* Hello. Hello, everybody. It's 630 and January 31st, the last day of January.

2:27

* And we are here for the second class sell this year.

2:38

* 624 So let me just start asking you if you have any question that is something you want to discuss.

2:43

* A There was something in the material worth reviewing somehow.

2:56

* So. Oh, things are going.

3:03

* All right. So. I mean that next time try to be more into questions.

3:13

* So to make it more interactive. So the whole experience of online is what it is.

3:24

* I mean, it's not like being in person, but let's try to make it more more personal somehow.

3:33

* So I'm glad that you are switching.

3:44

* I have a question. Yeah. Good. Really answer right now.

3:48

* So for one of the like when I came to to change to one,

3:53

* I once I finally got the software download and everything working when I came to change the name.

3:59

* There's no way to change the name of like like if you see the way I submitted it, it says homework like p16789 something, something or something.

4:04

* And I don't have any way to change that. But I was able to change it like I changed my homework name.

4:14

* I just couldn't change the name of the actual folder. Yeah.

4:20

* I mean, but I mean, the incumbents are own or in your own former.

4:23

* No in my folder and byton. Okay. So I mean, generally speaking, what you have on the end is in is in pie chart, right?

4:30

* Yes. I mean, in the pie chart. Let me share the screen and let me show you.

4:42

* That's probably easier. Okay.

4:54

* So that's. That's by Sharma.

4:57

* You have? Yes. Yes. You see all the way to the left? Yep. You see what's on the left.

5:02

* So instead of me being able to name the label the folder m600,

5:07

* I was able to to label it where it says exercise 00. P lives and able to change that name.

5:12

* But I don't know how to change the folder to 624.

5:18

* Yeah. I mean the only way it's basically to go here and to go in the folder like in this case that folder is EMC X 24.

5:22

* And is this folder here? Mm hmm. And just run the way.

5:38

* Yeah. You need to change here. Uh, okay.

5:42

* I was trying to change it from there, and it wasn't. Yeah, no, I mean, I'm not even sure if there is a rename here, but.

5:46

* No, no, no. Yeah. I mean, the only way would be changing here.

5:53

* And then when you will. And then I'm in the closet.

6:00

* Change it closer the entire program, change the name and then open with the new name.

6:05

* And that's. I think everything else is pretty straightforward.

6:11

* Yeah. Yeah. I mean, that is not a real file management area.

6:16

* I give you some things, but is more for displaying what is inside than for doing something active.

6:24

* Okay. Thank you. Sure. All right.

6:34

* Okay. So. Uh, let me keep sharing, actually.

6:38

* Go back and watch it.

6:47

* All right, so we are in module two, software engineering and some python coding.

6:52

* So. We had some issues last week with this quiz and 185, Check Your Knowledge.

7:03

* I republished it. Some of you already answered the question and know this as some of you already have

7:16

* knowledge of coding in general and knowledge of coding in Python in particular,

7:28

* but some other you do not.

7:35

* So the course up to the midterm is going to be quite basic meaning.

7:40

* I cannot leave people behind, meaning that I will start with the very basic elements of Python and then we will be up from after the midterm.

7:49

* Things will be more engaging for pretty much everybody.

8:04

* In the meantime, there will be other portions that are less strictly on Python and more on writing code,

8:09

* managing software, development projects and things like that.

8:20

* So. If you didn't do yet, please go back to the M1 B5, check your knowledge and do the quiz is very basic.

8:27

* And I mean, the two most important questions are, are you familiar with coding offers?

8:41

* Are you familiar with coding in Python disciple? All right.

8:48

* So. During the lecture today, we will talk about software engineering.

8:53

* A little bit of history.

9:02

* Some example is going to be just an introduction because we are going to have a more detailed section on those aspects in a few weeks.

9:04

* Then we will continue with the. By Donna, we will do an in-class exercise and then a will present.

9:16

* We will discuss the solution of the in-class exercise, and I will introduce the next homework.

9:34

* All right. So let me start the fourth.

9:47

* With some slides. So. The first group of lights is on developing software.

9:54

* So again, that is going to be relatively short.

10:04

* So software initially was just a part of our work, so that people were developing and installing the hardware was also the people doing the software.

10:08

* Then they realized that that there were competencies that should have been developed in different ways.

10:26

* And then that's the beginning of the.

10:34

* Jobs of the coders, programmers or whatever you want to call them.

10:39

* Initially that we are from the people who developed the hardware, then mathematicians and then develop the in its own discipline.

10:44

* When I already mentioned that when I graduated from the University of Rome.

