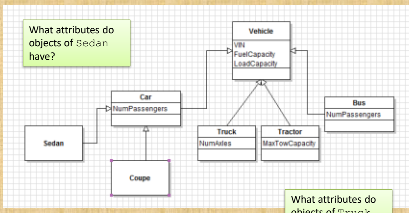
CSCI 204: Data Structures & Algorithms

Modules and Exceptions

WHAT'S LEFT LAST TIME

Example: Vehicles



What attributes do objects of `Sedan` have?

What attributes do objects of `Truck` have?

Design Exercise

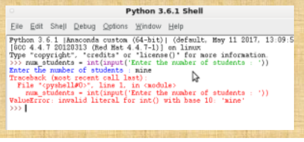
- Take out your computer
- Write the code for class Vehicle and its subclasses Car and Truck in a file named *vehicle.py*
- Write the code in a separate file named *vehicle_app.py* for testing the Vehicle class that creates a few Car and Truck objects and prints their information.

"Exceptions" in Python

Look back the example we had last time.

```
num_students = int(input('Enter the number of students : '))
```

What if we typed a non-numerical input?



```
Python 3.6.1 |Anaconda custom (64-bit)| (default, May 11 2017, 13:09:5
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "copyright", "credits" or "license()" for more information.
>>> num_students = int(input('Enter the number of students : '))
Enter the number of students : mine
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    num_students = int(input('Enter the number of students : '))
ValueError: invalid literal for int() with base 10: 'mine'
>>> 
```

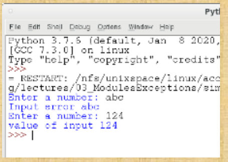
When "Exception" Happens

- Python will complain and stop execution of the program
- What if we want to handle the case(s) ourselves so we can control the program execution?
- In the example above, we'd like to ask the user to try again if the input is wrong.
- For example, we want the user to input a numerical value, we can also require a specific value range!

Python Defined Exceptions

```
num = 0
while True:
  try:
     v = input("Enter a number: ")
     num = int(v)
     break
  except Exception :
           print('Input error ' + v)

print('value of input ' + str(num))
```

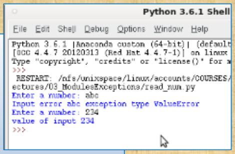
simple_exception.py



```
Python 3.7.6 (default, Jan  8 2020,
[GCC 7.3.0] on linux
Type "help", "copyright", "credits"
>>>
= RESTART: /nfs/unixspace/linux/acc
g/lectures/03_ModulesExceptions/sim
Enter a number: abc
Input error abc
Enter a number: 124
value of input 124
>>> |
```

Find the Type of Exceptions

```
num = 0
while True:
  try:
     v = input("Enter a number: ")
     num = int(v)
     break
  except Exception as ex:
     # if v is a non-numeric string, the type of exception is "ValueError“
     print('Input error ' + v + ' exception type ' + type(ex).__name__)

print('value of input ' + str(num))
```



```
Python 3.6.1 |Anaconda custom (64-bit)| (default
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "copyright", "credits" or "license()" for m
>>>
 RESTART: /nfs/unixspace/linux/accounts/COURSES/
ectures/03_ModulesExceptions/read_num.py
Enter a number: abc
Input error abc exception type ValueError
Enter a number: 234
value of input 234
>>>
```

read_num.py

Catch a Specific Exception

```
num = 0
while True:
  try:
     v = input("Enter a number: ")
     num = int(v)
     break
  except ValueError:
     # if v is a non-numeric string, the type of exception is "ValueError“
     print(‘Value Error’)

print('value of input ' + str(num))
```

catach_value_error.py

User Defined Exceptions

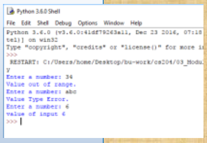
- In the above example, we used Python defined *Exception* or *ValueError* exception.
- There are many pre-defined exceptions
  - https://docs.python.org/3/library/exceptions.html
- There are occasions in which the programmers want their own exceptions.
- For example, we want to control the range of input, in addition to the type being int.

Try 1: use conditions

- Get out your computer, write a Python program segment based on the program in the previous example to enforce the range of input values. Let’s try to use conditions first.

Try 1: use conditions

```
num = 0
low_limit = 0
hi_limit = 10
while True:
  try:
     v = input("Enter a number: ")
     num = int(v)
     if num >= low_limit and num <= hi_limit:
        break
     else:
        print(‘Value out of range.’)
  except ValueError:
     # if v is a non-numeric string, the type of exception is "ValueError“
     print(‘Value Type Error’)

print('value of input ' + str(num))
```



```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18
tel)] on win32
Type "copyright", "credits" or "license()" for more i
>>>
 RESTART: C:/Users/home/Desktop/bu-work/cs204/03_Modu
y
Enter a number: 34
Value out of range.
Enter a number: abc
Value Type Error.
Enter a number: 6
value of input 6
>>> |
```

limit_range.py

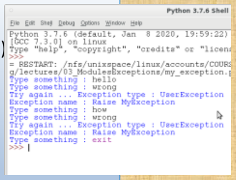
Try 2: define your own exception

```
class UserException(Exception):

    def __init__(self):
        self.__name__ = ‘MyException’

    def __str__(self):
        return ‘Raise ‘ + self.__name__
```

my_exception.py

Use the exception you defined

```
while True:
  try:
    value = input('Type something : ')
    if value == 'exit':
      break
    elif value == ‘wrong’:
      raise UserException
  except UserException as ex:
    print('Try again ... type ‘ + type(ex).__name__)
    print(‘Exception name : ‘ + str(ex))
```



```
Python 3.7.6 (default, Jan  8 2020, 19:59:22)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "licen
>>>
= RESTART: /nfs/unixspace/linux/accounts/COURS
g/lectures/03_ModulesExceptions/my_exception.p
Type something : hello
Type something : wrong
Try again ... Exception type : UserException
Exception name : Raise MyException
Type something : how
Type something : wrong
Try again ... Exception type : UserException
Exception name : Raise MyException
Type something : exit
>>> |
```

my_exception.py

Your Exercise

- Define a Python exception class in a separate file named “*value_exception.py*” to handle the cases when a value is out of range.
- Use the exception class to enforce that a user must type in an integer in a given range.
- Use the user-defined exception in a program “*limit_range.py*”