   
Hello. I mean, I know that there are several issues.

1:28

 We are we are fixing it. Sorry.

1:36

 Thank you, sir. I can appreciate it. I'm sure. I don't know why.

1:43

 It shows me the homework that is. Like the one that's there, but not the one that's due.

1:47

 Okay. Okay. So there must be something in setting up the old courts that is not working fine.

1:55

 But we will fix it. Thank you. Sure.

2:05

 Thank you for your patience. And I'm sorry about the inconvenience.

2:08

 Okay. The 629. Let's give people another minute or so and then we will start.

2:19

 So in the meantime, any particular issue?

2:36

 I have a question about the previous homework.

2:46

 Yeah. So I had difficulty coming up with that to end the loop.

2:50

 So I was trying different things, but I couldn't get it to work.

2:58

 Maybe you could explain. Yeah. Do you want to share it?

3:03

 Because, I mean, I don't have it in front of me.

3:07

 If you have it. I have it on a different computer.

3:12

 Unfortunately so. All right.

3:16

 Uh, okay. Y you will do the class assignment.

3:20

 I will retrieve it, and we will discuss by the end of the class.

3:26

 Okay. All right. Okay.

3:30

 So it's 631. Today is February the seventh and this is 624.

3:34

 So. Let me start with sharing the screen.

3:44

 We are already recording. Let me share the screen. And let me goo here.

3:51

 So minimize this. All right.

4:00

 So. That was the previous week.

4:04

 So module two was a software engineering and some coding in Python with the in-class assignment and the exit site zero one.

4:11

 And let me go right away on.

4:26

 The code. So. So the assignment was relatively easy.

4:33

 So it was on. On converting a temperature in Celsius to a temperature up in Fahrenheit and.

4:43

 I use the same loop that we used in previous exercises.

5:03

 So the loop is a y looper, just as we did the previously input with the commenter.

5:08

 What is the temperature in Celsius that you want to convert the type done to exit?

5:19

 And then the name of the variable towards a celsius temperature.

5:25

 Just to have a mnemonic. Variable.

5:31

 Name of variable. If this value is done, then thanks for using the tool and that breaker would go out of the loop and the program will end.

5:34

 It wouldn't be the same not to have the print statement here, but outside of the loop.

5:46

 But again, it would be exactly the same.

5:54

 Then I'm checking if the number if what the user typed is a number, if it is a false meaning, is no the number.

5:57

 Then you go back to the beginning again.

6:11

 That's not the only way to do it. We know that there is also the possibility to do in different ways.

6:15

 We will go back and say again, just as a reminder, if is not numeric, continue meaning going back.

6:24

 If it is numerical. Then I calculate the temperature in Fahrenheit using the formula and print it.

6:32

 So let me go here for a second.

6:41

 So just as a reminder there, there are two ways to get to test.

6:46

 If a number if a variable is numeric or not, there are many ways, but that's one of the ways.

6:53

 Two of the ways so one is using is digital and one is using try.

7:01

 Except let me run this.

7:07

 So if you type a number, an integer is digit, that is working fine.

7:13

 If you type a floating point number like 11.22, then you would get the is now the number meaning that is digit.

7:20

 That is not recognizing the number with decimal values.

7:33

 The. Right except the is working in a different way.

7:42

 Same number 11.22 and then is number because in this case what it's doing is basically trying to do the conversion into floating.

7:47

 If we get an error, then it will print the number.

8:00

 Let me run it again. So if I do a Q.

8:07

 There's no number. If I do Q In the second case, it will be the same.

8:14

 So again, with a try, except that I'm able to test the also the floating point numbers.

8:20

 So numbers with decimals with the is number is digit.

8:30

 I can not. So in this case, I use that is digit, but is clearly not the best way to do it.

8:36

 But it works in particular if the number is integer otherwise without.

8:43

 So if I run it, if the number is 37.

8:52

 Then I have 98.6.

8:59

 Pretty much periodically. It is an approximation and that obviously done.

9:03

 What? Okay. So that's basically the.

9:12

 Uh, exercise number one.

9:21

 And let me share again.

9:25

 And let me go back here. So for today, what we are planning will be again revising, reviewing the assignment, in particular the coding part.

9:31

 Again, that was a relatively easy code, just I mean, that is the second assignment.

9:49

 So the complexity will grow, but we will grow gradually now with big steps.

9:59

 So we will talk about software testing.

10:09

 We will give some examples.

10:13

 We will talk about the errors, exceptions in Python.

10:18

 Then we will go back to Python and we will talk about some data structures.

10:23

 I will stop after that for a Q&A.

10:31

 Then we will start the in-class exercise.

10:35

 We will discuss the solution, and I will present the next assignment.

