

(Refer Slide Time: 00:02)

```
File Edit View Shell Help 1
1. Program on If statement

# Sample Python/Pygame Programs
# http://programarcadegames.com/

# Variables used in the example if statements
a = 4
b = 5
c = 6

# Basic comparisons
if a < b:
    print('a is less than b')

if a > b:
    print('a is greater than b')

if a <= b:
    print('a is less than or equal to b')

if a >= b:
    print('a is greater than or equal to b')
```

Hi everyone again welcome to this lecture of the Linux programming class today we are going to actually talk about some examples on Python they went through an extensive study of Python in the last several lectures culminating in the CGI programming essentially so today we are going to just look at some of the programs how to write it in Python and then I will just go through the examples and give you some hints.

And then pretty much like I mean all that we have already learnt and then if any you have any more questions PS can answer those questions so the first one is actually a program on the conditionals example the if statement here so the here we are going to actually like I mean all these programs are there in this particular website you can go and edit these are basically in Python or fight game programs.

And then in this particular program basically we have like several variables so the variable a that is defined as 4 b is defined as 5 defined as six and once we have these three variables defined now we can do to some basic comparisons we can see that actually so the first statement is if a < B then we just print A < B if a is > B then print A is < B so in this case which one do you think will be printed out obviously like this.

So we should look for hope this 1 is < D and then if A is <= B then it prints out a <=B and we have all these things define therefore let us see how the program the outputs.

(Refer Slide Time: 02:54)

```

Python Programming.pdf - Adobe Reader
File Edit View Window Help

if a == "B" or "b":
    print("a is equal to b. Maybe.")

# This is the proper way to do the if statement.
if a == "B" or a == "b":
    print("a is equal to b.")

# Example 1: If statement
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
print("Done")

# Example 2: Else statement
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
else:

```

Now that is the scroll a little bit down if A is >= B then it ends with an are we going to be a simple program and then you can see that basically like it follows our unique invention like this is the expression and then the : actually separates And then there is an intent for the, the print statement and then after that the print actually is there so that all the intended ones are part of this if statement.

Now the == the tendency to mix them and you know that actually like = is an assignment and w = a logical check essentially so that is what is in this comments and you know like I am involving should be coming straight back comments as well if a == B which is basically like a logical check whether it is not equal to then print this and the not equal to is bang then you put it is not equal to then it will be this now if a is < B and a is < C.

Then it moves up this you then and non-exclusive is a < B are a < P then print now there is a Boolean data type which is a = true and then this is actually legal so again you can do this assignment a true group and then if a then B is true which is what it is going to be and then if not a then one day is false so now if a is = true and B through the false and then a and B then a and B both the true.

Now we make another for few more attendance in A=3 b = 3 and C =A==B so now then print see so now let us see what ,what that will output and then as you know like I mean in Python a one is actually a true value so one print one if a print a anything that is 0 exactly what is also something so and then we also do this if 0 then print 0 and this will not trigger because the 0 is false so this expression is false.

Now another comparison comparing variables to multiple values the first if statement basically like here in this one is no paperwork but it always trigger as true even if the variable a is not

equal to B because b itself is considered through B alone so will equal to see if a == B or B then print I = B maybe so basically like even if a is not equal to be this, this will be triggered because our B is physically the proper way to put it is basically.

Going to say in B or A == the lower case T then it will never do trigger, so now the other one basic temperature

= int input what is the temperature in Fahrenheit so now you know that actually like input is query to get a value from the screen and then this is the question and essentially like I mean whatever the number but to type in it is converted into the Indies data type and then that is assigned temperature. And then if temperature is greater than 90 then print it is not outside and then it prints done so you see basically look again this goes with the print statement goes with the F and then the other print is actually stays outside so whether this true or false basically of this print done with all these print done and then there is an else class that you get defined so I am bills of the like all in and then print it is not hot outside.

(Refer Slide Time: 08:32)

```
print("It is not hot outside")
print("Done")

#Example 3: Else if statement
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
elif temperature < 30:
    print("It is cold outside")
else:
    print("It is not hot outside")
print("Done")

# Example 4: Ordering of statements
# Something with this is wrong. What?
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
elif temperature > 110:
```

And then again print them now the else if statement which is the third category this is also like we saw essentially here we defined the if statement then the else if statement and then finally else statement that actually the ELSE form after the this is the second conditional which is the temperature <30 and then we need to put a colon after that so with after each of the expressions we put a : in to denote that it at the end of the expression And then the remaining ones are the block of code that.

