Programming Fundamental

A Hitchhiker Guide to Coding with Python

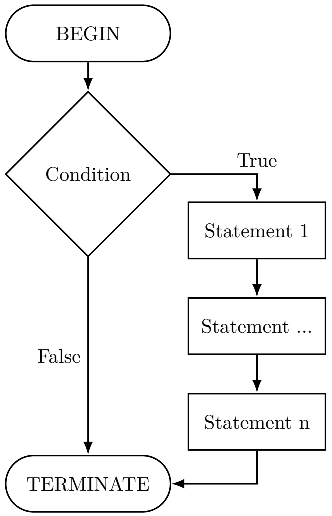
Lesson 4: Power of Iterations

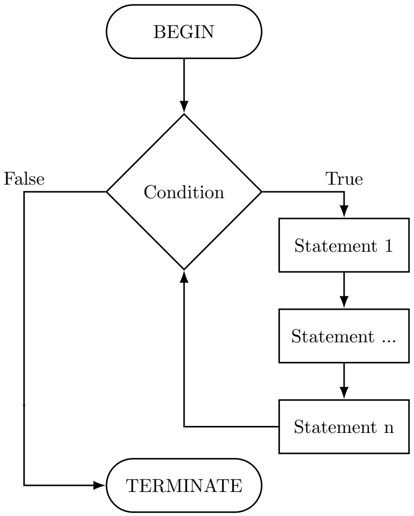
Akarate Singta

Department of Mathematics
Faculty of Science and Technology, RMUTT

Lesson Outline

Conditions vs Loops



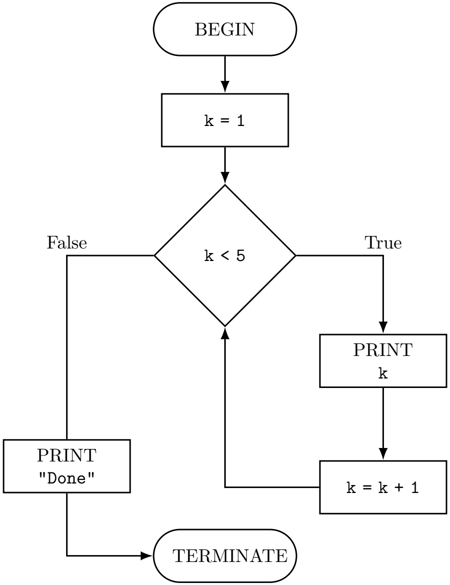


Python While Loops

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
```



Python While Loops

loop_ex2.py

```python
k = 1
while k > 0:
    print(k)
    k = k + 1
print('Done')
```

```
>_ python loop_ex2.py
1
2
3
4
.
.
.
```

Always True!

This is an **infinite loop**

Only Exit with
ctrl + C
or
ctrl + D

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
-------------------------
```

Examples

ftimes_table.py

```python
print('Times-table Generator')
run = 1
while run > 0:
    n = 1
    k = int(input('Enter an integer: '))
    while n <= 12:
        x = n * k
        print(k, '*', n, '=', x)
        n = n + 1
    print('-' * 25)
print('Program is terminated.')
```

```
>_ python ftimes_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
-------------------------
Enter an integer:
```

Examples

rtimes_table.py

```python
print('Times-table Generator')
run = 1
while run > 0:
    n = 1
    k = int(input('Enter a number: '))
    while n <= 12:
        x = n * k
        print(k,'*',n,'=',x)
        n = n + 1
    print('-' * 25)
    key = input('Continue? ')
    if key == 'exit':
        run = -1
print('Program is terminated.')
```

```
>_ python rtimes_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
-------------------------
Continue?
```

Examples

circle_area.py

```python
pi = 3.14159265359
run = 1
while run > 0:
    r = float(input('Enter a radius: '))
    area = pi * r ** 2
    print('Area is', area)
    print('-' * 25)
    key = input('Continue? ')
    if key == 'exit' or key == 'Exit':
        run = -999
print('Program is terminated')
```

```
>_ python circle_area.py
Enter a radius: 2
Area is 12.56637061436
-------------------------
Continue?
Enter a radius: 44
Area is 6082.12337735024
-------------------------
Continue?
Enter a radius: 5.5
Area is 95.0331777710975
-------------------------
Continue? exit
Program is terminated
```

