|  |  |
| --- | --- |
| Título  Código 1 | |
| Código  print("Hello World!")  print("Hello again")  print("I like typing this.")  print("This is fun.")  print('Yay! Printing.')  print("I'd much rather you 'not'.")  print('I "said" do not touch this. ') | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 3 | |
| print("I will now count my chickens:")  print("Hens", 25 + 30 / 6)  print("Roosters", 100 - 25 \* 3 % 4)  print("Now I will count the eggs:")  print(3 + 2 + 1 - 5 + 4 % 2 - 1 / 4 + 6)  print("Is it true that 3 + 2 < 5 - 7?")  print(3 + 2 < 5 - 7)  print("What is 3 + 2?", 3 + 2)  print("What is 5 - 7?", 5 - 7)  print("Oh, that's why it's False.")  print("How about some more.")  print("Is it greater", 5 > -2)  print("Is it greater or equal?", 5 >= -2)  print("Is it less or equal?", 5 <= -2) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 5 | |
| Codigo  my\_name = 'Zed a. Shaw'  my\_age = 35 # not a lie  my\_height = 74 # inches  my\_weight = 180 #lbs  my\_eyes = 'Blue'  my\_teeth = 'White'  my\_hair = 'Brown'  print(f"Let's talk about {my\_name}.")  print(f"He's {my\_height} inches tall.")  print(f"He's {my\_weight} puonds heavy.")  print("Actually that's not too heavy.")  print(f"He's got {my\_eyes} eyes and {my\_hair} hair")  print(f"His teeth are usually {my\_teeth} depending on the coffee.")  # this line is tricky, try to get it exactly right  total = my\_age + my\_height + my\_weight  print(f"If I add {my\_age}, {my\_height}, and {my\_weight} I get {total}. ") | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 6 | |
| Codigo  types\_of\_people = 10  x = f"There are {types\_of\_people} types of people."  binary = "binary"  do\_not = "don't"  y = f"Those who know {binary} and those who {do\_not}."  print(x)  print(y)  print(f"I said: {x}")  print(f"I also said: '{y}'")  hilarious = False  joke\_evaluation = "Isn't that joke so funny?! {}"  print(joke\_evaluation.format(hilarious))  w = "This is the left side of..."  e = "a string with a right side."  print(w + e) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 7 | |
| Codigo  print("Mary had a little lamb. ")  print("Its fleece was white as {}." . format('snow') )  print("And everywhere that Mary went.")  print("." \* 10) # what'd that do?  end1 = "C"  end2 = "h"  end3 = "e"  end4 = "e"  end5 = "s"  end6 = "e"  end7 = "B"  end8 = "u"  end9 = "r"  end10 = "g"  end11 = "e"  end12 = "r"  #watch end = '' at the end. try removing it to see what happens  print(end1 + end2 + end3 + end4 + end5 + end6, end=' ')  print(end7 + end8 + end9 + end10 + end11 + end12) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 8 | |
| Codigo  formatter = "{} {} {} {}"  print(formatter.format(1, 2, 3, 4))  print(formatter.format("one", "two", "three", "four"))  print(formatter.format(True, False, False, True))  print(formatter.format(formatter, formatter, formatter, formatter))  print(formatter.format(      "Try your",      "Own text here",      "Maybe a poem",      "Or a song about fear"  )) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 9 | |
| Codigo  days = "Mon Tue Thu Fri Sat Sun"  months = "Jan\nFeb\nMar\nApr\nMay\nJun\nJul\nAug"  print("Here are the days: ", days)  print("Here are the months: ", months)  print("""  There's somethins going on here.  With the three double\_quotes.  We'll be able to type as much as we like.  Even 4 lines if we want, or 5, or 6.  """) | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 10 | |
| Codigo.  tabby\_cat = "\tI'm tabbed in."  persian\_cat = "I'm split\non a line."  backslash\_cat = "I'm \\ a \\ cat."  fat\_cat = """  I'll do a list:  \t\* Cat food  \t\* Fishies  \t\* Catnip\n\t\* Grass  """  print(tabby\_cat)  print(persian\_cat)  print(backslash\_cat)  print(fat\_cat) | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 11 | |
| Codigo.  print("How old are you?", end=' ')  age = input()  print("How tall are you?", end=' ')  height = input()  print("How mouch do you weight?", end=' ')  weight = input()  print(f"So, you're {age} old, {height} tall and {weight} heavy.") | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 12 | |
| Codigo  age = input("How old are you? ")  height = input("How tall are you? ")  weight = input("How much do you weigh? ")  print(f"So, you're {age} old, {height} tall and {weight} heavy.") | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 13 | |
| Codigo  from sys import argv  # read the WYSS section for how to run this  script = argv[0]  first = argv[1]  second = argv[2]  print("The script is called:", script)  print("Your first variable is:", first)  print("Your second variable is:", second) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 16 | |