(Refer Slide Time: 09:14)

```
File Edit View Window Help
# Example 3: Use if statement
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
elif temperature < 30:
    print("It is cold outside")
else:
    print("It is not hot outside")
print("Done")

# Example 4: Ordering of statements
# Something with this is wrong. What?
temperature = int(input("What is the temperature in Fahrenheit? "))
if temperature > 90:
    print("It is hot outside")
elif temperature > 110:
    print("Oh man, you could fry eggs on the pavement!")
elif temperature < 30:
    print("It is cold outside")
else:
    print("It is ok outside")
print("Done")

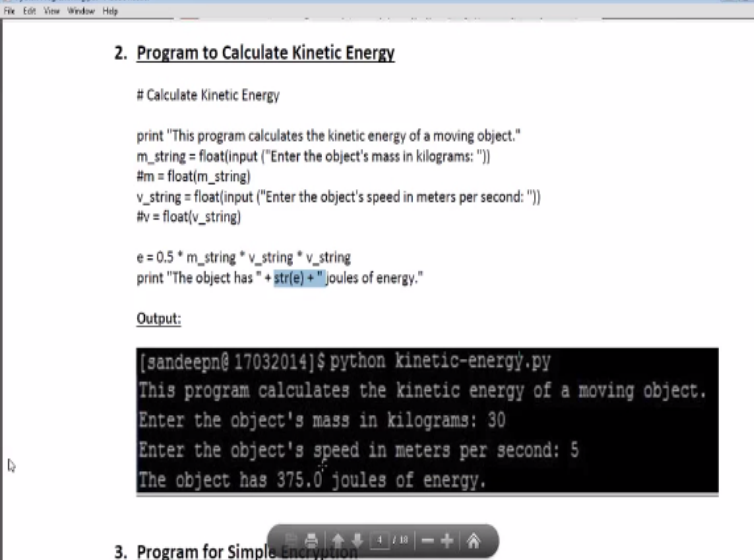
Output:
```

Now the ordering of statements the basically the temperature basically this is the one and then it is not outside it is > 110 then oh man you could fry eggs the payment so what is the mistake here if you look at it basically will give this > 90 it says it is odd how tough side then only if this condition is false then it will go into the next one so if it is >90 it will print this and then it will exit not so this condition will never get executed. So in this one the proper improper ordering should be like I mean this should be first one then that then the 30 and then it is okay outside and then come out let us see what are the things basically we see here so the first one we saw that it A is< B so it prints out just as A< B does not print out the actual values it so even though we have signed a s4 a as 5, from a distance a < B that is because we did not put the values and send output value. (Refer Slide Time: 11:09)

```
File Edit View Window Help
Output:
```

```
[sandeepn@17032014]$python if_statement_examples.py
a is less than b
a is less than or equal to b
a and b are not equal
a is less than b and c
a is less than either a or b (or both)
a is true
True
1
A
a is equal to b. Maybe.
What is the temperature in Fahrenheit? 100
It is hot outside
Done
What is the temperature in Fahrenheit? 50
It is not hot outside
Done
What is the temperature in Fahrenheit? 20
It is cold outside
Done
What is the temperature in Fahrenheit? 10
It is cold outside
Done
```

So in the same thing how do we make sure that these things can be output as values or that we need to make sure that these are like the percentage bees and things like that again and , after the code we need to specify a and B so that the values themselves get printed you ,you so this is all the various command output I want you to go through this basically along with this particular all the programs this program pretty much action may need exercise. Is then if then so if else if else conditionals pretty well in including all the different expressions as well you see a good understanding of all the logical operations you can perform. (Refer Slide Time: 11:47)



```
2. Program to Calculate Kinetic Energy

# Calculate Kinetic Energy

print "This program calculates the kinetic energy of a moving object."
m_string = float(input("Enter the object's mass in kilograms: "))
#m = float(m_string)
v_string = float(input("Enter the object's speed in meters per second: "))
#v = float(v_string)

e = 0.5 * m_string * v_string * v_string
print "The object has " + str(e) + " joules of energy."

Output:

[sandeepn@17032014]$ python kinetic-energy.py
This program calculates the kinetic energy of a moving object.
Enter the object's mass in kilograms: 30
Enter the object's speed in meters per second: 5
The object has 375.0 joules of energy.
```

3. Program for Simple

So now let us go to the next one this is the program to calculate the kinetic energy so first of all what is the equation for kinetic energy or a moving object so basically we computed as the mass times velocity squared so and then times have the sequence of past times mass times because it is one that is the equation for avoiding the kinetic energy so we need to take the input as, as the mass and this is a floating point number then.