10:42

 So that's basically the menu for the day. So.

10:47

 At this point. Let me go to the power point and let me start with this one.

10:53

 I'm not going to spend much time on the test driven development.

11:03

 The main reason is because a true test driven development is something that

11:10

 you would do when you have a quite complex program so that you would write.

11:19

 But you need to be aware that generally speaking, the problem of testing a software is essential.

11:30

 So there is no software that comes out of the developer right away with no error.

11:38

 So you can assume that in a way you would in either there would be an error.

11:48

 The point is how to use, how you spot it, how you determine what the areas where the error is and how to fix it.

11:54

 So being able to start from the beginning with an approach that, that that can give you the possibility to test that all the branches,

12:07

 that the logical branches you have in your code is really essential.

12:20

 So you need to be sure that you define unit of testing per each one of the alternative sets that you have in your code.

12:27

 In theory, you should define the testing strategy before doing the code, meaning you define the logical structure of your code.

12:42

 That is pretty much a sort of a flowchart of the code.

12:56

 And then you write the code. So you define the structure, you define how you are going to test it.

13:01

 You develop the code, then you tested the and then you check.

13:09

 Most of the time this is not what is happening.

13:13

 Most of the time you do a sort of retrofitting.

13:18

 You already have the code and you want to be sure that you will test all the possible alternatives, all the possible branches that are in your code.

13:22

 Again, you want to be sure that testing is an essential part of your development.

13:39

 When you test the you basically test the four carrots.

13:51

 So you want to be sure that, one, there are no formal areas.

13:57

 And secondly, that there are no logical areas.

14:04

 So let me go briefly into. The types of that that you can get.

14:08

 So there are three main types of area.

14:16

 So one is a syntax error. So we know that in Python you have a.

14:20

 Collins at the end of the conditional statements like if at the end you are call on a while through your column.

14:30

 And you also know that after call on that you have an indentation.

14:40

 If you know that if you if equal something, the equal something is not a single equal sign number is a double.

14:46

 If you miss one of those, you will get a syntax ever.

14:58

 Then there are execution errors.

15:03

 Execution errors is when you do something that is not in line with what Python is allowing.

15:07

 Like you want, you are adding a string and a number and you will get an error.

15:17

 So you are breaking the the python rules.

15:27

 So again, execution error is when you have something that is against the python rules.

15:33

 Example, mixing data times. And then there are the more and is key difficult to detect the error out design add ons meaning the

15:42

 program is not leaving any error or any transfer but is not generating the results that you expect.

15:57

 So there is something wrong in the logic of the program.

16:09

 So those types of error are more difficult to detect because I mean,

16:13

 with the the previous to the interpreter Python will tell you there is a this particular type of Arora on the line whatever is the number of the line.

16:22

 So you go there, you check it, you fix it. When is the sign the point?

16:37

 Python is not telling you anything because from a python standpoint, everything is fine, but the program is not doing what it's supposed to do.

16:44

 So at that point, a problem testing is what is really relevant.

16:53

 You need to really understand that the logical structure of the program.

16:59

 You need to understand that what are the alternatives that the program should do.

17:06

 So let's go back here for a second. So in this case, for this program that you have, so you want to test the mean,

17:13

 one of the four think if the program will exit when the user is typing done.

17:25

 So the branch, if done, is something that you need to test to see if it's working and then if it's detecting the done,

17:34

 if it's doing what it's supposed to do when the user is typing done that is axing exiting the program, then the second is testing.

17:48

 If the program is testing for being numerical or the input.

18:02

 And there's the second test that you need to do so.

18:10

 And then if it's testing is generating the expected the message or result and there's the second test.

18:14

 And then the loop.

18:25

 So it's looping the proper way, meaning that once they have what they have, is it going back and asking me for another time pressure?

18:26

 So those are the questions that you need to ask to create the testing strategy that

18:37

 you need to have to be sure that you are testing all the branches in the code.

18:48

 One of the ways that we normally do to be sure that we are checking all the boxes is to put some print statements in the key steps of your process.

18:57

 So, like, I don't know.

19:19

 If I mean in these cases or in painting something.

19:26

 So in all the cases is printing something meaning is not necessary.

19:31

 But if I didn't have. I mean that. If the requirement was only when it's done, just exit.

19:37

 And adding a printer to say I'm passing through this branch is something that you want to happen.

19:46

 Printing may not be just saying I was here, but can also be the type of variable or the value of the variable.

19:55

 That is what I expected. So those kind of temporary pop up print statements are working for the debugging.

20:06

 Once you fix it, you may want either to delete them or to make them comments because you are not totally sure.