10:58

* There was no information technology. It was a beginning of AIDS.

11:08

* There was no information technology.

11:14

* And I graduated as a master in math, and it was applied math.

11:18

* And part of the applied math was something close to information technology, but was not a separate discipline.

11:26

* And we are talking 1980.

11:36

* It is not 1880. So that's just to have a timeframe.

11:39

* When you develop software, that is a component that is all in terms, of course, that is the development course.

11:49

* And there is also a component that is on testing and testing.

11:57

* The software can take a big chunk of the overall costs.

12:01

* Generally speaking, you don't have the same people are doing the development and doing the testing because the goal

12:06

* of someone doing the development is basically to make it run and to make the code running.

12:15

* The goal for people doing the testing is just the opposite.

12:21

* To find the failure point, make the code fail.

12:25

* Two different mental attitudes to different skills.

12:33

* But we need the both of them in additional costs.

12:37

* It's cost of maintenance. So software is doing a certain job.

12:43

* The job can have all been time. There could be new routes.

12:48

* You are in marketing. There are new campaign. There is something changing the way the code was originally intended to.

12:52

* So cost of development, the costs of testing, and then once the product has been or the solution has been deployed.

13:02

* The costs of maintenance will stop.

13:13

* When you design software, there are different ways to do it.

13:19

* The. Typically when you develop something, you do your requirements, you do you design the solution,

13:25

* you implement the design, and then you pass the solution to the user and then the maintenance will stop.

13:36

* This approach is normally called the waterfall model was the base for most of the so-called legacy systems for quite a few time.

13:46

* Let's say for a good 15 years. Advantages and disadvantages.

14:01

* So advantages are quite obvious. You can define the requirements in depth.

14:08

* You can design the system based on the requirements in the most appropriate way.

14:17

* You have to allocate the resources in an optimal way.

14:23

* That's all in theory. But then the drawback is that there is no flexibility for change.

14:27

* Meaning all the beautiful plane that you did may not work because the condition changed, because you didn't really understand the flight well,

14:35

* what the requirements where for any reason the lack of flexibility is the main reason why the vast majority of the projects didn't went well.

14:48

* Using this approach 1995. That was probably the peak of this waterfall approach.

15:02

* Only 16% could be called the successful.

15:10

* It was obvious at that point that this approach was not the right approach.

15:16

* So that's why people started thinking in terms of software engineering,

15:21

* so how we can apply methods, approaches coming from engineering to the development of software.

15:27

* Again. We will talk with more details about that.

15:36

* But. Software engineering is a pretty much the system engineering applied to software development.

15:40

* So is the same holistic approach in the same thinking that there is no system living in a vacuum.

15:52

* But you need to consider all the other components that can be related to that.

16:01

* One of the point of reference in software engineering is the Software Engineering Research Center that is

16:10

* headquartered in the Software Engineering Institute that is headquartered in Carnegie Mellon University.

16:27

* And they develop the several stages of their methodologies that has been applied for quite long time in developing software.

16:39

* I was a partner member of the Software Engineering Institute for a few years.

16:51

* Some basic principles when you develop software.

17:00

* News. Open source. As much as possible, it is cost effective.

17:05

* There are drawbacks that there is no one really standing behind the product because it is open source.

17:10

* But on the other hand, it has the flexibility that you may need in many cases.

17:18

* Consider you have something that is not working the way you want on the software.

17:23

* If it is a commercial software, the only way to do it is submit a ticket and wait for the issue to be addressed.

17:30

* It can be in what it is to release or can never be incorporated in a future release because it is considered not strategic by the developer.

17:40

* So with open source, you can change the code yourself to make it like you want.

17:54

* Use the industry standard. There are two standards for many of the things that we are doing from interacting with the web,

18:03

* interacting with servers, exchanging file suites, but use that instead of reinventing the wheel.

18:13

* Third point make the graphical user interface.

18:21

* In general, the user interface separated from the backend system, the system with the logic.

18:26

* So they have different lifestyles.

18:35

* So the graphic, the user interface can change because you have a new platform,

18:39

* you have new system, you have a new user, but the backend may stay the same or vice versa.

18:44

* You can change the criteria where you develop, you provide the answers,

18:54

* but the format will be the same and you are not going to change the user interface.

19:02

* So generally speaking, keep those two aspects separated.

19:07

* So those are sort of common sense on developing software.

19:13

* Question so far. All right.

19:21

* So let's keep moving. And let's talk about Python.

19:29

* So some of the things are well known because we already introduce them last class.