We also need to input the objects speed and then we will say basically, let us go you that us a the unit we define basically ,be cuddling the MPS units of the SI unit essentially you are familiar with those kind of terms so the SI unit basically the mass unit is kilograms and then the ,the speed of the velocity unit is in meters per second so in this example we start by saying that basically what with this program built.

So like this is the documentation line which is this program calculates the kinetic energy of moving object so now we define a variable m string and this is the floating point and then this is where we input the mass of the body this so enter the object smashing program when we leave it

and then we call it as an input and then we assign it to the float assign so now this basically this results in the prompt and then somebody has to enter as been.

A particular number and then that is getting a that it is designed to the string in the form of a floating point number you let us not worry about the comments here the comment is essentially like a miracle if you miss this float header you can always have this is a big one so that is what I am extensible now to get the speed of the this particular object we define another variable because the V string and then V string is also a float .

And then here we, we get the speed in meters per second and then once we have 2 numbers every student will see we can simply calculate the kinetic energy as times the mass times, velocity squared and then now we need to just output this one the form of a string so be that basically the e is complete floating point you need to convert it into a string in order to display this so to convert that into a string we basically use the STR you want them.

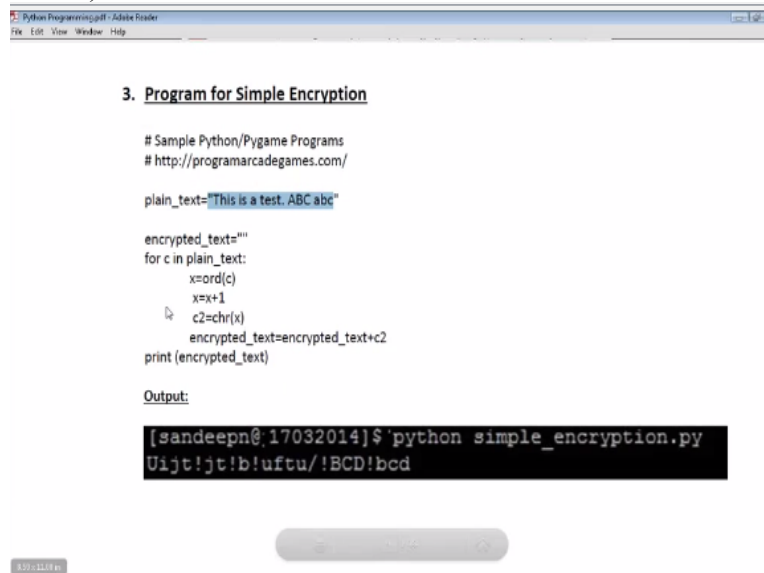
And then here what we are doing is basically like print first this particular phrase the object has and then we add the string E and then we also then print the remaining ones which is the joules of energy so here one simple example enter the mass in kilograms 30 objects speed in meters per second five and then outcomes basically 375 joules of energy and this is computed BC a half x 13 x 25 and then that is number.

So pretty straightforward you can also construct this in a different way where we can say basically like what object has and then the %d or because of it is kind of formatting spoon and then , directly do the e and then close campuses and that will directly print the value as a string it as a ,floating point number so certainly not and convert that into a string and then put the value out and then you can see that actually like it is converting them to string.

Because I mean the it is actually computing in the floating point because these one this once can be an integer but you know the put is actually like the integer.0 and then again if you do not specify this as a floating point and instead say integer here you are still going to get a floating point number for the ,the energy can you tell me why because if you remember actually like this is the same thing.

That is happening in TCL the starting number is the float and then all the remaining ones are automatically add this to that particular time in order to compute that number so even if we take the m string as the integer it will still result in the same thing the only thing here is we cannot put this as a string itself we have to take this as an integer or a floating point number so I would, I would urge you to try actually like change this float here To int and then this float to int and then try this program it should run exactly the same way you okay.