20:22

 Eventually you will change something and you may need to go back to the same point.

20:33

 The meaning you are a pop up print will be useful again.

20:38

 So at that point you comment the print, adding the number, sign the in front of it, and then once you complete the full debugging, you can remove it.

20:45

 So again, three types of errors syntax error, execution error.

20:59

 Those are easier wrote. They are easy because Python is telling you the type of error.

21:04

 And when the error occurred, the designer roles are more tricky because the interpreter python is not telling anything,

21:12

 but the program is not doing what is supposed to do.

21:25

 At that point, you really need to do a through debugging.

21:29

 Questions so far. All right.

21:36

 So let's keep going. And let's go through some other python thinks that we may want to add.

21:44

 So question. Yeah, go ahead. Sorry, I didn't realize it was on hold.

21:59

 So I was looking through the exercise 33 to 36 or something like that from the reading assignment.

22:05

 And I was trying to do the percent deal. But the percent d and then the percent I with our homework.

22:13

 But it didn't work out. Is it because it's not an array by any chance?

22:23

 Okay. Can you share them? Because I really don't remember.

22:28

 I can tell you I can maybe share the book, so.

22:34

 Sure. Because I deleted it because it didn't work.

22:39

 Okay. Okay. Okay. Okay. Yeah. I just share what the questions were and then we will review them.

22:44

 Okay. I think it's in the book exercise the the the hard way the prodding the hard way.

22:59

 I believe it's an exercise 32. So if you have it and you you if you can shed the screen.

23:07

 Otherwise, I will I will check it. I don't.

23:14

 Okay, that's fine, I. If. All right.

23:20

 If you can make it a little bit bigger would be great.

23:40

 Okay. I think it is.

23:51

 I'm trying to find it to believe it's here.

23:55

 Yes. Is this. I got percent. Ah, and this is percent I.

24:00

 Where is that integer or not integer. The, the value I guess.

24:05

 But I tried to do that with my print in the at the very end this says Celsius.

24:11

 I did the same thing. Let's say the answer is percent.

24:17

 I put D and then I did the same exact syntax here, but percent temperature C.

24:23

 But it didn't work for me and I wasn't sure why.

24:29

 Yeah. Oh, okay. So there are a couple of issues.

24:33

 There is one issue and one consideration on that print.

24:38

 The first issue is because the code that you have is for Python two with Python to you have no parentheses.

24:42

 So you need to have parentheses meaning is I mean the up to the the percentage sign and then I whatever it is just before that you need

24:53

 to close the parentheses so the public will start before the quotation sign and then will end after the last the last quotation sign.

25:09

 Gotcha. All right. I'll try to work on it when I have.

25:22

 Yeah. And then. Yeah, yeah. And then the second point, we generally don't use this way to do the printing with there are two ways of doing it.

25:26

 The one is print the parentheses F and then you have a the name of the variables in curly brackets.

25:40

 And the second is using the percentage sign and the corresponding value after the parentheses in the print.

25:50

 It's kind of confusing. I don't like it that much.

26:02

 I really like more a plain way of printing.

26:07

 If you stop sharing for a second, I go back to my code and yeah, and I will just use the code as an example.

26:12

 But thank you for raising this point and that's a very good point.

26:24

 So. If you consider this print right here.

26:29

 So in this case, apart from the new line, I have a comment.

26:37

 I have but one value. I have another comment or a stronger.

26:42

 And then another volume. I mean, this way, in my opinion, is very clear and straightforward.

26:49

 If I start using a variable, I mean, it is more parametric, but it's less readable.

26:56

 So I strongly encourage you, I mean, again, that the course is not on the esthetic of coding but is more on maybe functionally work.

27:04

 So I really don't care how you do the printing, but in my opinion that this way of doing the printing is more readable.

27:19

 So in Python three, don't forget the parentheses at the beginning of the printing and the end,

27:29

 and then inside that whatever you print as to be separated by comma.

27:38

 So you have this first portion that is a string and the fix.

27:43

 The string is not a variable. Then you have a comma, a variable you can or cannot have.

27:47

 A space is up to you again to use this basis because it is more readable.

27:53

 And then you can have as many variables and strings in your print you do not want

27:59

 to do thinks of that too long because otherwise it will become not readable.

28:07

 But I'm in my opinion that this way is more readable.

28:12

 What do you think we are now? Yeah, I did. I did exactly what you had.

28:18

 I just wasn't sure. Yeah, y why you didn't work, but.

28:22

 Okay, that makes sense. Yeah. All right. Okay, so let me go back to sharing and let me go here.

28:25

 So we are going to talk about some other Python related things.

28:37

 We will go we will review the variable types and how to move across types.

28:45

 So with Python, you have several types of variables.