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

$$1 + 3 + 5 + ... + 999.$$

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

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
```

Exercise

Write a Python program to find

$$\frac{1}{2\cdot 3}+\frac{2}{3\cdot 4}+\frac{3}{4\cdot 5}+\ldots+\frac{998}{999\cdot 1000}.$$

$$\sum_{n=1}^{998}\frac{n}{(n+1)(n+2)}$$

Exercise

Write a Python program to find

$$\left(\frac{1}{2\cdot 3}\right)\times\left(\frac{2}{3\cdot 4}\right)\times\left(\frac{3}{4\cdot 5}\right)\times\ldots\times\left(\frac{998}{999\cdot 1000}\right).$$

$$\boxed{\prod_{n=1}^{998}\frac{n}{(n+1)(n+2)}}$$

Exercise

Write a Python program to find

$$n! = (n)(n-1)(n-2)\cdots(3)(2)(1)$$

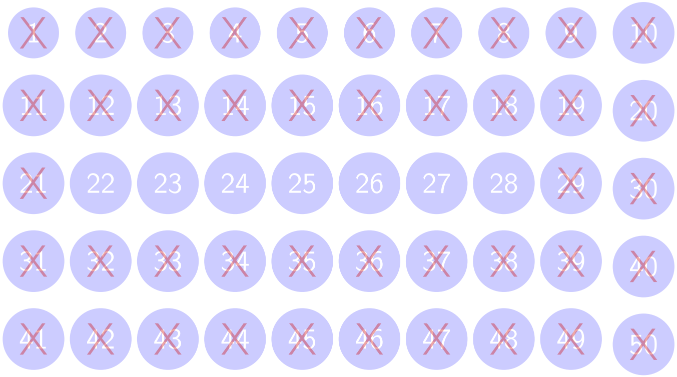
for any positive integer n.

Exercise

Guessing Game

> *The computer has a secret integer from 1 to 50. A player keeps guessing numbers until he find the computer's number, and the computer will tell the player each time if the guess was too high or too low.*

Write a Python program for this game.



Challenge!

Guessing Game+

This is as same as the previous guessing game, but it comes with options allowing player to choose a difficulty. Each difficulty affects the secret integer lying in different ranges as follows;

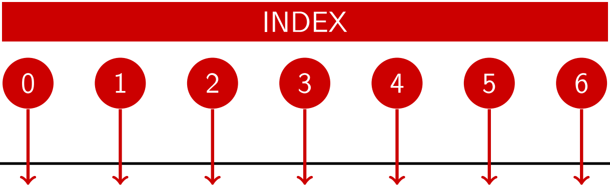
Difficulty	Secret Integer Range
Easy	[1, 10]
Normal	[1, 50]
Hard	[-100, 100]
Custom	[?, ?]

Write a Python program for this game.

Python List

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

Python List



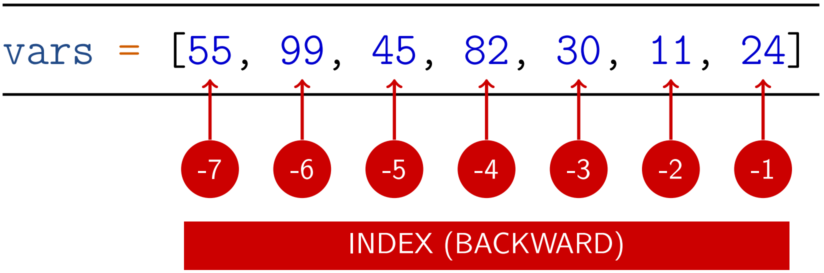
```
vars = [55, 99, 45, 82, 30, 11, 24]
```