| Codigo.  from sys import argv  script, filename = argv  print(f"We're going to erase {filename}.")  print("If you don't want that, hit CTRL-C (^C).")  print("If you do want that, hit RETURN.")  input("?")  print("Opening the file...")  target = open(filename, 'w')  print("Truncating the file. Goodbye!")  target.truncate()  print("Now I'm going to ask you for three lines.")  line1 = input("line 1: ")  line2 = input("line 2: ")  line3 = input("line 3: ")  print("I'm going to write these to the file.")  target.write(line1)  target.write("\n")  target.write(line2)  target.write("\n")  target.write(line3)  target.write("\n")  print("And finally, we close it.")  target.close() | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 17 | |
| Codigo  def print\_two(\*args):      arg1, arg2 = args      print(f"arg1: {arg1}, arg2: {arg2}")  # ok, that \*args is actually pointless, we can just do this  def print\_two\_again(arg1, arg2):      print(f"arg1: {arg1}, arg2: {arg2}")  # this just takes one argument  def print\_one(arg1):      print(f"arg1: {arg1}")  # this one takes no arguments  def print\_none():      print("I got nothin'.")  print\_two("Zed","Shaw")  print\_two\_again("Zed","Shaw")  print\_one("First!")  print\_none() | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 19 | |
| Codigo  def cheese\_and\_crackers(cheese\_count, boxes\_of\_crackers):      print(f"You have {cheese\_count} cheeses!")      print(f"You have {boxes\_of\_crackers} boxes of crackers!")      print("Man that's enough for a party!")      print("Get a blanket.\n")  print("We can just give the function numbers directly:")  cheese\_and\_crackers(20, 30)  print("OR, we can use variables from our script:")  amount\_of\_cheese = 10  amount\_of\_crackers = 50  cheese\_and\_crackers(amount\_of\_cheese, amount\_of\_crackers)  print("We can even do math inside too:")  cheese\_and\_crackers(10 + 20, 5 + 6)  print("And we can combine the two, variables and math:")  cheese\_and\_crackers(amount\_of\_cheese + 100, amount\_of\_crackers + 1000) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 20 | |
| Codigo  from sys import argv  script, input\_file = argv  def print\_all(f):      print(f.read())  def rewind(f):      f.seek(0)  def print\_a\_line(line\_count, f):      print(line\_count, f.readline())  current\_file = open(input\_file)  print("First let's print the whole file:\n")  print\_all(current\_file)  print("Now let's rewind, kind of like a tape.")  rewind(current\_file)  print("Let's print three lines:")  current\_line = 1  print\_a\_line(current\_line, current\_file)  current\_line = current\_line + 1  print\_a\_line(current\_line, current\_file)  current\_line = current\_line + 1  print\_a\_line(current\_line, current\_file) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 21 | |
| Codigo  def add(a, b):      print(f"ADDING {a} + {b}")      return a + b  def subtract(a, b):      print(f"SUBTRACTING {a} - {b}")      return a - b  def multiply(a, b):      print(f"MULTIPLYING {a} \* {b}")      return a \* b  def divide(a, b):      print(f"DIVIDING {a} / {b}")      return a / b  print("Let's do some math with just functions!")  age = add(30, 5)  height = subtract(78, 4)  weight = multiply(90, 2)  iq = divide(100, 2)  print(f"Age: {age}, Height: {height}, Weight: {weight}, IQ: {iq}")  # A puzzle for the extra credit, type it in anyway.  print("Here is a puzzle.")  what = add(age, subtract(height, multiply(weight, divide(iq, 2))))  print("That becomes: ", what, "Can you do it by hand?") | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 24 | |
| Codigo  print("Let's practice everything.")  print('You\'d need to know \'bout escapes with \\ that do:')  print('\n newlines and \t tabs.')  poem = """  \tThe lovely world  with logic so firmly planted  cannot discern \n the needs of love  nor comprehend passion from intuition  and requires an explanation  \n\t\twhere there is none.  """  print("--------------")  print(poem)  print("--------------")  five = 10 - 2 + 3 - 6  print(f"This should be five: {five}")  def secret\_formula(started):      jelly\_beans = started \* 500      jars = jelly\_beans / 1000      crates = jars / 100      return jelly\_beans, jars, crates  start\_point = 10000  beans, jars, crates = secret\_formula(start\_point)  # remember that this is another way to format a string  print("With a starting point of: {}".format(start\_point))  # it's just like with an f"" string  print(f"We'd have {beans} beans, {jars} jars, and {crates} crates.")  start\_point = start\_point / 10  print("We can also do that this way:")  formula = secret\_formula(start\_point)  # this is an easy way to apply a list to a format string  print("We'd have {} beans, {} jars, and {} crates.".format(\*formula)) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 29 | |