(Refer Slide Time: 18:33)



```
Python Programming - Adalek Reader
File Edit View Window Help

3. Program for Simple Encryption

# Sample Python/Pygame Programs
# http://programarcadegames.com/

plain_text="This is a test. ABC abc"

encrypted_text=""
for c in plain_text:
    x=ord(c)
    x=x+1
    c2=chr(x)
    encrypted_text=encrypted_text+c2
print(encrypted_text)

Output:

[sandeepn@17032014]$ python simple_encryption.py
Uijt!jt!b!uftu/!BCD!bcd
```

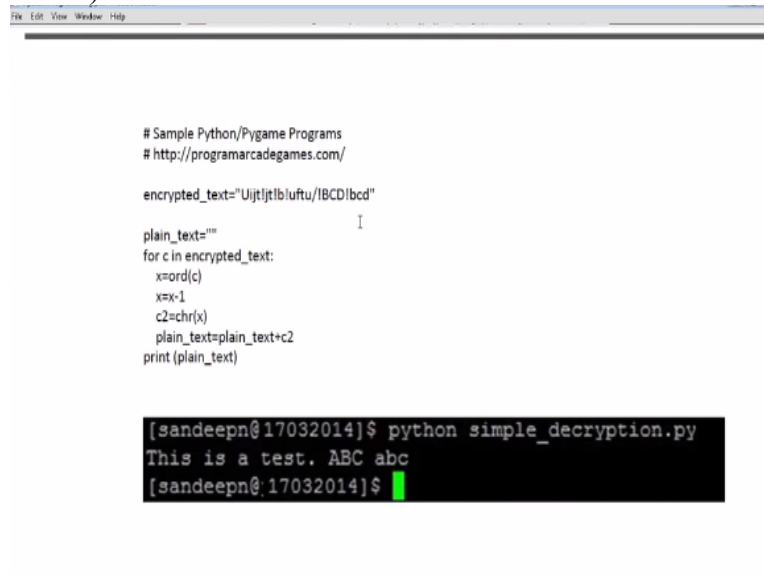
So now we go to the next one which is a program for a simple encryption so the encryption essentially is, we change the value of a text to a different one and we kind of input the, the text essentially so there is a simple algorithm here we will probably use this we have a plain text which is this is the phrase that we are going to encrypt so here we can specify any anything basically for testing purposes.

But make sure that you use the encryption and then the decryption followed by that in order to verify whether your encryption program is working correctly, so the plaintext is the string which is this is a test ABC ,ABC the interpret text we define as an empty way for core course it is winners the intake spring now we start reading each character at the time on the main text so for see in plain text.

So we define X is the ORD of see essentially and then we essentially like we just say basic X is $X + 1$ and then we dump out a new character which is the character X basic and then the encrypted text is nothing but basically we just add this t2 to it to the entrance text will be appended one by one and then once this loop finishes need this printable into the text so let us see how this works again the, the particular text is essentially like this is a test ABC ,ABC so the ORD function as you know how this is this gives you in the in the alphabetical order where this particular character is and then this is the last one is basically the next character.

Is what is going to give so this becomes U I JT which is the U is next T is next to H days next to I and then T is next to S and then they are ever there is a space basically the next character with a bang so that is what gets printed out and then we are in the full stop is the \ the next one so these are all the is actually cactus from the architects.

So now this works and then we have some encrypted data set now let us do how do we decrypt this one it is fairly simple we capture this one now this is our new string and then we go for the odd of this one and then this reverse this one reverse that is X is X -1 and then we create the character set based on this so let us see how the program works.
(Refer Slide Time: 22:46)

A screenshot of a code editor window showing a Python script for decryption. The script defines an encrypted string and a function to decrypt it by shifting each character back by 2 positions. Below the code, a terminal window shows the command to run the script and the resulting output.

```
# Sample Python/Pygame Programs
# http://programarcadegames.com/

encrypted_text="Uijt|t|b|uftu|/BCD|bcd"

plain_text=""
for c in encrypted_text:
    x=ord(c)
    x=x-1
    c=chr(x)
    plain_text=plain_text+c
print(plain_text)
```

```
[sandeepn@17032014]$ python simple_decryption.py
This is a test. ABC abc
[sandeepn@17032014]$
```

So as I mentioned we have the interpret texts then we say basically we go and find work for an ordinal of particular character and then we basically say x is x-1 and then we build a character set and then we open up so now when we decrypt this particular string we get back our original string which is this is a test ABC ,ABC I think this one is pretty interesting and pretty straightforward you can actually try with new stuff and in fact what you should do is you should put both these programs into a single program.

And instead of actually defining this as a , as a string you should get this string from on the command line and then you can have fun with it tends to you can first output the input string that is print the interpret a text and then later on you get you users can decrypt that and then print the big mystery and then something else to try is basically how do you instead of the encoding algorithm as the X is =X + 1.

Try new ones basically meaning X = X +5 or something basically and see how that works so these are all like some of the fun things that you can do with this program so I want you to experiment with it and then so that you feel more comfortable running the Python programs, okay.