```
>>> 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



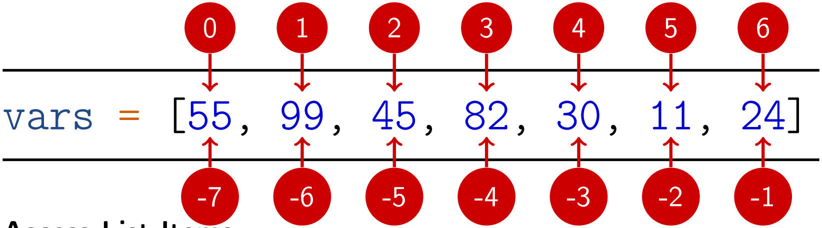
```
vars = [55, 99, 45, 82, 30, 11, 24]
```

```
>>> 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



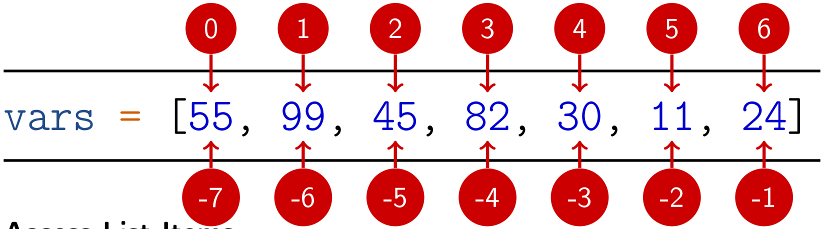
```
vars = [55, 99, 45, 82, 30, 11, 24]
```

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]
```

Python List

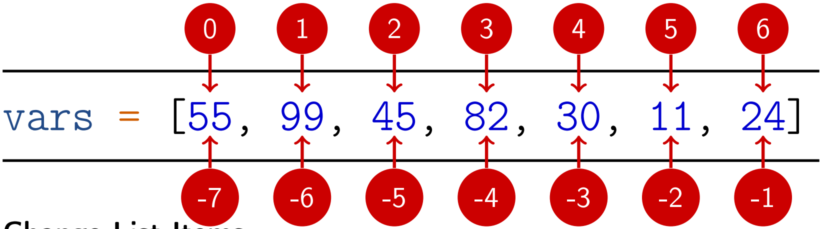


Access List Items

```
>>> vars[1:]
[99, 45, 82, 30, 11, 24]
>>> vars[-6:]
[99, 45, 82, 30, 11, 24]
```

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

Python List



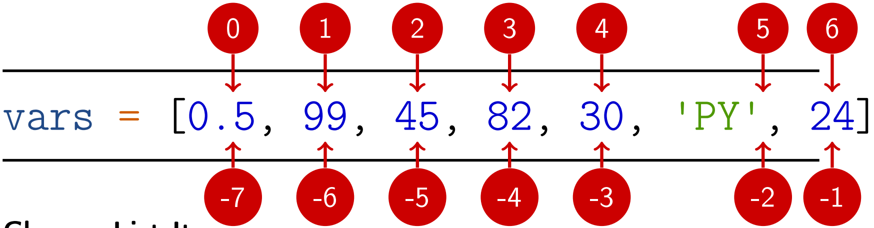
```
vars = [55, 99, 45, 82, 30, 11, 24]
```

Change List Items

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

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

Python List

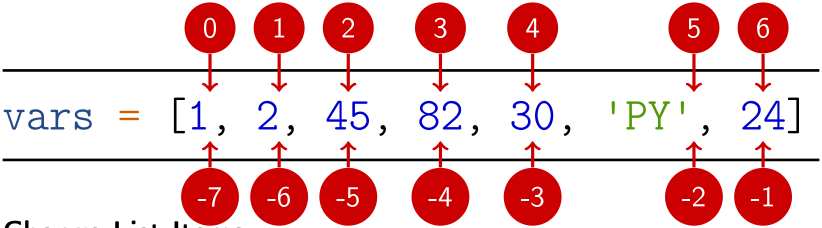


Change List Items

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

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

Python List



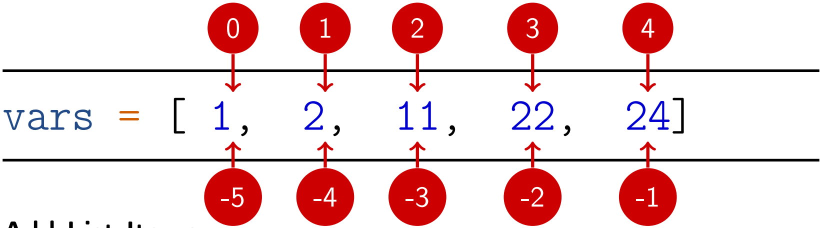
```
vars = [1, 2, 45, 82, 30, 'PY', 24]
```

