Computational Basics

Repeating Yourself with Loops
Question

\[ x = (3.5 + 1j) \]
\[ y = 1 \]
\[ z = x + y \]

What is the type of \( z.\text{imag} \)?
x = (3.5 + 1j)
y = 1
z = x + y

What is the type of z.imag?
A int
B float
C complex
x = (3.5 + 1j)
y = 1
z = x + y

What is the type of z.imag?

A int
B float ★z is complex, not its components!
C complex
\( x = (3.5 + 1j) \)
\( y = 1 \)
\( z = x + y \)

What is the value of \( z.\text{imag} \)?

A 4.5 + 1j
B 4.5
C 1j
D 1.0
x = (3.5 + 1j)
y = 1
z = x + y

What is the value of z.imag?
A 4.5 + 1j
B 4.5
C 1j
D 1.0 ⭐
String Data Type
String operations

- **Concatenation**: combine two strings
  - Uses the + symbol
  - ’RACE’ + ’CAR’

- **Repetition**: repeat a string
  - Uses the *
  - ’HELLO ’*10

- **Formatting**: used to encode other data as string
  - Uses % symbol
Formatting operator

- Creates string with value inserted
  - Formats nicely
  - Requires indicator of type inside of string

```python
x = 100 * 54
s = "String is: %i" % x
print(s)
```
Formatting operator

- Creates string with value inserted
  - Formats nicely
  - Requires indicator of type inside of string

```python
x = 100 * 54
s = "String is: %i" % x
print(s)
```
name = "Neal"
grade = 2 / 3
m1 = "Hello, %s!" % name  # %s = string
m2 = "Your grade is:  %f." % grade  # %f = float
print(m1)
print(m2)
name = "Neal"
grade = 2 / 3
m1 = "Hello, %s!" % name  # %s = string
m2 = "Your grade is: %f." % grade  # %f = float
print(m1)
print(m2)

Hello, Neal!
Your grade is 0.66667.
x = 3
s = ("%i" % (x+1)) * x**(5%x)
print(s)

What does this program print?
A 333333333333
B 4444444444
C 9999
D %i%i%i%i%i
x = 3
s = ("%i" % (x+1)) * x**(5%x)
print(s)

What does this program print?
A 333333333333
B 4444444444 ★(Trace the steps!)
C 9999
D %i%i%i%i%i
Example

```python
x = 3
s = ("%i" % (x+1)) * x**(5%x)
```

```python
s = ("%i" % (3+1)) * 3**(5%3)
```

```python
s = ("%i" % (4)) * 3**2
```

```python
s = ("%i" % (4)) * 3**2
```

```python
s = "4" * 9
```

```python
s = "444444444"
```
Example

\[
x = 3 \\
\text{s} = \left(\%i \ % (x+1)\right) \ * \ x**(5\%x) \\
\text{s} = \left(\%i \ % (3+1)\right) \ * \ 3**(5\%3)
\]
Example

```python
x = 3
s = ('%i' % (x+1)) * x**(5%x)
s = ('%i' % (3+1)) * 3**(5%3)
s = ('%i' % (4)) * 3**2
```
Example

\[
\begin{align*}
\text{x} &= 3 \\
\text{s} &= (\text{"%i" } \% (\text{x}+1)) \times \text{x}^{\text{x} \% \text{x}} \\
\text{s} &= (\text{"%i" } \% (3+1)) \times 3^{5 \% 3} \\
\text{s} &= (\text{"%i" } \% (4)) \times 3^{2} \\
\text{s} &= (\text{"4"}) \times 9
\end{align*}
\]
x = 3
s = ("%i" % (x+1)) * x**(5%x)
s = ("%i" % (3+1)) * 3**((5%3))
s = ("%i" % (4)) * 3**2
s = ("4") * 9
s = "4" * 9
x = 3
s = ("%i" % (x+1)) * x**(5%x)
s = ("%i" % (3+1)) * 3**(5%3)
s = ("%i" % (4)) * 3**2
s = ("4") * 9
s = "4" * 9
s = "4444444444"
s = "ABCDE"
i = 3
x = s[i]

What is the value of x?
A 'A'
B 'B'
C 'C'
D 'D'
E 'E'
s = "ABCDE"
i = 3
x = s[i]

What is the value of x?
A  'A'
B  'B'
C  'C'
D  'D' ⋆
E  'E'
What is the value of $y$?