| Codigo.  people =20  cats = 30  dogs = 15  if people < cats:      print("Too many cats! The world is doomed!")    if people > cats:      print("Not many cats! The world is saved!")  if people < dogs:      print("the world is drooled on!")  if people > dogs:      print("The world is dry!")  dogs += 5  if people >= dogs:      print("People are greater than or equal to dogs.")  if people <= dogs:      print("Peple are less than or equal to dogs.")  if people == dogs:      print("People are dogs.") | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 30 | |
| Codigo  people = 30  cars = 40  trucks = 15  if cars > people:      print("We solud take the cars.")  elif cars < people:      print("We should not take the cars.")  else:      print("We can't decide.")  if trucks > cars:      print("Thas too many trucks.")  elif trucks < cars:      print("Maybe we could take the trucks.")  else:      print("We still can't decide.")  if people > trucks:      print("Alright, let's just take the trucks.")  else:      print("Fine, let's stay home then.") | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 31 | |
| Codigo.  print("""You entre a dark room with two doors.  Do you go through door #1 or door #2?""")  door = input("> ")  if door == "1":      print("There's a giant bear here eating a chesse cake.")      print("What do you do?")      print("1. Take the cake.")      print("2. Scream at the bear.")        bear = input ("> ")        if bear == "1":          print("The bear eats your face off. Good job!")      elif bear == "2":          print("The bear eats your legs off. Good job!")      else:          print(f"Well, doing {bear} is probably better.")          print("Bear runs away.")  elif door == "2":      print("You stare into the endless abyss at Cthulhu's retina.")      print("1. Blueberries.")      print("2. Yellow jacket clothespins.")      print("3. Understanding revolvers yelling melodies.")        insanity = input("> ")        if insanity == "1" or insanity == "2":          print("Your body survives powered by a mind of jello")          print("Good job!")      else:          print("The insanity rots your eyes into a pool of muck.")          print("Good job!")  else:      print("You stumble around and fall on a knife and die. Good job!") | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 32 | |
| Codigo.  the\_count = [1, 2, 3, 4, 5]  fruits = ['apples', 'oranges', 'pears', 'apricots']  change = [1, 'pennies', 2, 'dimes', 3, 'quarters']  # this first kind of for-loop goes through a list  for number in the\_count:      print(f"This is count {number}")    # same as above  for fruit in fruits:      print(f"A fruit of type: {fruit}")    # also we can go through mixed lists too  # notice we have to use {} since we don't know what's in it  for i in change:      print(f"I got {i}")  # we can also build lists, first start with an empty one  elements = []  # then use the range function to do 0 to 5 counts  for i in range(0, 6):      print(f"Adding {i} to the list.")      # append is a function that lists understand      elements.append(i)  # now we can print them out too  for i in elements:      print(f"Element was: {i}") | |
| Ejecución. | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 33 | |
| Codigo  i = 0  numbers = []  while i < 6:      print(f"At the top i is {i}")      numbers.append(i)      i = i + 1      print("Numbers now: ", numbers)      print(f"At the bottom i is {i}")  print("The numbers: ")  for num in numbers:      print(num) | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 35 | |