19:36

* But I just want to be sure that they are just clearer and well fixed in your memories.

19:46

* So like many other languages, like all the languages, Biden has variable operators functions.

19:56

* We will talk about functions later on conditionals, cooperators, loops and all kind of operations.

20:04

* Um, there are some functions that are built in.

20:14

* There are some types of variables that are built in.

20:21

* So you have a, in terms of types of numeric integer floating, you have strings, lists to build.

20:25

* So we will go into the details of all of those.

20:33

* You may have dictionaries. So all of those are different types for, let's say, basic different types for Python.

20:37

* You can assign, as we know, values to variables.

20:52

* In this case is a I mean that is we are in the strengths area.

20:59

* So in this case is a list.

21:10

* You can have numbers, lists.

21:13

* We will talk about lists shortly, but a list is basically a collection of elements in a square brackets where each element is separated by comma.

21:16

* And the elements can be pretty much anything, can be numbers, can be strings, can be other lists.

21:30

* And you address the single element by calling them by number.

21:40

* And Python is starting from zero. So X is the element zero.

21:48

* Coffee is the element two. Strings can be both strings, and these can be changed, meaning you can't replace the values.

21:52

* We had some examples from the strings. Naming variables, we mention that there are some words that you cannot use to name variables.

22:08

* Those words are a result of the words. So you have words like and the F or from.

22:23

* So all all of those are or into are reserved words.

22:31

* You cannot call a variable a print.

22:35

* You can call a print with a capital P because up in in python has more letters and capital letters are considered a definite.

22:39

* Again, names are a case sensitive. So you can use any one of those with capital letters and there will be no conflict.

22:51

* But you want to avoid that because it could be confusing at the very end.

23:00

* You also want to have a name. So. That we will remind you what is the function that the body of Ebola is having in your program?

23:06

* Those are called the mnemonic variables, names, variable names, some meaning the names that are remaining.

23:19

* Again, the the role of that variable within the script.

23:31

* Comparison. You have the usual. There is not much difference from pretty much any other language,

23:39

* probably apart from two one that is that equal to the equal in the comparison is a two equal signs.

23:47

* So we use one equals sign to assign by the user to by their both.

23:58

* So we use a two equal signs for the comparison, all for the conditional statements.

24:03

* Another one that may be different, that is not equal to that is exclamation marker equal.

24:13

* In some other languages you has, you have a combination of less than a greater than one after the other.

24:19

* But I think in some versions of Python that works as well.

24:27

* But what is most commonly used is a exclamation mark equal.

24:32

* In computational theory, there is a theorem that is called Yeah,

24:42

* that is named after the creator of the theorem that is being been theorem saying that every

24:46

* logical operation or process can be represented with the combination of three elements sequences,

24:55

* conditionals and loops, meaning a language to be able to solve any logical problem needs to have those three components in it.

25:06

* So in Python, sequentially, it starts from the top, going to the bottom, from the left, going to the right.

25:18

* So in this case. If you have a program where X is equal to X is equal to x, you will get two.

25:27

* Then you do x equals x plus two, x will be four.

25:36

* So that's the sequence. Conditional.

25:41

* We mention that there are operators for comparison.

25:46

* Those operators are used, among other things, for the conditional.

25:52

* So conditional are with the if or if.

25:58

* So if today equal so double equal sign the October 30th pre into happy birthday John elif means else if.

26:03

* Meaning if this is not not true, then I am asking is today June 21st then that if yes, print Happy birthday, Loreen.

26:15

* And if not the meaning this is these fail that these fail that you are here else meaning for anything else you print.

26:26

* Good morning. Keep in mind that two things.

26:36

* One, you need to have a call on that at the end of the conditional statement.

26:40

* The second that you need to have. The true case, meaning when the conditional statement is through the condition, the condition, the statement,

26:46

* the meaning of the statement that will be executed when the condition is true, that has to be indented.

27:00

* So after a call on that, you always have an indentation.

27:11

* That's one of the most common error at the beginning of coding in Python.

27:17

* So on the left, you have a graphical representation, a flowchart for.

27:24

* An example of conditionals. So you have x equals five in the room by the you have the FS.

27:32

* So I'm asking if X is less than ten.

27:40

* If yes. And you're putting smaller then and you go here then I'm meaning this is the case for her.

27:44

* Yes. But if is no will go here anyway.

27:54

* So if is yes will print smaller if is greater than 20.

28:01

* And it would be bigger bigger if is not the either one will bring finish.

28:06

* So again, double equal call on the indentation.