A  'A'
B  'B'
C  'C'
D  'D'
E  'E'
s = "ABCDE"
i = (11 % 3) - 7
z = s[i]

What is the value of z?
A 'A' ✤
B 'B'
C 'C'
D 'D'
E 'E'
s = "WATER MAIN"[2:6]
t = int(3.7)
x = s[-1] + s[t-2]

What is the value of x?
A "NA"
B "E"
C "R"
D "ME"
s = "WATER MAIN"[2:6]
t = int(3.7)
x = s[-1] + s[t-2]

What is the value of x?
A "NA"
B " E" ★
C " R"
D "ME"
s = "WATER MAIN"[2:6]
#0123456789
s = "TER "
t = int(3.7)
t = 3
x = s[-1] + s[t-2]
x = " " + "E"
x = " E"
User Input
input is a built-in function.
- Argument: string prompting user
- Return value: input from user (as str!)
Example: if statement

```python
ans = input( "Enter a number:" )
if ans < 0:
    print( "The number was negative." )
```
ans = input( "Enter a number:" )
ans = float( ans )
if ans < 0:
    print( "The number was negative." )
A program should achieve a goal.
A program should achieve a goal.
Let’s implement the quadratic equation.
Example: Quadratic equation

```python
print( "QUADRATIC SOLVER" )
print( "a x**2 + b x + c = 0" )

a = float( input( 'a = ' ) )
b = float( input( 'b = ' ) )
c = float( input( 'c = ' ) )

root = ( b**2 - 4*a*c ) ** 0.5
denom = 2 * a
```
Example: Quadratic equation

```python
pos = (-b + root) / denom
neg = (-b - root) / denom

message1 = "\%f + \%fi" % (pos.real,pos.imag)
message2 = "\%f + \%fi" % (neg.real,neg.imag)

print("Solution 1: %s" % message1)
print("Solution 2: %s" % message2)
```
Loops
ans = input( "Enter a number:" )
ans = float( ans )
if ans < 0:
    print( "The number was negative." )
if ans > 0:
    print( "The number was positive." )
if ans == 0:
    print( "The number was zero." )

END
Flowcharts

Loops
number = 10
while number > 0:
    print(number)
    number = number - 1
print('Blast off!')
A `while` loop has only:
- the keyword `while`
- a logical comparison (bool-valued result)
- a `block` of code
ans = 'Repete'
while ans == 'Repete':
    ans = input( 'Pete and Repete were in a\n    boat. Pete fell out.\n    Who was left? ' )
The following code should increment $x$ if the hundreds place contains a zero:

```python
x = 3
while x > 0:
    print("Hello")
    x -= 1
```

How many times is 'Hello' printed?

A zero
B once
C twice
D thrice
E four times
The following code should increment $x$ if the hundreds place contains a zero:

```python
x = 3
while x > 0:
    print("Hello")
    x -= 1
```

How many times is 'Hello' printed?

Unroll the loop in a spreadsheet!
The following code should increment \( x \) if the hundreds place contains a zero:

```python
x = 3
while x > 0:
    print("Hello")
    x -= 1
```

How many times is 'Hello' printed?

A. zero
B. once
C. twice
D. thrice ⭐
E. four times
Make sure that your code always has a way to end!
```python
while True:
    print('Hello!')
```
Infinite loops

- Make sure that your code always has a way to end!
  ```python
  while True:
      print('Hello!')
  ```
- Use Ctrl+C to break free.
Design patterns are common structures we encounter in writing code.

