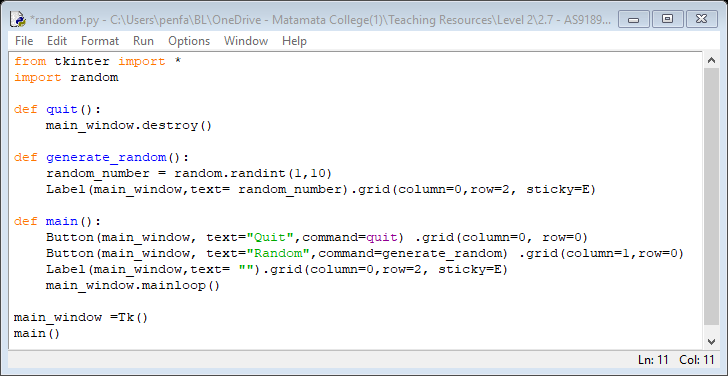
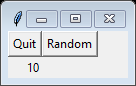
# time to move on

**Random functions**

* Hopefully by now you have the basics of creating a program in a GUI
* We are going to work on extending our programming skills by learning more functions
* The GUI will be used for all code we do, so it would be good to save a basic program
* This will let us add and modify it each time without starting from the beginning
* Most programs will need a quit button, one or more buttons to click on, plus input/output
* In other words, the program we currently have will be a good starting point

# Getting random

* The random function allows you to generate random numbers
* Did you notice that when we added row and column to grid it no longer kept adding to the bottom of the grid - instead it overwrote the same cell in the grid? This is generally a good thing
* We will be importing **random** in the same way we imported **math**

* The extra **Label** function has been added in **main()** - this is so the box exists from the start
* The generating of the random number is in the **generate\_random()** subroutine not in **main()** as this is the logical place for it to go. Always put as much code into its own subroutine

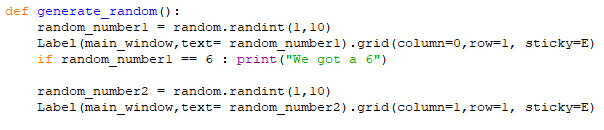
Task time

* Change the random numbers so they are 10 to 20
* There is an un-needed empty space next to the **Random** button. Remove it
* Have two correctly named buttons - one generates number 1-10, the other 10 to 20
* Have one button that generates the random numbers and puts each into its own cell

# the if statement

**Introducing IF**

* Making decisions is a basic part of programming - the first one to learn is **if**
* When you think about the word **if** it suggests two possibilities
* For example, **if** it is raining use an umbrella. This implies no rain, no umbrella
* There is no in-between with **if**. There is no ability to handle light rain. It is, or it is not
* Our first program will look to see if the first number randomly generated is a 6
* Note: only the changed part of the previous program is shown

## If explained

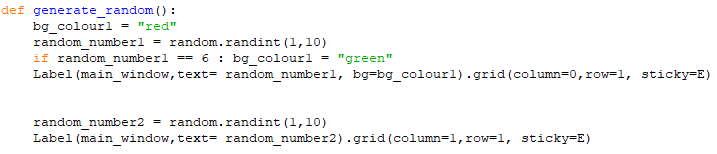
* If you are comparing two things double equals is used **==** so it is not confused with single equals **=**
* A colon **:** is used at the end of the if statement - **if** will get more complex so this is needed

Task time

* Change it so when the second number is a 6 it does a print
* Have a separate **if** statement for each random number
* Make each print statement unique so you know if it is the left or right one that is 6

# Putting it on the GUI

* The Label statement allows you to put in a colour - we will change it if we get a six



* You will notice it resets itself to red each time. That is because each time you click on the random button the subroutine is run again – including the line that sets the colour to red
* You will also notice you end up with red and green at the same time - **width** will fix this

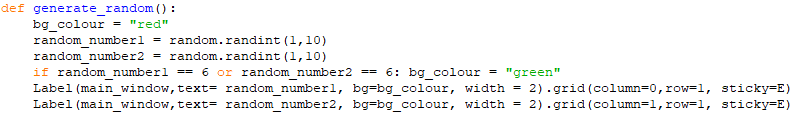
Task time

* Change the second random number so it also changes colour
* Get the colours to change to yellow if you get a 6