28:15

* So those are three things that you need to keep in mind.

28:20

* When you do conditionals.

28:26

* We already mentioned that loops so loops meaning repaired repeating statements so that can be one single statement or a bunch of them.

28:31

* So in this case I initialize the variable n with the value five.

28:47

* Then I'm asking is and greater than zero.

28:54

* If yes, it will print an and will subtract one from n and we'll continue printing n fill.

28:59

* It will become negative and meaning not greater than zero.

29:07

* And then at that point that will go here and will print the finish.

29:11

* Couple of things. So one. We have several ways to represent loops in Pi Donut.

29:17

* So during the first assignment you experience why that was a loop.

29:25

* The other option that can be why conditioner through then you would exit the

29:32

* if is not true anymore order for for in this case you have a list of names.

29:40

* For anybody, any name or variable to say name in the list of names.

29:50

* Print the name. So the first round, the variable name will have the value of the first element in the list.

29:56

* That is frank. So the first round will print Frank.

30:07

* Then the second round it will point to. The second one again is a loop memory and so on till the end.

30:11

* What is really important when you do a loop?

30:20

* You want to be sure that there is no indefinite or infinite loop.

30:24

* So you need to have, if I remove this one and this statement, an equal.

30:29

* And minus one, he will stay in the loop forever, so it will burn your CPU.

30:37

* So be sure that you have a condition that will let the program to leave the.

30:46

* This is another example of a new person in this case that is using Whyalla.

31:00

* So while countdown at least a hundred. Bring the count down and then you decrease one element from that I that countdown.

31:09

* So initially, countdown is 100. That is not I mean, initially is at 25.

31:22

* So you are here. Yeah. Less than 100.

31:33

* Yes. We printed will be 24 and so on I think would be one.

31:37

* And then it will stop. So again, call on indentation.

31:43

* So the indentation is basically creating a sort of logical block.

31:52

* So you can have as many statements here.

31:56

* All right. So. You can use in the looper two statements that are particularly useful.

32:10

* You can use breaker. We use that in exercise zero and one.

32:25

* So in this case, what we are doing, we are generating a bunch of numbers and of numbers.

32:31

* So then we will print the numbers in a loop and if the number is five will break.

32:37

* So it will start with the first number, that would be zero.

32:46

* So range then is generating a list of numbers from 0 to 10.

32:49

* So when the number is equal five, it will break.

32:55

* Breaking mean means getting out of the loop.

33:00

* In this case. It's another statement that is continue.

33:05

* So I have a range of numbers from one to 1 to 1 with step one.

33:10

* I mean, you don't need to memorize all the all of those.

33:19

* You will use them. And you will remember then that you start the loop.

33:23

* Over here. I'm saying if the remainder of the number divided by ten is different from zero, then continue.

33:31

* Meaning don't do anything. That means that I'm looking for numbers that are multiple of ten.

33:42

* So because the number as a reminder at zero if is a multiple of that number.

33:52

* So in this case, it's ten when I have the first nine numbers.

34:00

* I will just continue doing nothing. When the number will become a ten, the remainder will be zero zero.

34:07

* And then I will print the number and then the string is a multiple and then we'll continue until the I mean, the value of 100.

34:16

* That example. So.

34:31

* In this case, a list of elements.

34:38

* And they're that names are the same names that we saw before the looper in this case and saying if a name start with M,

34:44

* then you will a printed meaning in the loop.

34:53

* It will be Mary and Mohammad, this one age 20, while age less than 66.

34:56

* You add one to age.

35:04

* Then when you go out of death, meaning when you are not less than 66, then you will bring the age and retirement.

35:08

* When we work with Python or similar languages, you can easily run all the code you want directly in the Python console.

35:26

* But then you don't have a trace of what happened.

35:39

* And when you closed your editorial, your computer, those variables, I mean,

35:43

* the next section, next session or Python, the previous ones would be deleted.

35:50

* So that's why we created we added the statements in one single file.

35:57

* In general, use the dot pie as an extension.

36:07

* Those files that we run executed by the Python interpreter.

36:14

* So what's wrong in this? So in this case, you don't have the indentation here.

36:21

* So that's what is missing in this other case.

36:27

* And you have. Numbers.

36:33

* You have a knife. But unfortunately the if there's one single equal sign, none too.

36:38

* So over here there was the missing the indentation over here, the error is the missing second equal sign.

36:47

* And then you have all the links for.

37:01

* I mean, more information on the on the Python website Internet.

37:06

* So let me stop sharing. And that is he.