(Refer Slide Time: 24:35)


```
Python Programming - Aditya Rastogi
File Edit View Window Help

5. Program for Exception Handling

import sys

# Divide by zero
try:
    x = 5/0
except:
    print("Error dividing by zero")
    print(sys.exc_info()[0])

# Invalid number conversion
try:
    x = int("fred")
except:
    print("Error converting fred to a number")
```

So the next one is a program for exception handling this is something that we saw like I mean how to handle the exceptions in fact the same thing like the try and except where the commands that we saw so here we have to import this particular package the system that the we also like learnt about how to input this in pretty much like last but one lecture I think so first we will find out what the /0 error is, so we define try and then $X = 5 / 0$ except.

Now we say print error / 0 and then we give also like a exception info bit basically like 0 so we need to print this particular exe info 01 so we will see like how that gets picked it out and then there are some more things visiting like a valid number conversion so if you say like int thread and that is assigned to X.

(Refer Slide Time: 26:20)

```
Python Programming - Aditya Rastogi
File Edit View Window Help

# Error opening file
try:
    f = open('myfile.txt')
except:
    print("Error opening file")

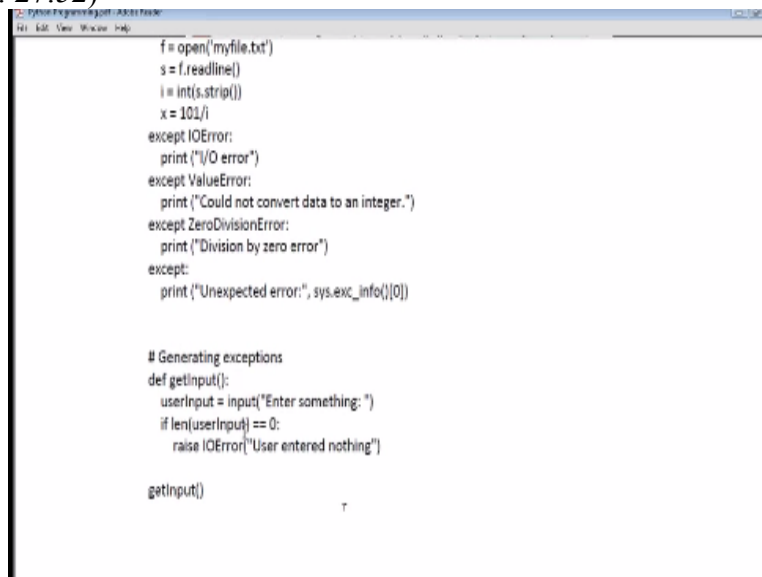
# Multiple errors
try:
    f = open('myfile.txt')
    s = f.readline()
    i = int(s.strip())
    x = 101/i
except IOError:
    print("/I/O error")
except ValueError:
    print("Could not convert data to an integer.")
except ZeroDivisionError:
    print("Division by zero error")
except:
    print("Unexpected error:", sys.exc_info()[0])

# Generating exceptions
```

So here we say basically like a Fred to a number and then we also like give some of the system exception and then so number entered equal to false while the number entered equal to false number string is input enter integer so here we are very few on converting the number string to an integer and the number enter equally true and then if it is not then you say basically like error invalid integer and then opening file types of error.

S= open my file. txt and then basically like if that file cannot be opened then it prints out are error opening file and then all these things can be combined basic you so if say basically gave AFO if A = open my file text s is = F . Read line so you know about this object and basically its method the red lines method and then I = int f . Strip that is another method and then we x is = 101 / I now you can print many of these X is an error IO error value error 0. Division error and then in some unexpected errors.

(Refer Slide Time: 27:52)



```
f = open('myfile.txt')
s = f.readline()
i = int(s.strip())
x = 101/i
except IOError:
    print("/O error")
except ValueError:
    print("Could not convert data to an integer.")
except ZeroDivisionError:
    print("Division by zero error")
except:
    print("Unexpected error:", sys.exc_info()[0])

# Generating exceptions
def getInput():
    userInput = input("Enter something: ")
    if len(userInput) == 0:
        raise IOError("User entered nothing")

getInput()
```

So now to generate exceptions essentially this is another function that we are defining basically it input and then user input is input campaign and then if the length is 0 that it is like for either is not entering anything basically then we just say IO error user entered nothing.