28:53

 So we mention strings numbers.

28:59

 So strings are defined by quotation science and can be single or double.

29:03

 For Python, that is just the same you want to say with either or, but mixing and matching doesn't work.

29:13

 Recently I use single up, but I mean really the same numbers can be inside.

29:23

 That can be number. It can be a.

29:32

 Alphabetical characters or whatever you want.

29:39

 Numbers can be integers or a floating point, but.

29:43

 You can do operations between a body of both.

29:51

 So on the left side, the you have strings.

29:57

 On the right side, you have numbers.

30:01

 You can add the strings, variables containing strings like in this case, when you add the they will combined attach the one to the other.

30:04

 Obviously if they are numbers, you do the arithmetic operation with that same thing with multiplication you can apply the multiplication

30:17

 to a strings and it will be the same string repeated the time each time you attach one to the other.

30:29

 When you do with number two, you have the arithmetic operation.

30:41

 Again, there is no mix and match.

30:49

 Excuse me, because otherwise you will get an execution ever.

30:56

 So, like, in this case, you have an error.

31:02

 The good pointer is that python that will tell you the line that when the error record.

31:06

 You can ask by doing the type of variable that you are working on that seems to be irrelevant.

31:15

 But sometimes when you do debugging, when you when you are doing testing to a code that you do an operation, you get an error.

31:22

 The program is pretty long. You don't know why you have a wrong combination of variables because you expect the one

31:35

 variable to be numerical and you are adding a number to that variable and you got an error.

31:46

 Why? This variable is not numerical.

31:52

 You can go back. And add some print statements to the code, the printing, the type or the variable before you do the operation.

31:56

 Just to be sure that you trace the evolution of the values of the variable while you are creating that.

32:12

 So Typer is useful. So if in this case is a stringer,

32:24

 you do type-A for the value for the I mean the variable and you will get the in a minor bit less than greater than the characters type.

32:30

 And then the type of variable that that particular string is the particular.

32:49

 But for this same thing, if it was an integer, you would get the list floating, things like that.

32:58

 So again, type is something I spending more time than I should, but it is important.

33:09

 We have integers, floating points. We know that you can convert one to the other if the conversion makes sense.

33:20

 We know that if you try to do a conversion of a string that has not numbers in it, but into integer or floating it, you will get an error.

33:30

 Those are other examples. So you have.

33:47

 Operation said that out on the algebraic outside the operations that are.

33:54

 So you are transforming content and then you doing an arithmetical operation.

34:02

 In this case you try, as we know, to do a conversion of a string into a number,

34:08

 but the content of the variable, it's not a number and you will get an error.

34:17

 Again. That's another example. We know how to do the management of errors in this case,

34:27

 keeping in mind that when you do try accept that you can specify one particular

34:43

 type of error and have eventually different messages for different types of air.

34:49

 If you check online in the pilot on the website or in StackOverflow, you will get the codes for the different type so that there are several.

34:57

 User input. We know how to do it.

35:13

 We know that you can convert it if it's convertible.

35:17

 We mention that you can do operations with strings.

35:25

 Now let's talk about lists. So lists are another formula, another type of variable, not that are a container, so they are defined by square brackets.

35:30

 So there is a square back at the beginning, a one at the end, and then you have elements inside of those elements a.

35:47

 Are separated by commas. And they can be anything they can.

35:56

 They can be any type of variable. They can be numbers.

36:03

 They can be strings. They can be other lists.

36:07

 Meaning you can have a list that is nested into another list.

36:13

 We will talk about that. Uh.

36:17

 Lisa, our new double meaning.

36:22

 You can change the content. You can change the order.

36:25

 You can add elements. So let's say that you have this list here.

36:29

 Like. Pretty much everything in Python.

36:36

 We start from zero up to the end. So the first element is the element zero.

36:41

 So if you want to recall the first element, then you need to point to that element.

36:47

 So that's the list that we created. Again, the square brackets, commas, separating the elements and any type of element, any type of variable inside.

36:56

 If you in that, I mean, that is sort of pseudo code that the brackets are missing here.

37:09

 So example at zero, meaning you are pointing to one, two, three and you would get it.

37:18

 The example is three.

37:24

 You are pointing to the fourth element that is fish. You can change an element.

37:27

 So in this case and saying example is that three equal tilapia.

37:32

 So I changing from fish to tilapia. When I printed that, instead of getting a fish, I would get the new body.

37:39

 And one more feature that is really useful when you deal with the ball with lists or strings is a slicing them so.

37:50

 We mention that you can point to one particular element.

38:04

 So if you are saying this leads to a2h3 meaning I'm pointing to the fourth element that is a the I can

38:08

 do three column that meaning is from the fourth element on and you have all this piece of the list so.

