```
>>> import random
>>> random.randint (0, 10)
```

these two items one after the other will generate a random integer (hence, rand- int-)

gb_list = [‘good’, ‘bad’] - two strings inside this list, “good” and “bad” are the strings, gb_list is the list, the brackets indicate the list

```
print(“It is “ + random.choice(gb_list))
```

This will generate a randomized answer from the two answers possible in the list.

Note - in order to do this, you must first “import random”.

Putting random.choice before a variable will generate a random choice, so long as you’ve initially imported random.

—

**strings are just pushing two things together, i.e. ‘5’ (string), + ‘7’ (second string) = 57 - integers are actually adding them, i.e. 5 + 7 = 12

—

Important values:
“*” = multiply
“%” = divide
“/“ = can also be divide
“\” = special character (can precede ‘ or a space or “n” or “p” (which would be next or previous))
gb

—

a loop happens when a list is printed because it will print as many times as there are strings, meaning,

in list = [0, 1, 2, 3, 4], there are 5 pieces and therefore it will print 5 lines of code.

then another piece of printing this list is that it goes as follows:

```
num_list_5 = num_list_5 + [5]
num_list_5
[0. 1, 2, 3, 4, 5]
for num in num_list_5:
        (tab) print(num)
```

—

```
list_two_piece = [“yes”, “no”]
for piece in list_two_piece:
        print(‘a’)
…
…
a
a
```

It will run the loop twice over, replacing both pieces with a.

—

```
for character in “hello”:
        print(character)
h
e
```

```
l
l
o
```

(will print each time it sees a new character - runs a command when it sees “h”, then “e”, then “l”, “l”, “o”)

```
for character in [“hello”]:
        print(character)
```

it will only run ONE TIME because the brackets have made it a list, and within that list is one string, which is “hello”.

—

the tab implies that it is inside the loop (it follows the rules).

```
for char in “hello”: (important!! you need that colon!!)
        print(char)
        print(‘a’)
```

this will print like so:

```
h
a
e
a
l
a
l
a
o
a
```

because you’ve asked it to print both every character and also add a in a new line after every character if “print(char)” or “print(‘a’)” were NOT INDENTED, they would only run one time - printed like that’s all you needed from that function. Because it is in indented, it will run as many characters as there are in “hello”.

—

in the case of the Emoticon exercise,

```
*important first step* import random
*first variable* eyes = [':', ';', '8', 'B']
*second variable* noses = ['0', 'o', '-', '~']
*third variable* mouths = [')', '(', 'c', 'C', 'O', 'L', '|']
*define out_string as blank* out_string = ''
*defining “num” as the amount of variables, in this case three for each facial part (eye, nose, mouth)* for num in [0,1,2]:
*tab, define out_string as itself (currently blank) plus random.choice plus the variable you want to go first*
        out_string = out_string + random.choice(eyes)
*same thing now adding out_string to the second variable*
        out_string = out_string + random.choice(noses)
*same thing but I added a blank space at the end for aesthetic purposes*
        out_string = out_string + random.choice(mouths) + " "
*then you print the whole out_string you’ve created by compiling these items* print(out_string)
```

this should come out as a different thing every time, but some random combination of these features, example being:

```
:O) :o( ;~)
```

—

```
num_list = [0,1,2,3,4]
for num in num_list:
        if num > 2:
```

```
        print(num)
```

this prints
3
4

because it checks each line in the list to see if True, and does not run if not - we asked (we tabbed in and so we asked) for the num to be greater than 2 in order to print.

—

Differentiating str and int (string and integer) -
‘5’ + 6 cannot exist because ‘5’ is a string and 6 is an integer.

You can, however, distinguish these things as long as you clarify when there’s an integer -

sleep_amount = int(sleep_amount) will transfer whatever someone says in response to sleep_amount = input(“how many hrs of sleep?”) into just a number - transferring it into becoming an integer.

—

you can transfer an item into an integer or a string by adding int(variable) or str(variable).

—

example of grading -

```
score = input("What is your score? > ")
score = int(score)
if score >= 90 and score <= 100:
   print("You got an A! Congratulations.")
```

```
if score >= 80 and score <= 89:
    print("You got a B! Still good!")
if score >= 70 and score <= 79:
    print("You got a C. It's fine.")
if score >= 60 and score <= 69:
    print("You got a D. Better luck next time.")
if score >= 0 and score <= 59:
    print("You got an F. Buddy, you failed.")
```

*important things here -

```
if variable (less than, less than equal, etc) colon
        tab print(“whatever you want it to say”)
```

so if someone said in response “89” in this case, it would say, “You got a b! Still good!”

—

there is also a modified element which is “%”, or modulus, which will divide and give the remainder. This is advanced and I don’t even understand it!