(Refer Slide Time: 28:15)

```
Output
[sandeepn@17032014]$ python exception_handling.py
Error dividing by zero
exceptions.ZeroDivisionError
Error converting fred to a number
exceptions.ValueError
Enter an integer: 5
Error opening file
I/O error
Enter something: 7
Traceback (most recent call last):
  File "exception_handling.py", line 58, in ?
    getInput()
  File "exception_handling.py", line 55, in getInput
    if len(userInput) == 0:
TypeError: len() of unsized object
[sandeepn@17032014]$
```

So now let us look at what are the outputs so the first one we actually did the ,the / 0 so the first one was the fight / or that we get the error \0 and they also get tensed we also like them to the system info system package for the particular exception it gives the accession info has a 0, now when we try to convert thread to the number we get a value as in error as I mean error, now when we try to enter the integer.

Integer number then try to convert that into an or we ask the user to enter so then the user enters 5 and then this is essentially like I mean it's basically, it goes through basically because this is there is no error so, so valid integer so it was so if you put anything something invalid then it will catch the invalid integer error, now the opening in the my files. Text essentially that is not there so basically it just prints there are opening file.

Now these are the things where we had the multiple ones so actually so for the opening file basically like I mean before that actually the so the IU error , so in this one essentially we need be so we had multiple error cases and the only thing there is essentially so then it is reading the line it has nothing basically so it is more like an i/o error that for done its output but it could be going yeah if there is a value and then value is 0 then we could have gotten a 0 division error and if something else has happened.

Then basically forces come out with an unexpected error now the next one is basically like meeting with into something which is seven and then , and then this one basically again in the user inputs length is 0 and then it can give these kind of errors basically when we try to run the get input , so that is the brief look in the exception generation now the next one is program on the on list.

(Refer Slide Time: 33:27)

```
Python Programming - Aki's Profile
File Edit View Window Help

6. Program on Lists

#!/usr/bin/python -tt
# Copyright 2010 Google Inc.
# Licensed under the Apache License, Version 2.0
# http://www.apache.org/licenses/LICENSE-2.0
# Google's Python Class
# http://code.google.com/edu/languages/google-python-class/
# Basic list exercises

# A. match_ends
# Given a list of strings, return the count of the number of
# strings where the string length is 2 or more and the first
# and last chars of the string are the same.
# Note: python does not have a ++ operator, but += works.

def match_ends(words):

    count = 0
    for word in words:
        if len(word) >= 2 and word[0] == word[-1]:
            count = count + 1
```

So this particular program is taken from Google's python class so again then go into this particular website this is this gives you the present class and basically like that several examples there so we will talk about few of them so here the main thing is the first one that we are going to look at it is matching the ends so given a list of strings return the count of number of strings where the string length.

Is two or more and the first and the last characters of the string have exactly the same so one thing to notice python does not have this increment operator and there is glass plugs it only has an incremental assignment which is += so keep in mind about that and that's what we will use so this particular, function is called the match ends and then the arguments are open so we define a counter essentially 0

And then forward in words essentially for each of the words we basically see if the, the length of word is there >= 2 and if the first one is not the same as the last one as word[0] == word[-1] so we already studied this these notations like the word[0] represents so actually the first element of the character of the string and then word[-1] represents one from the last one and then the last is just specific last, actually I get this denotes the last character to the -1.

(Refer Slide Time: 36:11)

```
Python Programming (off) - Akki Reddy
File Edit View Window Help

return count

# B. front_x
# Given a list of strings, return a list with the strings
# in sorted order, except group all the strings that begin with 'x' first.
# e.g. ['mix', 'xyz', 'apple', 'xanadu', 'aardvark'] yields
# ['xanadu', 'xyz', 'aardvark', 'apple', 'mix']
# Hint: this can be done by making 2 lists and sorting each of them
# before combining them.

def front_x(words):

    # Put each word into the x_list or the other_list.
    x_list = []
    other_list = []
    for w in words:
        if w.startswith('x'):
            x_list.append(w)
        else:
            other_list.append(w)
    return sorted(x_list) + sorted(other_list)

# Extract the last element from a tuple -- used for custom sorting below.
def last(a):
    return a[-1]

# C. count last
```

So if this is the same then we increase the pound and then we found our work essentially exists, so we just returned the count for how many such things can exist so we will feel like I puts an output of the stem programs later now this second one is actually it's called front on this 4x this is the list of strings return a list with the strings in sorted order each exit the group all the strings that begin with X first.

So if I like mix the XYZ Apple artwork it result in X anode x 1 z are raw apple and so here the hint is basically like we can do it by making two lists and sorting each of them before combining them and do you know like I mean so one list is the all the words that begin with X and then the second one is the regular started one and then we can actually 4-6.