37:15

* If you have questions, those links will have the commands of staff for reference.

37:21

* Yes. But I mean, generally speaking.

37:31

* Quite honestly, I rarely go directly into the python.

37:38

* Her website obviously is the only place with official information about what you are doing,

37:50

* but there are different ways to get the information that you may need.

37:59

* So let me share the screen in a second. For example, if I do.

38:05

* I don't know, transformer.

38:15

* Right on the list.

38:23

* Two strings. I mean, there are generally million different links you have on a campus building to StackOverflow.

38:30

* That's generally the way to go.

38:49

* So if you go to StackOverflow, you are going to have all the information you need to check the popularity of the question and the answer,

38:53

* just to have an idea on how reliable the answer can be.

39:08

* So in this case, I mean, that's a very basic but common question and that's why you have more than 1000 people

39:13

* upvoted the question and this answer with this in statement is super popular.

39:25

* So I mean, you can go to Python and check exactly what the manual is saying, but the StackOverflow is probably the easiest way to go.

39:37

* I personally I mean, I called for million reasons.

39:52

* So just to give you an idea.

39:58

* One of the things that they're doing a. Is managing the scheduling of courses and classes within the School of Systems and Enterprises.

40:02

* Now we hired someone who will be just on this one, but for about two years I did it by myself, but I'm still involved.

40:14

* One of the issues that we have in comments is, I mean, like all the companies that the data are not very well-structured.

40:31

* So when you need that information, that piece of information can be all over the places.

40:46

* So that's why I created this script.

40:52

* The. Just to give you an idea of how I mean, in daily life you can use Python.

40:58

* So in this case, a download from the system that we use as a backend in a commencé is called what they.

41:08

* The list of all the courses that are happening in a given semester are at sea events.

41:18

* And then.

41:25

* I basically clean the file because it's a giant file with all the courses for all the schools, but they need only the courses that are in my programs.

41:28

* Im IAC as is w and s way is.

41:40

* So I clean the data, I add then some features.

41:45

* What is the program? What is the level ground with undergraduate, corporate and information like like that.

41:53

* So when they do something like that, I mean, I'm familiar with Python.

42:07

* So I have I mean, I wrote I don't know how many probably by the thousand programs,

42:13

* but there is always something that you don't remember but you're not sure.

42:25

* And we tend to be lazy and I'm not proud of that.

42:28

* But these a matter of facts. And instead of going back in my memories and trying to find a solution, I Google it.

42:33

* So most of the time I end up in StackOverflow and I use it.

42:42

* So now I'm using also her chart, the GP.

42:49

* I'm pretty sure you're familiar with that. The. It is a very interesting tool.

42:54

* So it has been trained using pretty much everything that is available in open source, including the content on GitHub.

43:02

* So GitHub is a huge repository of code where you can have your own repository with your own code.

43:16

* Generally, if you have a space and inside this space you have your different projects.

43:30

* GitHub is owned by Microsoft, but Microsoft is one of the largest investors in open.

43:38

* And yeah, that is the company that created a chart to cut it.

43:44

* Besides using, I mean, everything that you can get from the Internet.

43:52

* I don't know if this is true, but this is what they said.

43:59

* It has also access to everything that is publicly available in GitHub, meaning it can really give you a way to do the coding.

44:05

* I'm not particularly happy with the result.

44:18

* So most of the time is missing.

44:21

* Something is using pseudocode instead of code.

44:25

* But you think it is called the you try to run. It is not working but can be helpful.

44:31

* I am writing a book on the societal implication of AA and machine learning and I'm actually using it as a sort of sidekick.

44:38

* So when I have an idea instead of google it, I use the bot and will give me a sort of a compiled version of what is available in open source.

44:52

* But in in education, we are all kind of scared of how students can use that because we all know that there is always a risk of plagiarism with that.

45:09

* These tools or tools like that, the the risk of plagiarism is even higher.

45:25

* We are still developing tools, able to detect what is coming from Jupiter and what is now her that again, you may use it to cheat her.

45:33

* I don't know if this is beneficial for your education.

45:48

* Most likely not much. So I encourage you to play with it because somehow we'll be part of our lives.

45:53

* But before copying and pasting things, switch your brain on and read what is inside and be sure that it is doing it in the proper way.

46:04

* We will run a chat group on some of the questions to compare the results, and we will apply the same concept that we apply for the copying in general.

46:18

* So if you are using chat, the GP and someone else is using the same.

46:34

* Probably your code will be similar, if not identical.