# and & or

**Introducing IF**

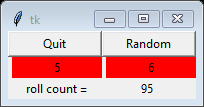
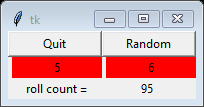
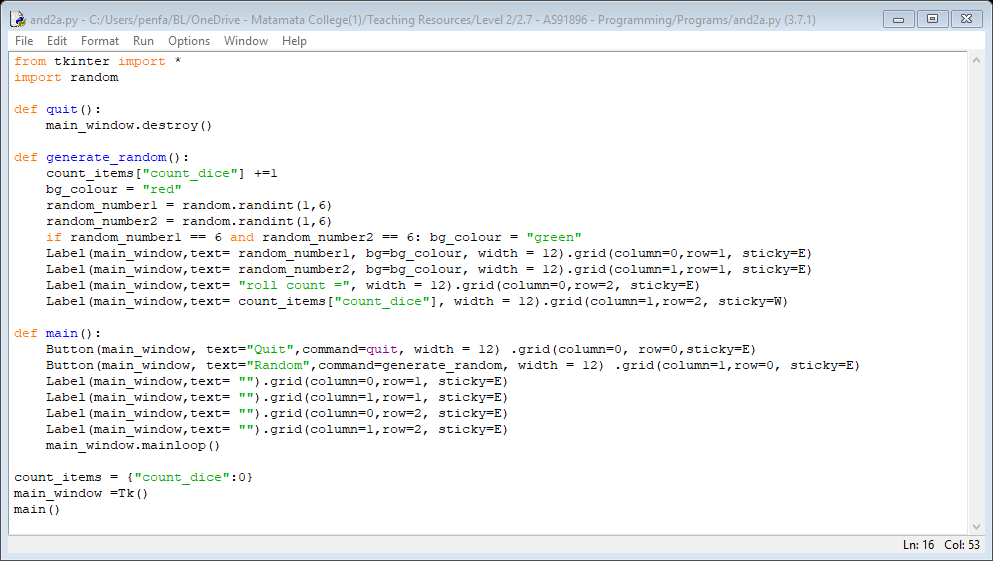
* Let’s start by looking at these words - we use them every day
* With **or** either condition can be true. If it is raining **or** bright sun, use an umbrella
* With **and** both conditions must be true. If it is raining **and** you are outside use an umbrella
* Let’s change the program and look at **or** first



* Notice how **width=2** has been added to get rid of the issue of red and green together
* The width argument is in characters - we use **2** as it makes it wide enough
* You can see the code has been re-arranged so the if statement can work with both numbers

Task time

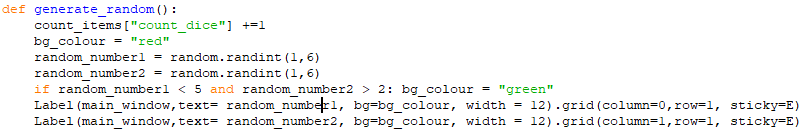
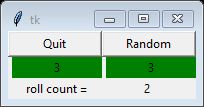
* Change it so both numbers need to be 6 before the colour changes
* Add and show a counter so you know how many times before you get two sixes. Hint, you will need to use a dictionary to do this
* Add a third number and check when this is a 6
* Change it so it is like flipping a dice and you can only get 0 and 1



* We needed to use a dictionary containing the count to make this work
* It ensures **count\_items** starts at zero at the start of the program

## Other operators

* So far, we have been working with **==** only. There are many other comparisons that can be made
* For greater than use **random\_number >5** so 6,7,8 etc will all pass the test
* For greater or equals use **random\_number >= 5** so 5,6,7 etc will all pass the test
* For less than you use **<** instead or **<=** for less than or equals
* A different one is not equals to. Use **random\_number != 5** so 2,3,4,6,7 will pass the test, not 5

