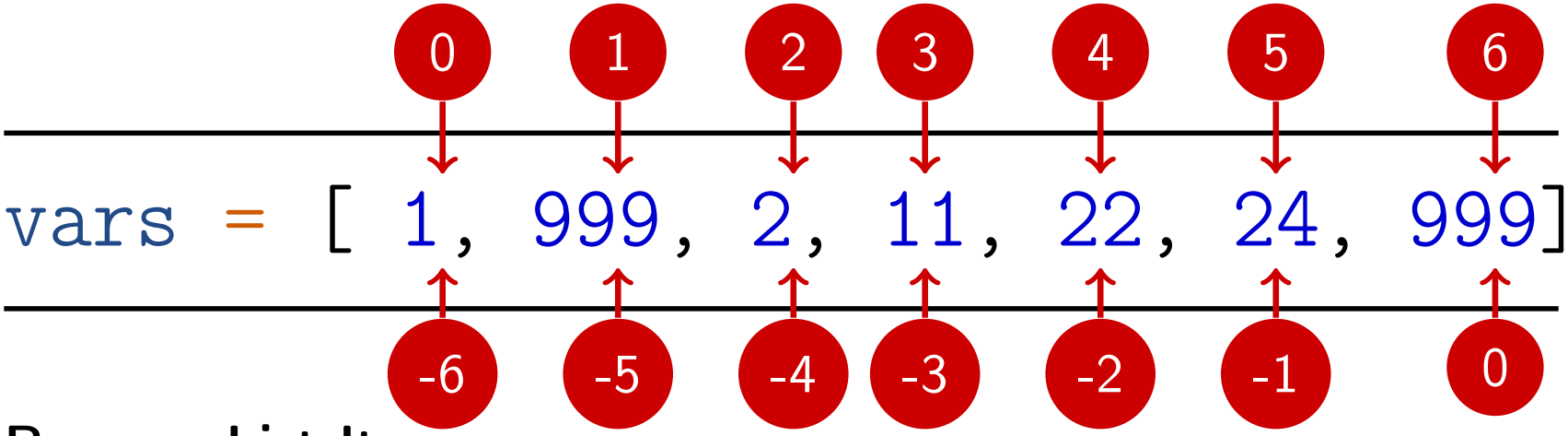
```
>>> vars
[1, 2, 11, 22, 24, 999]
```



Add List Items

```
>>> vars.insert(1, 999)
```

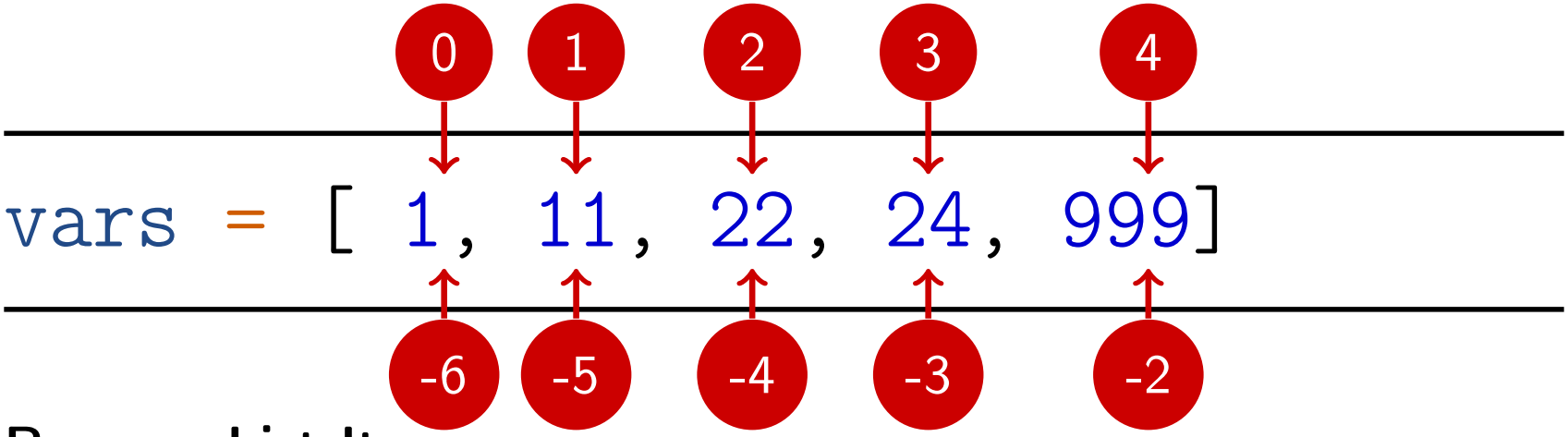
```
>>> vars
[1, 999, 2, 11, 22, 24, 999]
```