38:18

 I can do up to that point the meaning all.

38:30

 But the fourth element that before the fourth amendment, then you will get A, B, C, you can slice a piece of it.

38:35

 So you are saying from three to 3 to 5.

38:44

 So you have from the fourth element to the fifth element.

38:49

 I mean, from the fourth element to the sixth element, the excluded.

38:54

 So you have the any. You can do backwards.

39:01

 So you can do minus one, meaning the last one you can do minus to call on meaning the last two.

39:06

 So this lighting is something that you will practice and is pretty useful when you are dealing with managing text in particular.

39:19

 One thing that is relevant is the use of append and remove append.

39:35

 That means that you can add the elements to a list.

39:42

 So you have an existing list like what we used before, and you want to add the number 42 with append that will add the in the end.

39:47

 You can remove elements. So you just call the element you want to remove and you do name on the list remove.

40:01

 And in parentheses you have the value that you want to remove.

40:11

 One thing that is sometimes is generating a little bit of confusion is the difference between concatenation and appending.

40:17

 So you have the same lists. The first is a one, two, three, four, five, six.

40:27

 You do a concatenation, so you do C plus a C equal to A plus B when you print in the you will get the combination of the two lists.

40:34

 So you have a one, two, three, four, five, six. If you do append, you are upending an element to an existing list.

40:48

 So the same A and B, you do a append b.

40:58

 When you printed that you had the first three elements, and then another element that there is the entire list.

41:04

 Append is what is normally used.

41:11

 When you have a loop, you have a loop and you are creating you are generating a list, adding each iteration, an element.

41:14

 So at that point you are appending, just like we did here with 42, you are appending fortitude to the list.

41:24

 Think about a loop or instead of 42, you have an index and you have numbers from.

41:34

 A to Z, whatever R, E and Z in values.

41:41

 Each time you will add an element. At the end of the loop, you will have to create a list with all the elements.

41:47

 So append. Is what we normally use that when we are creating lists and we want to add at each adoration and element.

41:55

 Another useful thing to know is the existence of, in or not in.

42:10

 So in this case you have at least the but could be a string and you are asking if 42 is in the list and you get through.

42:17

 So in this case is not in the if. But if you have that in a if statement, then you can take different actions based on if it's true or if it's false.

42:27

 So if you ask 55 in a, you will get false.

42:40

 You can do not in that. And if you do 50 fi not in a then the result would be true.

42:45

 Lists are ordered, meaning the way you create them is the way they will stay.

42:54

 That seems to be obvious, but not all the data types in the variable types in python stay order.

43:01

 You can change the order because again, lists are mutable.

43:13

 So you can. Either add elements, move elements.

43:18

 You can delete elements. You can sort the elements.

43:25

 So all of those is doable. But if you do nothing of the three, the least that will keep the order that you used when you created it.

43:29

 They can be sorted again. You can sort them in ascending or descending order.

43:43

 If that is number, it is easy. If it is a string.

43:50

 It would be alphabetical order. Pretty much the same is for a string.

43:56

 So when you have three things you can do.

44:05

 Slicing. Just like we saw for lists.

44:09

 So five on meaning. I mean, instead of a I mean, from five on you skip the first five, but then you do the remaining.

44:13

 You can pick the fourth element that could be a D.

44:28

 You can. Do a loop within a string, and then at that point the loop will be on the single character in the string,

44:34

 including space because space is a character for all intents and purposes.

44:45

 There are some methods that can be used with the strings upper lower to capitalize either.

44:51

 All of those are things that can be added to any string to make it a little bit different.

45:01

 So if you have this combination of a lower and cubby the letter certainly in the string, if you do lower, you get low words and so on.

45:11

 You can also do start with them. And that's something that sometimes may be useful.

45:23

 So you want to put into all the courses started with em so you can have a list of courses,

45:28

 you do a loop, and then in each loop you have one course and then you start that.

45:41

 If you ask it and you have and if statement the if the name of the variable start with M, then do something, count it and so on.

45:47

 Keeping in mind that again is more lettering.

46:00

 Capital letters in Python are different.

46:04

 So if you test in this case a capital H e, you will get through.

46:08

 If you do a smaller H e, you will get folds.

46:15

 There are about 33 methods, and you can check them online.

46:20

 Uh. Again, that's another example that can be applied kind of a combination.

46:27

 So you are checking if one string is in.

46:35

 In the least. That's something that can be useful in many case.

46:40

 You have a list that you imported.

46:45

 You read from a file or whatever, and you want to see if a certain value is there or not.

46:50

 And then you do different things based on the different options before the results.