| Codigo.  from sys import exit  def gold\_room():      print("This room is full of gold. How much do you take?")      choice = input("> ")      if "0" in choice or "1" in choice:          how\_much = int(choice)      else:          dead("Man, learn to type a number.")      if how\_much < 50:          print("Nice, you're not greedy, you win!")          exit(0)      else:          dead("You greedy bastard!")  def bear\_room():      print("There is a bear here.")      print("The bear has a bunch of honey.")      print("The fat bear is in front of another door.")      print("How are you going to move the bear?")      bear\_moved = False      while True:          choice = input("> ")            if choice == "take honey":              dead("The bear looks at you then slaps your face off.")          elif choice == "taunt bear" and not bear\_moved:              print("The bear has moved from the door.")              print("You can go through it now.")              bear\_moved = True          elif choice == "taunt bear" and bear\_moved:              dead("The bear gets pissed off and chews your leg off.")          elif choice == "open door" and bear\_moved:              gold\_room()          else:              print("I got no idea what that means.")  def cthulhu\_room():      print("Here you see the great evil Cthulhu.")      print("He, it, whatever stares at you and you go insane.")      print("Do you flee for your life or eat your head?")      choice = input("> ")      if "flee" in choice:          start()      elif "head" in choice:          dead("Well that was tasty!")      else:          cthulhu\_room()  def dead(why):      print(why, "Good job!")      exit(0)  def start():      print("You are in a dark room.")      print("There is a door to your right and left.")      print("Which one do you take?")      choice = input("> ")      if choice == "left":          bear\_room()      elif choice == "right":          cthulhu\_room()      else:          dead("You stumble around the room until you starve.")  start() | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 36 | |
| Codigo  ten\_things = "Apples Oranges Crows Telephone Light Sugar"  print("Wait there are not 10 things in that list. Let's fix that.")  stuff = ten\_things.split(' ')  more\_stuff = ["Day", "Night", "Song", "Frisbee","Corn", "Banana", "Girl", "Boy"]  while len(stuff) != 10:  next\_one = more\_stuff.pop()  print("Adding: ", next\_one)  stuff.append(next\_one)  print(f"There are {len(stuff)} items now.")  print("There we go: ", stuff)  print("Let's do some things with stuff.")  print(stuff[1])  print(stuff[-1]) # whoa! fancy  print(stuff.pop())  print(' '.join(stuff)) # what? cool!  print('#'.join(stuff[3:5])) # super stellar! | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 39 | |
| Codigo.  states = {  'Oregon': 'OR',  'Florida': 'FL',  'California': 'CA',  'New York': 'NY',  'Michigan': 'MI'  }  # create a basic set of states and some cities in them  cities = {  'CA': 'San Francisco',  'MI': 'Detroit',  'FL': 'Jacksonville'  }  # add some more cities  cities['NY'] = 'New York'  cities['OR'] = 'Portland'  # print out some cities  print('-' \* 10)  print("NY State has: ", cities['NY'])  print("OR State has: ", cities['OR'])  # print some states  print('-' \* 10)  print("Michigan's abbreviation is: ", states['Michigan'])  print("Florida's abbreviation is: ", states['Florida'])  # do it by using the state then cities dict  print('-' \* 10)  print("Michigan has: ", cities[states['Michigan']])  print("Florida has: ", cities[states['Florida']])  # print every state abbreviation  print('-' \* 10)  for state, abbrev in list(states.items()):  print(f"{state} is abbreviated {abbrev}")  # print every city in state  print('-' \* 10)  for abbrev, city in list(cities.items()):  print(f"{abbrev} has the city {city}")  # now do both at the same time  print('-' \* 10)  for state, abbrev in list(states.items()):  print(f"{state} state is abbreviated {abbrev}")  print(f"and has city {cities[abbrev]}")  print('-' \* 10)  # safely get a abbreviation by state that might not be there  state = states.get('Texas')  if not state:  print("Sorry, no Texas.")  # get a city with a default value  city = cities.get('TX', 'Does Not Exist')  print(f"The city for the state 'TX' is: {city}") | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 44 | |
| Codigo  class Parent(object):  def altered(self):  print("PARENT altered()")  class Child(Parent):  def altered(self):  print("CHILD, BEFORE PARENT altered()")  super(Child, self).altered()  print("CHILD, AFTER PARENT altered()")  dad = Parent()  son = Child()  dad.altered()  son.altered() | |
| Ejecución | Código QR del repositorio en GitHub |

|  |  |
| --- | --- |
| Título  Código 51 | |
| Codigo  from flask import Flask  from flask import render\_template  from flask import request  app = Flask(\_\_name\_\_)  @app.route("/hello")  def index():      name = request.args.get('name', 'Nobody')      if name:          greeting = f"Hello, {name}"      else:          greeting = "Hello World"      return render\_template("index.html", greeting=greeting)  if \_\_name\_\_ == "\_\_main\_\_":      app.run() | |
| Ejecución | Código QR del repositorio en GitHub |