So ,so the way we will do it is basically all the things so it's actually all the things that start x and all the things that does not or do not start with X so those are the two lists that will so it's called front and the score x and then again we will pass the list of words, so we say basically X list and other list those are the two lists that the imaging so we define them as the empty list and then if the w starts with X we think so we know that start with this method which can be applied to this w object which is the word inside the list.

So then we basically happen this word into the interest it is not true then we will append the word to the other list some simple missing so X is not happened w else other list not append w and then once we have this then we just thought the x list separately and then sort the other lists of it so what we returned actually is sorted X list plus sorted other list, so again very simple as you can see basically was only like one two three four five six seven eight lines.

And then we could achieve that move things now the next example is to extract the last element from tuple use for custom sorting so the last element is denoted as we negative 1 so we basically get the last one and then that is a pretty much.

(Refer Slide Time: 39:43)

```
File Edit View Window Help
for w in words:
    if w.startswith('x'):
        x_list.append(w)
    else:
        other_list.append(w)
return sorted(x_list) + sorted(other_list)

# Extract the last element from a tuple -- used for custom sorting below.
def last(a):
    return a[-1]

# C. sort_last
# Given a list of non-empty tuples, return a list sorted in increasing
# order by the last element in each tuple.
# e.g. [(1, 7), (1, 3), (3, 4, 5), (2, 2)] yields
# [(2, 2), (1, 3), (3, 4, 5), (1, 7)]
# Hint: use a custom key= function to extract the last element form each tuple.

def sort_last(tuples):
    return sorted(tuples, key=last)

# Simple provided test() function used in main() to print
# what each function returns vs. what it's supposed to return.
def test(got, expected):
    if got == expected:
        prefix = 'OK'
    else:
        prefix = 'FAIL'
```

When the next program is basically to start for last essentially so given the list of non empty tables return a list sorted in the increasing order by the last element that means basically we have in integers 1713 3 4 5 22 then started on the last element which is 22 13 3 4 5 and, so here the way that we will be using is basically we are using a custom p equal function to extract the last element for me to the two.

But let us see how we do with it again it is just the two lines so start list to triples is the function and then basically what we do is with a started tuples and then where z equal to last may be easily written , so now let's see the next one which is this function used in Main to print basically so happened what each function returns this is what it is the post active so we define this function pearl test got.

(Refer Slide Time: 42:03)

```
Python Programming - Jupyter Notebook
File Edit View Window Help

def sort_last(tuples):
    return sorted(tuples, key=lambda t: t[-1])

# Simple provided test() function used in main() to print
# what each function returns vs. what it's supposed to return.
def test(got, expected):
    if got == expected:
        prefix = 'OK'
    else:
        prefix = 'X'
    print '%s got: %s expected: %s' % (prefix, repr(got), repr(expected))

# Calls the above functions with interesting inputs.
def main():
    print 'match_ends'
```

I expected those are the two things if God equal to expected then we will fix it ok else will profit with not okay which is X so here we say basically make print X / presentation s got and present it s expected and we define this basically like this prefix and then what is we get and then put what we get and what we expected , so in order to test all these functions that we defined earlier we will give like a number of programs and then. Basically will call this program multiple times with various arguments and then test letter this holds but, so the first one we will be testing the match ends or that we give like various, string there is so you so there is some strength basically like that very list and then we list it out and then we apply various scenarios to test it .