Remove List Items

```
>>> vars.remove(2)
```

```
>>> vars
[1, 999, 11, 22, 24, 999]
```

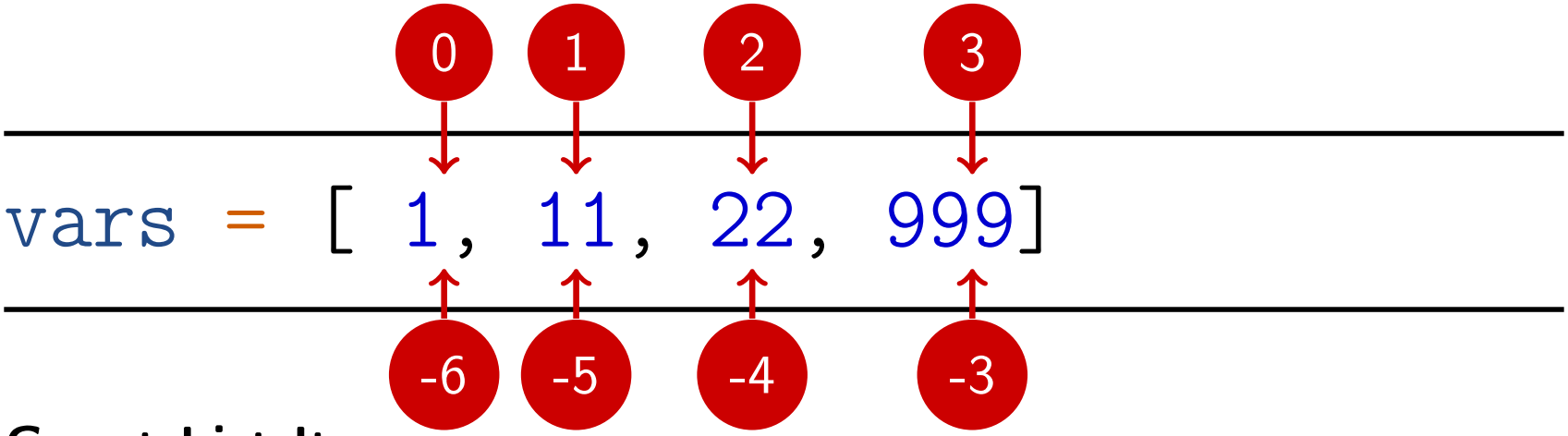


Remove List Items

```
>>> vars.pop(3)
```

```
>>> vars
[1, 11, 22, 999]
```

Python List



Count List Items

```
>>> len(vars)
4
```

Python List

```
xlist = [ 1, 11, 22, 999]
ylist = [888, 168]
```

Join Lists

```
>>> xlist + ylist
[1, 11, 22, 999, 888, 168]
>>> ylist + xlist
[888, 168, 1, 11, 22, 999]
>>> 2 * xlist
[1, 11, 22, 999, 1, 11, 22, 999]
```

Example

Write a Python program to find a summation of items in `xlist`.

summation.py

```
xlist = [3.22, 1.80, 46, 0.33, 4.5, 88, 76.23, 144.21, 36.77, 99.34, 60.32, 4.00, 45.33, 235.0, 453.22]
sumx = 0
num = len(xlist)
n = 0
while n < num:
    sumx = sumx + xlist[n]
    n = n + 1
print('Summation is', sumx)
```

```
>_ python summation.py
Summation is 1298.27
```