46:40

* At that point you will be penalized not for using GP, but because someone else did the same and you had the same code.

46:45

* So it is like doing after the fact because we cannot really intercept the cheating at the very beginning.

46:55

* So there is no way. In theory, we could create a classifier.

47:05

* So coding is very personal. So. We could create a classifier analyzing the code and detecting a sort of a footprint.

47:12

* So if there is the footprint of GPT three, then it's cheating, but the tool is not ready.

47:24

* Student University of Princeton created a sort of prototype that are the traditional tools 13 that is the most commonly used.

47:32

* The thing that is the most commonly used to detect plagiarism is developing a module on a chart.

47:44

* Gupta. But I mean that this tool has been released by the end of November.

47:54

* There is nothing that is ready yet. So but again, I really wanted to share with you very openly with what what is available if you need help.

48:02

* Apart from obviously myself and our work to StackOverflow and Google that are always good again, don't feel that you are doing something wrong again.

48:14

* I was coding since a while, but when I have even a trivial question so that I don't remember right away,

48:29

* instead of wasting my time, I googled it and they get the results.

48:37

* And then now we have a chart, the JPT, that can be an additional help.

48:42

* So that's the long story, probably longer than you want, but that's the full story.

48:48

* So is a seven, the 17?

48:57

* What we are going to do now will be an in-class exercise.

49:04

* So let me share the screen. And let me go.

49:10

* Be here. Okay.

49:19

* So I will publish it in a moment in in canvas.

49:23

* But I want to introduce it to you first.

49:31

* So you want to write a program that will you will call like changed the UI, that kind of a mimic, a conversion from U.S. dollars to another currency.

49:35

* So you will ask the user how many U.S. dollars you want to change.

49:46

* Let's say the input is hundred. Enter the name of the currency you want to convert the dollar into, let's say is yen.

49:53

* What is the exchange rate? Let's say the user is inputting 114 and the output will be you can exchange the number that you got from the user.

50:02

* It's 800 U.S. dollars for the multiplication are between hundred and ¥814.

50:16

* So you will use a loop that is pretty much similar to what you did the previous exercises.

50:26

* But to ask for the different input.

50:34

* And then you will check if the value that has been provided by the user is a number of now and you will use is digit as a.

50:39

* In a function that is associated to this to the string.

50:56

* So whatever is the name daughter is digit the open and close brackets Diawara

51:01

* that is digit the is pretty dumb and is considered number but only if there.

51:10

* Real numbers, meaning no minus sign, no decimal point, no comma.

51:19

* If you have one of those characters, it will not recognize it as a number.

51:26

* So in this exercise, that is obviously not.

51:34

* I mean that defeating the purpose of creating a currency conversion tool,

51:40

* you will consider only integers for the two numbers that you receive from the user.

51:47

* And then you will paint a blank line, whatever is the way you want to paint the blank liner that can be just printing nothing.

51:56

* Or you can use the special character a backslash and telling her new line The meaning is keeping a line.

52:05

* And then you will print her, you can exchange or whatever is the number of dollars for whatever is the result, whatever currency you are using.

52:12

* So let me make sure that.

52:24

* The. Okay.

52:31

* So the assignment has been published. The. What I will do now is to create some breakout rooms for.

52:41

* So I am creating for breakout rooms and you will be assigned automatically to those rooms.

52:59

* You will have about 20 minutes to do it.

53:08

* I will pause the recording for those 20 minutes and I will be here to whatever question you may have.

53:14

* And then after the 20 minutes, I will give you a warning and I will close the room and we will discuss the results.

53:22

* So you will have the opportunity to present what you did.

53:33

* If you want, I will present the possible solutions.

53:38

* And then we will talk about the assignment for next week.

53:43

* Okay. I open the room. I'm okay.

53:48

* So I am resuming recording. So we'll come back.

53:53

* Any volunteer just to share what you did.

54:00

* There is no judgment. There is no nothing just to share.

54:06

* Now I can share, Professor. So please. Yeah, sure.

54:11

* Okay. Let me know when you can see the screen.

54:14

* It's coming up. Yep. Here we go. Okay, let me just stay on somehow.

54:23

* Yeah. I don't know what the lines are for, but when we did is a while loop and we first asked the user for input in USD

54:30

* and we used the instruction from the assignment as to check if USD is digit.

54:41

* If it is a digit, then we would then proceed to ask the user for currency on the exchange rate and then we re proceed to cast both a USD,

54:48

* an exchange rate to a integer and then we have a variable call result.