(Refer Slide Time: 43:32)

```
File Edit View Window Help

test(match_ends(['aba', 'xyz', 'aa', 'x', 'bbb']), 3)
test(match_ends(['', 'x', 'xy', 'xyz', 'xx']), 2)
test(match_ends(['aaa', 'be', 'abc', 'hello']), 1)

print
print 'front_x'
test(front_x(['bbb', 'ccc', 'axx', 'xzz', 'xaa']),
       ['xaa', 'xzz', 'axx', 'bbb', 'ccc'])
test(front_x(['ccc', 'bbb', 'aaa', 'xcc', 'xaa']),
       ['xaa', 'xcc', 'aaa', 'bbb', 'ccc'])
test(front_x(['mix', 'xyz', 'apple', 'xanadu', 'aardvark']),
       ['xanadu', 'xyz', 'aardvark', 'apple', 'mix'])

print
print 'sort_last'
test(sort_last([(1, 3), (3, 2), (2, 1)]),
      [(2, 1), (3, 2), (1, 3)])
test(sort_last([(2, 3), (1, 2), (3, 1)]),
      [(3, 1), (1, 2), (2, 3)])
test(sort_last([(1, 7), (1, 3), (3, 4, 5), (2, 2)]),
      [(2, 2), (1, 3), (3, 4, 5), (1, 7)])

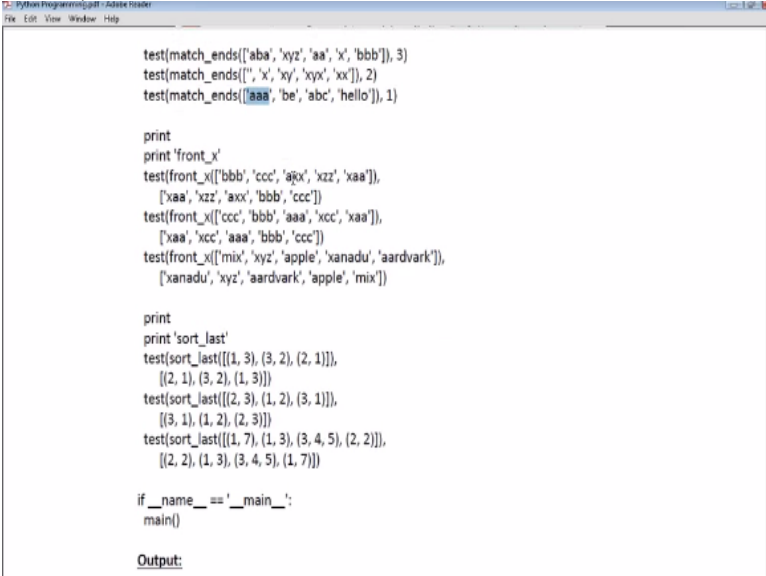
if __name__ == '__main__':
```

We also base the front X here and then we will also after the, got last one , so for the match ends pretty much we got perfect so let us look at our test vehicle and see basically where we can undo the thing so the first one is actually it identified three such matches so can we identify three such like this one is so okay this is one, this is the one and the third one is here so anything it is over I mean two or more is basically.

The target and then the first one is equal to last one because that is the so we know that actually in this list there are three and then that's what we expected so when we run the program we actually got through so it is it's a match now in this one actually our whole thing falls into like 123 these the elements but out of this only two of them are correct so we should get the expected x2 and then the last one there is only one.

That is really tight on the other ones are not matching for this one we actually successfully tested our little function and then basically we did 3x3 on the program.

(Refer Slide Time: 45:40)



```

test(match_ends(['aba', 'xyz', 'aa', 'x', 'bbb']), 3)
test(match_ends(['', 'x', 'xy', 'yx', 'x']), 2)
test(match_ends(['aaa', 'be', 'abc', 'hello']), 1)

print
print 'front_x'
test(front_x(['bbb', 'ccc', 'ajxc', 'xzz', 'xaa']),
      ['xaa', 'xzz', 'axc', 'bbb', 'ccc'])
test(front_x(['ccc', 'bbb', 'aaa', 'xcc', 'xaa']),
      ['xaa', 'xcc', 'aaa', 'bbb', 'ccc'])
test(front_x(['mix', 'xyz', 'apple', 'xanadu', 'aardvark']),
      ['xanadu', 'xyz', 'aardvark', 'apple', 'mix'])

print
print 'sort_last'
test(sort_last([(1, 3), (3, 2), (2, 1)]),
      [(2, 1), (3, 2), (1, 3)])
test(sort_last([(2, 3), (1, 2), (3, 1)]),
      [(3, 1), (1, 2), (2, 3)])
test(sort_last([(1, 7), (1, 3), (3, 4, 5), (2, 2)]),
      [(2, 2), (1, 3), (3, 4, 5), (1, 7)])

if __name__ == '__main__':
    main()

Output:

```

And then that for but now the next one which is the front - X so here as we said basically the something starts with X that needs to be sorted and then Bradley this needs to be appended before are appended before the other list which is sorted too so here is the string it is the list and then , they started one what we expect is AAA there is the last element followed by this second to last and then this and then bbb the CCC.

So same thing for the next one this is xaa- xzz -axx- BBB-C CC and similarly for the third one as well so here also like I mean we get okay and then we walk the same thing , finally the sort last script essentially which is so the number of triples basically again it is 133221 the started one

will be based on the last one which is 21,32,13 and then the same case for three and then all the three or more filling tested and we are getting okay.

So all these functions really work ,so try this out and then try out also like device how we can test those functions , so that is pretty much covers most of this one there will there are a few more examples that I want to go to and then I will probably go through it in the next lecture thanks.