Python For Loop

summation.py

```python
xlist = [3.22, 1.80, 46, 0.33, 4.5, 88, 76.23, 144.21, 36.77, 99.34, 60.32, 4.00, 45.33, 235.0, 453.22]
sumx = 0
num = len(xlist)

for n in range(num):
    sumx = sumx + xlist[n]

print('Summation is', sumx)
```

```
>_ python summation.py
Summation is 1298.27
```

num = 14

n = 0, 1, 2, ..., 13

Different Types of For-loop

```
xlist = [100, 200, 300, 400, 500, 600, 700, 800, 900]
num  = len(xlist)

for n in range(num):
    print(xlist[n])

for m in range(3, 6):
    print(xlist[m])

for k in range(1, 8, 3):
    print(xlist[k])

for x in xlist:
    print(x)
```

num = 9

n = 0, 1, 2, ..., 8

m = 3, 4, 5

k = 1, 4, 7

x = 100, 200, ..., 900

Example

summation_f_1.py

```python
xlist = [3.22, 1.80, 46, 0.33, 4.5, 88, 76.23, 144.21, 36.77, 99.34, 60.32, 4.00, 45.33, 235.0, 453.22]
sumx = 0
num = len(xlist)

for n in range(num):
    sumx = sumx + xlist[n]

print('Summation is', sumx)
```

summation_f_2.py

```python
xlist = [3.22, 1.80, 46, 0.33, 4.5, 88, 76.23, 144.21, 36.77, 99.34, 60.32, 4.00, 45.33, 235.0, 453.22]
sumx = 0


for x in xlist:
    sumx = sumx + x

print('Summation is', sumx)
```

Python Built-In Functions

Here are some functions you have already seen.

Function	Description
`print()`	Prints to the standard output device
`input()`	Allowing a user input and returning a string from it
`int()`	Return an integer number
`float()`	Return a floating-point number
`str()`	Return a string
`len()`	Returns the length of an object

User-Defined Functions

Defining a function

Name of the function

```
def function_name():
    statement_1
    statement_2
    statement_3
    ...
    statement_n
```

A block of code

Example

line_fn.py

```python
def line():
    x = '-' * 30
    print(x)


x = 10
print('x = ', x)
line()
x = x + 10
print('x = ', x)
line()
```

```
>_ python line_fn.py
x =  10
------------------------------
x =  20
------------------------------
```

Example

hi_offer.py

```python
def say_hi():
    print('Hello')
    print('Nice to meet you!')

def offer_meal():
    print('Please have some meal')
    print('It is very good pasta')


guest = 'Jame Doe'
print('Here is', guest)
say_hi()
offer_meal()
```

```
>_ python hi_offer.py
Here is Jame Doe
Hello
Nice to meet you!
Please have some meal
It is very good pasta
```

Function Inputs and Outputs

Function inputs (Arguments)

```
def function_name(args):
    statement_1
    statement_2
    statement_3
    ...
    statement_n
    return outputs
```

Function outputs

Example

line_fn.py

```python
def line(symbol):
    x = symbol * 25
    print(x)


x = 10
print('x = ', x)
line('-')
x = x + 10
print('x = ', x)
line('*')
```

```
>_ python line_fn.py
x =  10
------------------------
x =  20
************************
```

Example

line_fn.py

```python
def line(symbol, n):
    x = symbol * n
    return x


x = 10
print('x = ', x)
print(line('#', 15))
x = x + 10
print('x = ', x)
print(line('@', 25))
```

```
>_ python line_fn.py
x =  10
###############
x =  20
@@@@@@@@@@@@@@@@@@@@@@@@@
```

Example

circle_fn.py

```python
def circle_area(radius):
    pi = 3.14159265359
    area = pi * (radius ** 2)
    return area

def line(symbol, n):
    x = symbol * n
    return x


radius_list = [3.44, 1.56, 6.88]
for r in radius_list:
    print('Radius is', r)
    a = circle_area(r)
    print('Area is', a)
    print(line('-', 25))
```

```
>_ python circle_fn.py
Radius is 3.44
Area is 37.17635082552262
-------------------------
Radius is 1.56
Area is 7.645379881776625
-------------------------
Radius is 6.88
Area is 148.70540330209047
-------------------------
```