Change List Items

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

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

Python List

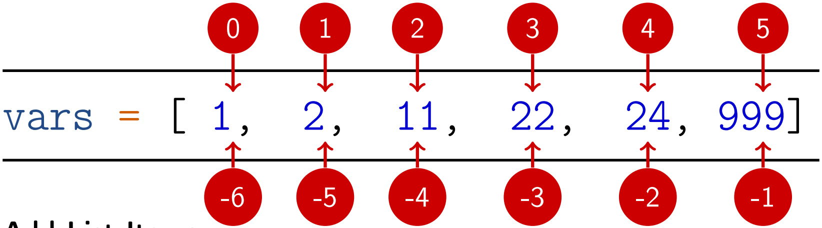


Add List Items

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

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

Python List



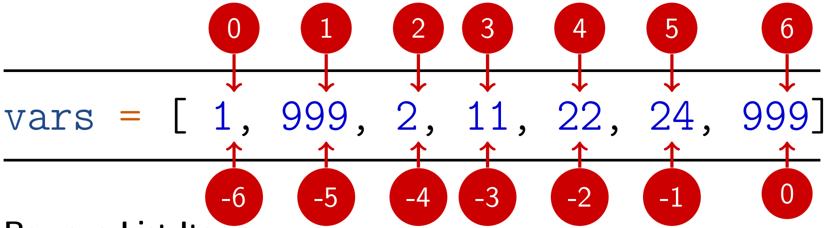
```
vars = [ 1,  2,  11,  22,  24, 999]
```

Add List Items

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

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

Python List



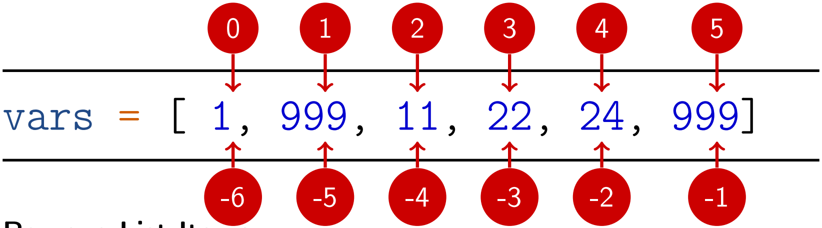
```
vars = [ 1, 999, 2, 11, 22, 24, 999]
```

Remove List Items

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

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

Python List

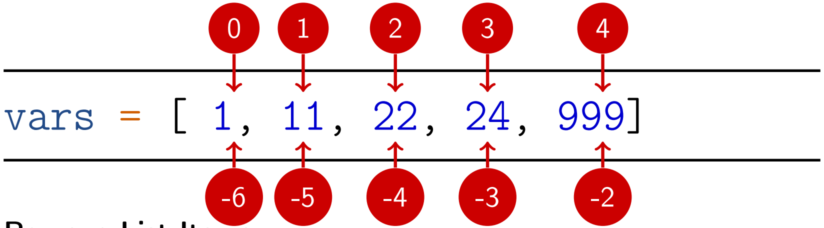


Remove List Items

```
>>> vars.remove(999)
```

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

Python List

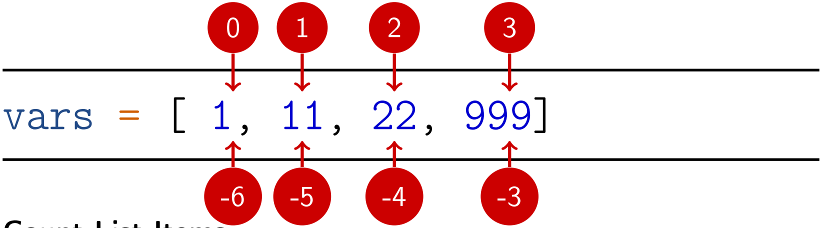


Remove List Items

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

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

Python List



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

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

```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)
n = 0
while n < num:
    sumx = sumx + xlist[n]
    n = n + 1
print('Summation is', sumx)
```

