Types

A Weighty Problem

- **Given**: An assayer owns a balance scale that is used to weigh ore samples. The samples are always placed on one pan and the weights on the other.
- **Find** a set of 5 weights that can be used to weigh any sample from 1 to 31 ounces.
- Is there any other set of 5 weights that would work?

Doing Chores

Albert, Betty, and Chris leave the barn on an early winter morning from the lower right corner at positions A, B, and C, respectively. Each has a chore to accomplish in the pasture before letting the horses out. Albert has to break the ice on the water trough at A', Betty has to repair a section of the fence at B', and Chris has to dump a bale of hay in the feeder at C'. A fresh blanket of snow has covered the pasture, and this gives Albert, a computer science major at Calvin College, an idea. He wonders: “Can all three of us do our chores without crossing paths made in the snow? ”

Doing Chores

![Diagram of the barn and pasture with positions A, B, C, A', B', C', and the towers labeled B' with a star and A with a square.]

Idle Ivan

Idle Ivan was lounging by a river trying to figure out a way to increase the amount of coins that he had in his pocket without doing a lot of work. A leprechaun appeared and made him a proposition. The leprechaun said he would double the money in Ivan’s pocket every time he crossed the bridge. All he asked in return was a payment of eight coins after each bridge crossing (and after doubling the money). Ivan accepted the proposition. He crossed the bridge for the first time, and his money doubled. He paid the leprechaun eight coins and crossed again. His money doubled again, and he again paid the leprechaun eight more coins. He crossed for a third time, and his money doubled yet again. However, he only had eight coins left and had to give them all to the leprechaun, thus leaving him broke.

- How many coins did Ivan start with?
- If Ivan started with a different number of coins, could he make a profit? If so, what is the minimum number of coins required?

How would you print:

%/'crazy-+\4

A. `print("%/'crazy-+\4")`
B. `print("%/'crazy-+\4")`
C. `print("%/'crazy-+\4")`
D. `print(r"%/'crazy-+\4")`
E. None of the above
Assume \( x = 'alphabet' \)

How could you get the \( b \)?

A. \( x(6) \)
B. \( x[6] \)
C. \( x[5] \)
D. \( x(-4) \)
E. \( x[-3] \)

---

What will the following code produce?

\( x = 'cut' \)

\( x[1] = 'a' \)

\( \text{print}(x) \)

• Error! String objects are immutable (i.e. cannot change!)

---

Draw the memory associated with the following:

\( s = "test" \)

\( s.\text{upper}() \)

\( \text{print}(s) \)

• Note that the \text{upper}() method creates a string with all uppercase letters.

---

\( a = 'hi' \)

\( b = a \)

\( a = 'bye' \)

\( \text{print}(b[0]) \)

• Output is:
  A. \( h \)
  B. \( b \)
  C. \( \text{error} \)
• Justify your answer with a picture

---

\( \text{print}(a[3]) \)

• If this statement does not produce an error, what could \( a \) be?
  A. dictionary
  B. list
  C. string
  D. A or C
  E. B or C
  F. A or B or C
  G. None of the above

---

\( A = [2, 3, 5] \)

\( B = A \)

\( B[0] = 100 \)

\( C = B + A \)

\( \text{print}(C) \)

A. \([2, 3, 5, 100, 3, 5]\)
B. \([100, 3, 5, 100, 3, 5]\)
C. \([102, 6, 10]\)
D. \([2, 3, 5, 2, 3, 5]\)
E. None of the above
A = [1, 2, 3]
B = A
B[1] = 'knight'
print (A)

A. [1, 2, 3]
B. ['knight', 2, 3]
C. [1, 'knight', 3]
D. error on line 3
E. error on line 4

Create a list named h with 4 elements that cannot be changed?
A. h = [1, 2, 3, 4]
B. const h = [1, 2, 3, 4]
C. h = (1, 2, 3, 4)
D. H = [1, 2, 3, 4]
E. This is impossible in python

ids = ('32', '52', '87')
ids.append('49')
print(ids)

A. ('32', '52', '87', '49')
B. Error line 1
C. Error line 2
D. Error line 3

A python dictionary is used to represent:
A. Associations
B. Strings
C. Lexicons
D. ????

Which do you use to create a new dictionary:
A. x = []
B. x = ()
C. x = <>
D. x = {

temps = {} 
temps['july'] = 87.2 
temps['jan'] = 12.8 
print(temps)

A. {'jan': 12.8, 'july': 87.2}
B. [12.8 : 'jan', 87.2 : 'july']
C. ['jan': 12.8, 'july': 87.2]
D. {12.8 : 'jan'; 87.2 : 'july'}
Which of the following is/are not Python types?

A. int  
B. float  
C. str  
D. list  
E. tuple  
F. dictionary

Binary Representations

- Binary representations use a base-2 positional numbering system.  
- This system has only two “binary digits” (aka bits):  
  - 0 (or “off”)  
  - 1 (or “on”)

Decimal Numbers

Decimal numbers are base-10 (using digits 0-9)

123  
1*10^2 + 2*10^1 + 3*10^0  
1*100 + 2*10 + 3*1  
100 + 20 + 3

Binary Numbers

Binary numbers are base-2 (using digits 0 & 1)

110_2  
1*2^2 + 1*2^1 + 0*2^0  
1*4 + 1*2 + 0*1  
4 + 2 + 0 = 6_{10}

Binary Implications

- Binary/decimal conversions  
- Are there numbers that cannot be represented?  
  - Overflow, underflow, truncation

Representing Integers

- Integers are represented in twos-complement notation  
- Here, the high-order bit indicates the sign:  
  2_{10} = 0000000000000010  
  1_{10} = 0000000000000001  
  0_{10} = 0000000000000000  
  -1_{10} = 1111111111111111  
  -2_{10} = 1111111111111110  
- This example is shown in 16 bits, but 32 are 64 are common.
Representing Real Numbers

Real values are often represented in 64 bits using the IEEE floating point standard:

- **Exponent**: (11 bits)
- **Sign**: (1 bit)
- **Mantissa**: (52 bits)