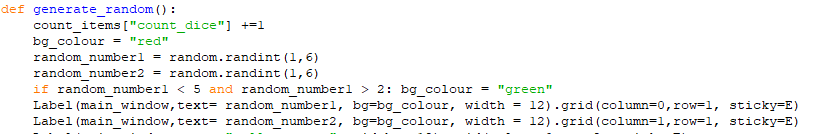
Task time

* Change the program so both numbers must be above 4 to make the background green
* Next, make it green if both numbers are not 5
* Next, make it green if either number is not 4

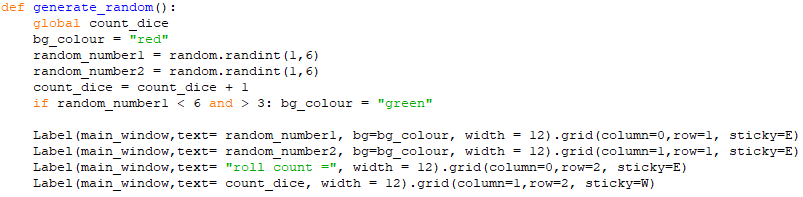
## the most common mistake

**More IF**

* So far, we have been using **and** & **or** to compare two different variables
* **if random\_number1 == 6 and random\_number2 == 6: bg\_colour = "green"**
* But what if we have only one variable we want to make two comparisons with, such as if the variable is less than 5 and greater than 2 (so numbers 3 & 4 are okay)



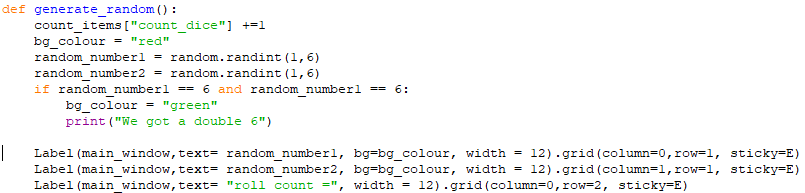
* You can see that it is the same as the previous program - we just have the same variable twice
* Many people will try to do what you see below - this does not work!



# if with more arguments

**Longer IF**

* So far, we have only wanted to do one thing when the if statement is true
* In many cases you want to do multiple things
* To do this hit enter - it should automatically tab in. If it does not check you put in the **:**
* Then just type in the rest of the lines you need
* Once finished use the delete key so you are no longer tabbed in



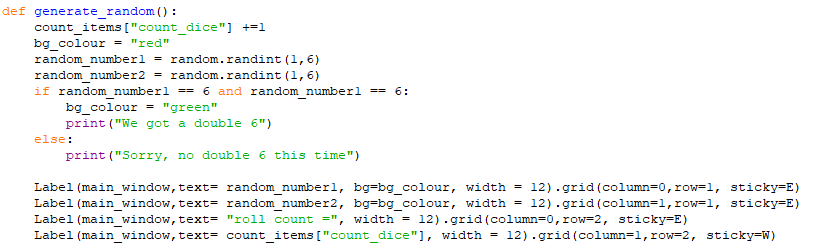
* Do not press the space key multiple times - it probably won’t work - delete and tab

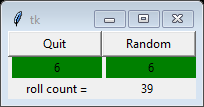
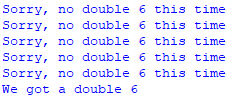
Task time

* Add to the print statement so it says how many times it took to get the double 6
* Add extra code to reset the counter after you get a double 6 (this is quite difficult)
* Add a second counter and get it to say the average number of times it takes to get double 6

# What else?

* **else** is what happens when the **if** condition is not met
* You should always use the multi-line **if** layout when using **else**



# don’t forget good coding practice

**Reminder time**

* To save space some of the good coding practice has not been shown on these notes
* This does NOT mean you should not be following good practice
* Variables that do not change within the program are constants
* Good comments are needed to explain the code



* Have a look at the comments, can you see any spelling or grammar errors?
* Python has no built-in spelling/grammar checker. You need to do it yourself
* Alternatively, copy the code into Word or Google Docs to check