The accumulator pattern uses an accumulator variable to track a result inside of a loop:

```
i = 0
sum = 0
while i <= 4:
    sum += i
    i += 1
```
i = 0
sum = 0
while i <= 4:
    i = i + 1
    sum = sum + 1
Example

i = 0
sum = 0
while i <= 4:
    sum += i
    i += 1

What is the value of sum?
A 6
B 10
C 15
D None of the above.
Loops

```python
i = 0
sum = 0
while i <= 4:
    sum += i
    i += 1
```

What is the value of `sum`?

A 6
B 10 \(\star\) 1+2+3+4
C 15
D None of the above.
Example

```python
i = 0
sum = 0
while i < 7:
    if (i % 2) == 1:
        sum += i
    i += 1
```

What is the value of `sum`?

A 9  
B 12  
C 16  
D 21  
E None of the above.
Example

```python
i = 0
sum = 0
while i < 7:
    if (i % 2) == 1:
        sum += i
        i += 1

What is the value of `sum`?
A  9  
B  12 
C  16 
D  21 
E  None of the above. ⋆(An infinite loop occurs.)
```
Exercise

Write a program to sum all of the digits in a number. *i.e.,*

\[
12145 \rightarrow 1 + 2 + 1 + 4 + 5 \rightarrow 13
\]
s = str(n)
i = 0
result = 0
while i < len(s):
    result = result + int(s[i])
i = i + 1
s = str( n )
result = 0
for x in s:
    result = result + int( x )
for i in range(0, 100):
    print(i)
Defining loops: `for`

- A `for` loop requires:
  - the keyword `for`
  - a loop variable
  - the keyword `in`
  - a set of values (often `range`)
  - a **block** of code
- `for` loops iterate over *iterable* types one at a time.
s = 'abcdefg'
t = '
for c in s:
    t = c + t

s = 'abcdefg'
t = ''
t = c + t

any letters left?

T

F

END

Loops
s = 'abcdefg'
t = ''
for c in s:
    t = c + t

What is the value of t?
A 'abcdefg'
B 'gfedcba'
C 'a'
D 'g'
s = 'abcdefg'
t = ''
for c in s:
    t = c + t

Flowchart:

1. s = 'abcdefg'
2. t = ''
3. For each c in s:
   3.1 c = 'a'
   3.2 t = 'a' + ''
   3.3 c = 'b'
   3.4 t = 'b' + 'a'
   3.5 c = 'c'
   3.6 t = 'c' + 'ba'
   3.7 c = 'd'
   3.8 t = 'd' + 'cba'
   3.9 c = 'e'
   3.10 t = 'e' + 'dcba'
   3.11 c = 'f'
   3.12 t = 'f' + 'edcba'
   3.13 c = 'g'
   3.14 t = 'g' + 'fedcba'
4. t = c + t
5. Any letters left?
   5.1 If yes, go back to 3.
   5.2 If no, End.
s = 'abcdefg'
t = ''
for c in s:
    t = c + t

What is the value of t?
A 'abcdefg'
B 'gfedcba' ✶
C 'a'
D 'g'
Write a program which counts the number of vowels in a string. Test it on ‘carolingian’ (6).
Write a program which counts the number of vowels in a string. Test it on 'carolingian' (6).

```python
my_string = 'carolingian'
vowel_count = 0
for char in my_string:
    if char in 'aeiou':
        vowel_count = vowel_count + 1
```
The `range` function returns a sequential set of integers.

Two arguments:
- (optional) the starting value of the range (inclusive)
- the ending value of the range (exclusive)
- (optional) the step size
for i in range(10, 0, -1):
    print(i ** 2)
Write a loop to reverse a given string, ’Newton’.

```python
newstring = ''
oldstring = 'Newton'
for char in oldstring:
    newstring = char + newstring
print(newstring)
```
Write a loop to reverse a given string, ’Newton’.

newstring = ''
oldstring = 'Newton'
for char in oldstring:
    newstring = char + newstring
print( newstring )
Calculate all factors of a number.

```python
from math import sqrt
for x in range(1, int(sqrt(n)) + 1):
    if n % x == 0:
        print(x)
print(n // x)
```
Calculate all factors of a number.

```python
from math import sqrt

for x in range(1, int(sqrt(n)) + 1):
    if n % x == 0:
        print(x)
        print(n // x)
```