54:57

* We did the calculation of exchange rate times USD and then we print the statement your exchange rate is USD US dollars for results.

55:02

* You can see the output below. We entered 100 for the currency name.

55:10

* We entered apples and for the exchange rate we entered 77.

55:14

* So now the output is then 100 USD 47700 apples.

55:18

* We also then proceeded to enter a random characters and continue the loop again and again with enter US dollars because it was not an integer.

55:23

* So good, good, good. I mean, the only thing that I can say is there is no way to be out of the loop.

55:36

* So if you remember except size zero, we had done with break.

55:46

* So in this case you don't have this option is that the assignment had the requirement to.

55:55

* No, no, no, no, no, but entirely. Oh, you're just making a statement.

56:05

* Okay. Yeah, I mean, it is working, but there is no way out.

56:08

* Yeah. So that's the only thing.

56:15

* It was not a requirement. Okay.

56:18

* But is generally something that you want to do because otherwise it will run forever.

56:22

* Yeah. Take it and call it out. Okay.

56:28

* Thank you. Anyone else who want to share?

56:31

* All right. So let me share the screen and we go to.

56:41

* Couple options. So does the basic one.

56:49

* No. That's the basic one. So the basic one is basic.

56:54

* I mean, just what you saw.

56:59

* So you have the input for the amount in U.S. dollars.

57:03

* If done, we break. Then checking if digit if not will print something and will continue.

57:09

* Then ask you for the currency conversion rate checking.

57:20

* Same thing if is digit and then calculating the value floating or integer in this case would be the same.

57:24

* And then printing. So if I run it.

57:34

* And I'd say 11. I have no idea what it is.

57:42

* And then done. When it's finished.

57:55

* I mean, it's all good. Nothing wrong with that. They only think, obviously, if we have a.

58:00

* Here. Whatever it is.

58:12

* If the exchange rate, let's say it's 1.4.

58:18

* I will give an error. So and that's basically the way is the Egypt works.

58:24

* So because the point was not consider a number meaning the old think is now the number.

58:31

* So another way that you have to intercept those airports is using tri except so the initial part is the same.

58:39

* So asking for the input, then the checking instead of using a digit and doing a tri except tri.

58:52

* It's basically a way to intercept errors.

59:02

* So meaning I'm asking by that to try to do this operation here.

59:07

* If there is no air ora will continue keeping the.

59:14

* Except if there is an error. Meaning.

59:17

* The amount is not dramatic other than that, except the statement that will kick in will bring to growing input and will continue going back and then.

59:22

* Yes, sorry, I have a question in Python version 3.9, the interpreter.

59:35

* Can you explain a difference between like accept with a colon versus accept with a explicit value like value error?

59:42

* Because it was asking for a specific exception,

59:50

* it was giving like a warning or something of that nature when you just leave accepted by itself without something more after the colon.

59:54

* Before the colon. Sure. I mean, I don't remember.

1:00:02

* What is the Python version that I'm using?

1:00:06

* Anyway. So I think I'm using nine. But that's not not the point.

1:00:13

* Um, um. Well, the try.

1:00:17

* Except that it can intercept the any kind of error.

1:00:22

* Uh, you can either leave it open.

1:00:29

* So the method that you received that is a warning is not something that is blocking the execution.

1:00:34

* Um, you can, I mean, if you go online, you will see what are the code there for the different errors.

1:00:42

* So you can probably you can get from the help here.

1:00:52

* Yeah. Yeah. So you can say, okay, try this one, accept this particular error and then do something.

1:00:58

* You can also have more than one exception, meaning you can really create a logic in there.

1:01:07

* So in this case, you can say, I don't know.

1:01:14

* I mean floating the idea or that you can get it if is not the number.

1:01:20

* But when you have statements that are more complex, you can have more than one number.

1:01:24

* I think you want to retrieve value from a database.

1:01:30

* So at that point, the database may not be available, that the file may not exist in the written code.

1:01:36

* It can be wrong. So each one will have a different the error code.

1:01:44

* So you can have different messages for different efforts?

1:01:49

* Yeah. Go ahead. Yes. Sorry.

1:01:53

* Would you say that it's best practice to be explicit about the exception and provide a specific where you want to accept or.

1:01:54

* No, no, no, no, no. I mean, it depends. If you don't have one particular reason for intercepting one particular errata, you don't need to do that.

1:02:02

* Keeping in mind that you can have, I think a use that.

1:02:16

* Here. So you can have a library.

1:02:24

* I mean, we are not there yet, but with Python, you can add functions.

