C: Notation and basic variable types

Some meta notation: code for talking about code

Notation	Meaning	
X ⬅ 1.23	**Set** variable X to value 1.23	(action, changes X)
X ☞ 1.23	Variable X **contains** value 1.23	(indicates current state of X)

Setting is done in Python with an "assignment statement":

```
X = 1.23
```

- Creates **variable X** and sets its value to **1.23**
- Alternative description: stores **1.23** in variable **X**
- X ⬅ 1.23

Basic variable types:

Lists of characters or "strings" (str):

Code	**Interpretation**	**Outcome**
X = "Maxwell"	X ⬅ "Maxwell"	X ☞ "Maxwell"
Y = "School"	Y ⬅ "School"	Y ☞ "School"
Z = X + Y	Z ⬅ X+Y	Z ☞ "MaxwellSchool"

Integers (int):

Code	Interpretation	Outcome
X = 4	X ⬅ 4	X ☞ 4
Y = 123	Y ⬅ 123	Y ☞ 123
Z = X + Y	Z ⬅ X+Y	Z ☞ 127

Floating point numbers have decimal portions (float):

Code	Interpretation	Outcome
X = 45.67	X ⬅ 45.67	X ☞ 45.67
Y = 11.11	Y ⬅ 11.11	Y ☞ 11.11
Z = X + Y	Z ⬅ X+Y	Z ☞ 56.78

Can be tricky with **numbers** in **strings**:

Case 1: unexpected result

Code	Interpretation	Outcome
X = "4"	X ⬅ "4"	X ☞ "4"
Y = "123"	Y ⬅ "123"	Y ☞ "123"
Z = X + Y	Z ⬅ X+Y	Z ☞ "4123" (not 127)

Case 2: error message

Code	Interpretation	Outcome
X = 4	X ⬅ 4	X ☞ 4 (int)
Y = "123"	Y ⬅ "123"	Y ☞ "123" (string)

Z = X + Y $Z \boxed{\leftarrow} X+Y$ **error**

- unsupported operand type(s) for +: 'int' and 'str'
  Interpretation: can't carry out 'int' + 'str'

Detailed example: demo.py in g02