```python
# Tutorial 2
# Python Foundations: Data Types
# B. Schoen-Phelan
# TU856/858 OOP-y2
# Sep 2020

# Data Types: which one do I have?

# booleans in Python are called bool
print(type(True)) #output is <class 'bool'>
print(type(False))

# integer holds whole values, their positives
# negatives and 0
print(type(34)) #output is <class 'int'>
print(type(0))
print(type(-34))

# numbers with a decimal point: floating point numbers
# holds positive and negative values and 0
print(type(0.0))
print(type(3.4))
print(type(-3.4))
print(type(5e10))
print(type(-5e10))

# Strings: sequences of characters enclosed by quotes
print(type("what am I"))
print(type('what am I'))
print(type(''))
print(type(" "))

# games with quotation marks
print(('I've been programming Python for several
years')) #gives an error
print("I've been programming Python for several years")
# this works
```

```python
print('I\'ve been programming Python for several
years') #this also works
print("'hello world'") #works
print('"hello world"') #also works
print(''hello world'') #doesn't work
print("hello "world" ") #doesn't work

print(type(7))
print(type("7"))

print(7+3)
print("7"+"3")
print(7+"3") #throws an error
print('7+3')

# now with variable names
# a variable is like a storage box for any kind of
# information or data
# what is in the variable can be accessed and changed
# any time you want
user_name = "Bianca"
# try with following hint type
print("User name: ",user_name)
#
user_name2, user_age = "Bianca", 21
print(user_name2, user_age)
#
user_name = 34 # hint type will be ignored here
print("User name: ",user_name)

2users = "Bianca and Susan" # throws and error

_users = "Bianca and Susan" # allowed but 'special'

__users = "Bianca and Susan" # allowed but even more
'special'
```

```
two_users = "Bianca and Susan"
print(two_users)

print("The Users", two_users)

# build strings
user_name = "Bianca"
last_name = "Phelan"
print(user_name+' '+last_name)
flight_number = 3456
print("Can passenger "+user_name+" please make her way
to flight number "+ flight_number) #will throw an error
print("Can passenger "+user_name+" please make her way
to flight number", flight_number) #works
print("Can passanger %s please make her way to flight
number %d" %(user_name, flight_number))

print("%30d"%(flight_number))

# format decimal numbers
print("%7.2f"%(1.2364)) #7 spaces in total allowed, 2
after the decimal point, does rounding
print('%7.2f' %(1.2364)) #7 spaces in total allowed, 2
after the decimal point, does rounding
print('%06.2f' % (1.2364))

# now with format
user_name = "Bianca"
flight = 123
message = "Can passenger {0:s} please go to gate {1:d}
for flight {2:d}".format(user_name, 10, flight)

message = "Can passenger {} please go to gate {} for
flight {}".format(user_name, 10, flight)

print(message)
```

```
# many string functions
print("This is a string".count('s'))
print("This is a string".count('s',4))
print("This is a string".count('s',4,10))
print("This is a string".count('S'))

print('Postman'.endswith('man'))
print('Postman'.endswith('man',3))
print('Postman'.endswith('man',2,6)) #change to ma to
make True

# find and index
print('This is a string'.find('s'))
print('This is a string'.find('s',4))
print('This is a string'.index('a'))

# is alphanum
print('123abc'.isalnum()) #try abc and 123 alone
print('1 23abc'.isalnum())
print('abc'.isalpha()) #try abc
print('123'.isnumeric())
print('123'.isdigit())
print('四'.isnumeric())
print('四'.isdigit())

#join
my_string = "Hello World"
separator ='-'
print(separator.join(my_string))

print('This is a string'.split())
print('This, is, a, string'.split(',',2))
print("   This is a string  ".strip())
print("This is a string".strip('s')) # only looking at
leading or trailing
print("This is a string".strip('T'))
print("This is a string".strip('g'))
```

```python
#lists
list_name = []
print(list_name)
list_name = ['Bianca', 'Susan']
print(list_name)
list_name.append('Cathy')
print(list_name)

# user_age = [10, 20, 30, 40, 50, 60, 70]
print(user_age)
print(user_age[1:3])
print(user_age[0:6:2])
print(user_age[3:])
print(user_age[:3])
print(user_age[:])
print(user_age[::2])




user_age[2] = 66
print(user_age)
del user_age[2]
print(user_age)
print(len(user_age))

my_list1 = [1,2,3]
my_list2 = ['a','b','c']
my_list1.append(my_list2)
print(my_list1)
my_list1.extend(my_list2)

print(my_list1+my_list2) #concatenation
print(my_list1)
print(my_list1.pop()) #put a value into pop
print(my_list1)
print(my_list1.remove('a'))
print(my_list1)

my_list1.reverse()
print(my_list1)

my_list3 = sorted(my_list1) # causes an error
my_list4 = [5,10,2,90,1]
my_list4.sort() #changes the list in place, no copy
returned
print(my_list4)

print(sorted(my_list4)) #creates a copy and returns it
print(my_list4) #original unchanged

print(my_list4*2)

my_tuple = ('Jan', 1, 'Mar')
print(my_tuple)

print(int('3')+3) #casting
```