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

Example

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)
n = 0
while n < num:
    sumx = sumx + xlist[n]
    print('n =', n, '--> sumx =', sumx)
    n = n + 1
print('Summation is', sumx)
```

```
>_ python summation.py
n = 0 --> sumx = 3.22
n = 1 --> sumx = 5.02
n = 2 --> sumx = 51.02
n = 3 --> sumx = 51.35
n = 4 --> sumx = 55.85
n = 5 --> sumx = 143.85
n = 6 --> sumx = 220.08
n = 7 --> sumx = 364.29
n = 8 --> sumx = 401.06
n = 9 --> sumx = 500.40
n = 10 --> sumx = 560.72
n = 11 --> sumx = 564.72
n = 12 --> sumx = 610.05
n = 13 --> sumx = 845.05
n = 14 --> sumx = 1298.27
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

While-loop vs For-loop

while_loop.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)

n = 0
while n < num:
    sumx = sumx + xlist[n]
    n = n + 1

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

for_loop.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)
```

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)
```

Example

Write a Python program to find a summation of numbers in data.

summation_for.py

```python
data = ['tha', 'THB', 60,
        'eng', 'GHB', 55,
        'deu', 'EUR', 75,
        'jap', 'YEN', 46,
        'esp', 'EUR', 78]
sumn = 0
num = len(data)

for n in range(2, num, 3):
    sumn = sumn + data[n]

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

summation_while.py

```python
data = ['tha', 'THB', 60,
        'eng', 'GHB', 55,
        'deu', 'EUR', 75,
        'jap', 'YEN', 46,
        'esp', 'EUR', 78]
sumn = 0
num = len(data)
n = 2
while n < num:
    sumn = sumn + data[n]
    n = n + 3
print('Summation is', sumn)
```

Example

Write a Python program to collect numbers from 10 inputs.

collect.py

```python
data = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
for n in range(10):
    x = float(input('Enter a number: '))
    data[n] = x
print(data)
```

```
>_ python collect.py
Enter a number: 34
Enter a number: 33
Enter a number: 23
Enter a number: 44
Enter a number: 66
Enter a number: 34
Enter a number: 23
Enter a number: 12
Enter a number: 55
Enter a number: 56
[34.0, 33.0, 23.0,
    44.0, 66.0, 34.0,
    23.0, 12.0, 55.0,
    56.0]
```

Example

Write a Python program to collect numbers from 10 inputs.

collect.py

```python
data = [ ]
for n in range(10):
    x = float(input('Enter a number: '))
    data.append(x)
print(data)
```

```
>_ python collect.py
Enter a number: 34
Enter a number: 33
Enter a number: 23
Enter a number: 44
Enter a number: 66
Enter a number: 34
Enter a number: 23
Enter a number: 12
Enter a number: 55
Enter a number: 56
[34.0, 33.0, 23.0,
    44.0, 66.0, 34.0,
    23.0, 12.0, 55.0,
    56.0]
```

Example

Write a Python program to collect exam scores from a number of students. Inputs must be positive numbers or zero. The program stops collecting with a negative input.

collect.py

```python
data = [ ]
run = True
while run:
    x = float(input('Enter a number: '))
    if x >= 0:
        data.append(x)
    else:
        run = False
print(data)
```

```
>_ python collect.py
Enter a number: 34
Enter a number: 33
Enter a number: 23
Enter a number: 44
.
.
.
Enter a number: 82
Enter a number: -9
[34.0, 33.0, 23.0,
    44.0, ...,
    82.0]
```