46:55

 Replacer again, you can replace elements to both lists and strings and mutable meaning.

47:01

 You can change them the way you want.

47:11

 You can do conversions with all the caveats that we mentioned many times when you convert a string into a list.

47:15

 The resulting list will be composed by the individual characters that can be letters, spaces, or any other character in the string.

47:29

 So in this case, when you are making a list out of the word cements, the resulting list would be, I mean, the individual letters in the name Stephens.

47:40

 That's something I want to draw your attention to.

48:01

 So this is a list where the first element is a number of the second is a string.

48:04

 The third element is a list, and the fourth is a number.

48:10

 So if you do, if you point to the third element, so it will be two.

48:14

 So if you do my list two, you will get this list if you want to get the.

48:22

 The second element of the NRA list.

48:29

 Then you have you need to have two indexes.

48:33

 So the first one will point to the inner list and then the second that will point to the element that within the list.

48:36

 So lists of lists are very common, and I strongly encourage you to become familiar with those.

48:48

 Conditional accepts. Again, we can have loops working on the lists.

49:00

 So you have a list of names and then you do a loop and each time you will print one.

49:08

 It is just an example of how to use lists in a.

49:16

 Loop. You can apply all the comparison operators.

49:22

 As we know, you can break out of loops using a break you can use continue for doing nothing.

49:29

 Meaning in this case, if the and the first element of line that would be the number sign continue meaning you go back,

49:40

 you do nothing if it's doesn't break. We know that in this case you are going out of the loop.

49:52

 Other application of loops.

50:03

 You have these values and you want to count to do the summation and then you want to bring the count to the summation and the specific part.

50:05

 So that's another example on how to use loops.

50:20

 And when you finish, when you are out of the loop, you can do a mean summation of the values divided by the number of values and you have the average.

50:26

 And then you have links to the.

50:42

 To the Python website when you can have more details on that.

50:49

 All right. It's 718 at this point.

50:58

 I would introduce if there is no question, I would introduce the in-class assignment and I will give you.

51:04

 About half an hour or 25 minutes for the assignment.

51:21

 Then we will, let's say 20 to 25 minutes.

51:26

 We would discuss the assignment and I will introduce the next exercise, and that would be the end of the class.

51:30

 And obviously, if you have questions, I will be happy to address it.

51:40

 Okay. So. Let me share the screen again.

51:47

 And let me. Go here.

51:55

 So they think class. Exercise as a user for a number depending on whether the number is even or the.

52:01

 You want to print an appropriate message. Meaning if the number is let's say two, then you will print.

52:10

 The number is odd. The number I'd say is a five.

52:18

 It's, it's, even if it is five, the, uh, the number is, uh, you want to print the,

52:22

 you are asked to print a different message if the input number is at four.

52:33

 So if it's for you, print something different.

52:40

 If it's not the it just odd or even.

52:44

 So let me stop sharing and let me create some breakout rooms.

52:50

 So I am creating three breakout rooms with three participants.

52:59

 Each room. I am opening it so I will knew them myself.

53:05

 I will be here all the time anyway. And.

53:13

 The rooms will stay open for about 20 minutes. Then I will close the rooms.

53:20

 I will send you a message before closing.

53:25

 And then that we will reconvene. We will discuss what we had.

53:30

 And then I would talk about the next class.

53:35

 The next assignment, see in about 20 minutes.

53:40

 Resume the recording. Oh, right.

53:55

 Welcome back. All right, so anyone wants to share what you did.

54:01

 Again, there is no grading. There is no judgment.

54:12

 It's just for sharing and sharing the comments.

54:15

 Your areas may be useful to someone else.

54:22

 So. You're strongly encouraged to share it.

54:25

 I can share, professor, during the volunteers? Sure, please.

54:32

 Okay. And so this one.

54:36

 We had two exercises. Is that correct, Professor? Well, it's one exercise.

54:47

 So, uh, let's see what you have.

54:55

 Okay, maybe I read something different because I was still seeing, like, all the assignments from, like, last year.

54:59

 Yeah. Yeah. So maybe I read the wrong one. But anyway, so again,

55:04

 this program prompted us to enter a number one or two to understand exercise one or exercise to exercise one will basically ask the user for a number.

55:07

 Write checks if it's hard, and then if it's not odd or check if it's even.

55:20

 Or multiple four. Sorry. If it's if it's multiple four and it's not zero, then it will print.

55:27

 You know, the this number is even but also multiple for otherwise in in the case the last case,

55:35

 if the user enters zero, it would enter, uh, you know, zero is even, right?

55:41

 So that's the case that's carried here. And in the second exercise with it prompts user for a name,

55:46

 you would enter your H and it would compute basically the year that you will reach 100 years old,

55:54

 including the days until that first day of that 100th year.