1:02:28

* So in this case, I'm out doing libraries, doing special things.

1:02:33

* This function here is ignoring the warnings because sometimes can be a mean in this case and

1:02:40

* adding it because one library is not fully compatible with another library generating errors.

1:02:52

* So because I know that there is an error, but there is nothing that I can do because it's just a matter of compatibility between libraries.

1:02:59

* I just use the function that is bypassing the warning messages.

1:03:07

* So if the warning method is annoying, you can remove it.

1:03:16

* But generally speaking, I really don't see the need to specify the error in the except unless you have a good reason.

1:03:22

* So in this case, quite honestly, the only error that could happen there.

1:03:35

* I mean, it could also be I misspelled the name of the variable and there could be the variable and not been assigned.

1:03:42

* So eventually those are two types of carrots and they can have two different accents.

1:03:55

* But I mean that you don't need to do that. Thank you, Professor.

1:04:02

* I hope it's not confusing, but that there is always more than one answer to most of the questions by the.

1:04:06

* Thank you. So. All right. So, I mean, in this case is doing a pretty much the same thing.

1:04:15

* So if I ran it that. I can have a, I don't know, $22.

1:04:22

* I want to convert in euros and let's say 0.97.

1:04:31

* So in this case, again, no error. So and then all the rest would be the same.

1:04:40

* So again, I will post both solutions on canvas.

1:04:47

* Generally speaking, there is the Gita, as are quite a lot of limitations.

1:04:55

* I mean, if there is a floating point number, it will never pass.

1:05:00

* So the try, except it's the way to go.

1:05:08

* All right. Okay. So let me now introduce the next six essays.

1:05:15

* So this exercise is, in a sense, easier to.

1:05:23

* Then the one you did in class is another conversion.

1:05:29

* So you want to convert a temperature in Celsius to a temperature in Fahrenheit.

1:05:36

* As you can imagine, being born in Europe.

1:05:44

* I am more used to censors than Fahrenheit and the conversion is something that either with a calculator or just mentally I do most of the time.

1:05:46

* So that's why I'm starting with Celsius to Fahrenheit instead of ISO.

1:06:01

* That's just to give you a background so the user will be prompt to input.

1:06:06

* Temperature in Celsius, for example, at 15.

1:06:15

* And then the system will do the calculation using the formula, the temperature in Celsius multiplied by 1.8 plus 32.

1:06:19

* I'm still like, sense your sermon.

1:06:34

* You may think why you are thinking in that sense. I like the idea of a freezing point being zero, so it's kind of easier to me.

1:06:38

* But anyway, so you ask for this input, you apply the formula and you will put it into the equivalent of this case.

1:06:47

* 15 degrees Celsius is at 60 Fahrenheit.

1:06:58

* So this is the formula, and you will print the result with one blank line.

1:07:02

* And that's basically it. So that's pretty much everything.

1:07:09

* Let me make sure that you have everything you need to.

1:07:17

* I am publishing the in-class.

1:07:23

* Exercises the to the by.

1:07:27

* Publishing the text of the assignment and publishing the slides that I use today.

1:07:33

* All right. So that's basically it.

1:07:47

* The questions.

1:07:51

* All right. So if you have any question down the road, either myself or you, we will be happy to help.

1:07:58

* If not. See you next week. Hey, Professor.

1:08:09

* Yeah. Yeah. Good. Module A modules and canvas is still messed up.

1:08:13

* Are you. Are you aware of that situation? Okay, which one?

1:08:21

* Well, when we look at assignments and everyone chime in if you can.

1:08:27

* It's showing with due dates is showing all of the assignments from.

1:08:32

* It looks like a previous term of last year. It shows us everything.

1:08:35

* Oh. Okay. I mean, in theory, they shouldn't be published.

1:08:40

* Okay. I really thank you for letting me know. Yeah. Okay.

1:08:47

* And also, in addition, I did send you two separate emails.

1:08:51

* Is your email address r c la forza at stevens idea?

1:08:55

* Yeah. Okay. I don't know if you had a chance to look at it or maybe you were getting the same types of emails, so you just corrected it.

1:08:59

* I did see the correction to the quiz, which you mentioned earlier.

1:09:05

* Oh, yeah, yeah, yeah, yeah, yeah. I mean, there was an error and a chunk.

1:09:09

* Yeah, yeah, yeah. Okay. Yeah, that's. That's basically all I had.

1:09:14

* All right. Yeah. So how could they? Okay. Thank you, you two.

1:09:19

* And thank you for letting me know. Yeah. Right.