56:04

 And so that's what Exercise two does. So I'll start with Exercise one, right and true one and turn number on like let's do 2.5.

56:10

 And as you discussed, that's not going to work. So let's do 20.

56:21

 So now you get 20 is even and also a multiple of four.

56:27

 Right. Well, let's try to assess case this. We'll start over with this one and our number 200 is even and that's the last which case there.

56:31

 So that's for exercise one. Exercise two is an attorney, Kevin, until your age 30.

56:42

 And then it says in seven years you will be 100 years old and that's 25,530 days away.

56:50

 So that's the last part and how I did.

56:59

 The last part is I use StackOverflow, as you recommended, to obtain some information about how to obtain the current year.

57:01

 Years remaining is a variable where I subtract the user provided age 100 minus that age and

57:10

 then the year at 100 or the current year plus the years remaining and to calculate the days.

57:18

 So the first day of that 100th year i u is the date time that now to get the current day and for day two,

57:24

 I use that time of passing three arguments, including the year at that that year that I'm talking about a hundred years old.

57:35

 And I subtract that right and then I assess the days using different days.

57:43

 So that's how I print that result there. Sounds great.

57:50

 I do have a good question, professor. Good. Is it bad practice to have the numbers like this in a variable and by the.

57:54

 Well, I mean, the names or the variables can be pretty much anything.

58:04

 Apart from the risks of words and keeping in mind that they should be mnemonic.

58:11

 So here hundred it is mnemonic.

58:19

 It's absolutely legitimate. And I mean, I don't know why I would have added an underscore here on this quote hundred, but that's me.

58:23

 I mean, that is absolutely fine. The main goal is to have names that will remind you what the content could be.

58:36

 So in this case, that is just fine. Okay.

58:45

 Are there any obvious issues you see on these two exercises or.

58:49

 It looks fine. No, they looks fine.

58:54

 I mean, as a general consideration, I mean, we are doing exercises in a sort of a gradual way.

58:57

 Yeah, but a more basic way of doing it would be not using functions, but just right.

59:10

 What is the day today? What is the object in the code that would be less effective?

59:20

 Less sophisticated? The more basic.

59:29

 I'm not saying that is the way to do it.

59:33

 That wouldn't be the way of doing it.

59:36

 But it would be the way that I would expect at this stage of the class.

59:40

 And that the only consideration, I mean, the way you are doing it is the way you're doing it that you don't want to have data in the code.

59:47

 Generally speaking, you want to have the code, the doing only the functional part of the process and the data being outside.

59:57

 Yeah, but I mean that at that sort of an early stage of the skills getting process to where you are.

1:00:08

 I don't care much if you okay in the script, but generally speaking, that's the way of doing it.

1:00:24

 Another approach that I generally follow is to have, but we didn't do yet.

1:00:32

 Handing files is to have a parameter that's like the data.

1:00:42

 I mean, they wouldn't want the effort, but with any form of parameter in a configuration file,

1:00:48

 meaning you read the file with all the data, the data will be incorporated in the code and becoming values for variables.

1:00:56

 Meaning if you want to change the values, you don't need to change the program, but you just go into the file with the values and change the values.

1:01:08

 So we will go to that in a few classes.

1:01:19

 So far, I think his next class when you will do read the right files.

1:01:25

 So for the time being, what you did was just good. Okay.

1:01:32

 Thank you for feedback for anything. Thank you. Any other.

1:01:37

 Yeah. I can share. Please. So we didn't know about the second son who was not there.

1:01:43

 Okay. So for the first assignment, we took a pretty simple approach.

1:01:56

 So I have this variable number and then we pass that number to an eight and divided by two to determine its half number,

1:02:02

 which is this other variable that we made. And we used this half number that there was there.

1:02:14

 That's why I did work anyway. The fourth thing worked before.

1:02:25

 But, uh, so we said the, the actual part of it is down here with the we use integer.

1:02:30

 And we said if half number is integer print that it's an even number.

1:02:38

 And then we said else it's an odd number.

1:02:43

 And then we have just some other cases up here where we rule out the zero case and the four case that I requested so I can go ahead and run it.

1:02:45

 But so it's not the, the prettiest, uh, code.

1:03:01

 Okay, but. We?

1:03:06

 She were. I remember we just said all the cases or whatever.

1:03:15

 Yeah. I mean, it's working.

1:03:25

 That's absolutely fine. A couple of considerations.

1:03:29

 The first one. You do a loop.

1:03:34

 You use a way through if there is a loop.

1:03:38

 So in this case, the loop is not use the match.

1:03:42

 It could have been used better. Adding it down when you want to end it instead of breaking the loop.

1:03:54

 Each time you reach a conclusion.

1:04:07

 You may want to have it continue in the testing if done at the very beginning, like we did in the previous assignment.

1:04:11

 Yeah. Yeah. I mean, it is working again.

1:04:22

 I'm not teaching. This time of coding is more a functional approach than teaching.

1:04:27

 But keep in mind that that that could be a good addition.

1:04:34

 Yeah, sure. The second point is divided by two is working.

1:04:39

 But you can use the function reminder, meaning if the remainder of the deviation by two zero, then it's equal to.

1:04:50

 Yeah. I saw that in the previous. Yeah. And that's another approach.

1:05:02

 But I mean both the options are working fine like divided by two because there's more narratively using the basic functions.

1:05:07

 And at the stage we are. If it's good to use the basic functions, then we will add a step by step, class by class, new elements.

1:05:20

 All right. Thank you. Okay.

1:05:33

 So you can stop sharing.

1:05:37

 And I will briefly introduce how we shared the screen and I would talk for a second about the qualities that.

1:05:41

 So in this case, certainly I'm doing the same way through.

1:05:58

 So I'm asking if. I mean, I ask him the number.

1:06:06

 Baking. If it's done. If it's done breaking, then I'm checking if it's numerical.

1:06:15

 If not, the meaning. If there will be an error here, will be intercepted by the accept the option will go back,

1:06:23

 ask for another one then and checking if is a multiple of four meaning the reminder from the division before it zero then is a multiple of four.

1:06:33

 And then if it's the remainder with the division of two, it zero is even.

1:06:45

 Otherwise is that all the and we go back.

1:06:53

 Abdul-Mahdi again, I could add that continue here, but it's redundant that we go back anyway.

1:06:57

 So if I ran it up. It will do?

1:07:03

 I don't know. 22. It's even 11.

1:07:09

 It's odd than that is out.

1:07:14

 That's very basic. So overall, I want to apologize for the kind of mess that is on campus.

1:07:19

 The main reason is because when I had the original content, there were a lot of parts missing.

1:07:31

 I uploaded the previous version the previous semester, but then instead of replacing, it kind of added to it.

1:07:40

 Meaning you may have multiple versions for the same thing and cleaning up.

1:07:52

 So be patient. My apologies from next class.

1:07:58

 This is not going to happen and doing all the cleanup.

1:08:04

 So let me go back to the assignments for next week.

1:08:09

 I strongly encourage you to pay attention to at least the beginning of the description of the assignment.

1:08:25

 So the assignment is two parts. One is on testing and one is on writing the code that they are related to the same problem.

1:08:35

 So the testing is not genetic testing, but is testing the code that you will write the in part two.

1:08:45

 So for the part one, you want to design testing.

1:08:55

 So to design the testing, you will use this template.

1:09:02

 So the template has the goal of the program.

1:09:08

 And the goal of the program is from the program as a loop and all the rest.

1:09:12

 So that's the goal of the program. Then you have the testing strategy, meaning what are the logical step so that the program will perform,

1:09:19

 for example, done to exit the program and printing the proper statement when the user enters,

1:09:30

 done checking if the price is a number, another string printing a proper statement when the user enter a price that is not multiple of five.

1:09:37

 So all the alternatives will be here in English.

1:09:49

 No, no code here, no python statements here.

1:09:53

 And then for each one of those you will write the data that the user would use to go in each one of the options.

1:09:57

 So if you are to print in the proper statement that when the user enters done, it will be input done.

1:10:09

 Output goodbye input at 12 output price not a multiple of $0.05 something that that.

1:10:18

 So again. Part one is reality two.

1:10:29

 Part two is not a theoretical question. So part two is basically write a program that will ask the user for a number and then a.

1:10:35

 The number has to be a multiple of five and that's the procedure.

1:10:50

 So you prom the user for the price in cents, meaning it has to be a multiple of $0.05.

1:10:58

 You need to check if the number if the input is numerical, if it's not negative,

1:11:06

 and if it's a multiple of five and you will provide different messages for the different roles, if the user will enter done, then goodbye and stop.

1:11:12

 If the user enter a price that is actually a multiple or a five, you will print the price and you will repeat the old think.

1:11:23

 You will submit the doc or PDF for the testing based on this document and in your programmer as a p y document in canvas.

1:11:36

 So I will review the submissions for the previous assignment that again, there was some confusion on your side, on our side.

1:11:50

 And I apologize for that. We will fix everything from next class and for the time being and just accept my apologies that.

1:12:04

 Okay. So. Questions.

1:12:19

 So, Thomas, I'm for a second. You have a question.