FT Bibliotecas DateTime, Math e Random

Cria uma pasta com o nome **UFCD10793_BibliotecasMathRandomDateTime**. Escreve cada um dos programas das alíneas seguintes num ficheiro distinto a guardar na pasta criada.

Reproduz o seguinte código num ficheiro .py (um por biblioteca) para explorares as funcionalidades da biblioteca

Biblioteca DateTime

```
import datetime

################################################################################
#             O Objeto Datatime (Contem data e hora)
################################################################################
agora=datetime.datetime.now() #Cria Objeto datetime com a data e hora atual
aniversario=datetime.datetime(1988, 10, 23, 8, 20, 0, 0) #Cria Objeto datetime com a data e
hora personalizado

#Aceder a cada valor do objeto datetime
print(agora.year)
print(agora.month)
print(agora.day)
print(agora.hour)
print(agora.minute)
print(agora.second)
print(agora.microsecond)


#Fazer calculos com datas
```

```
dif=agora-aniversario
print(dif)

#Cria uma string com data num formato especifico
data_str=agora.strftime("%H:%M:%S")
print(type(data_str))
print(data_str)

#Cria um objeto datetime através de uma string
festival="20/10/2023"
festival_dtime=datetime.datetime.strptime(festival, "%d/%m/%Y")
print(type(festival_dtime))
print(festival_dtime)

################################################################################
#             O Objeto date (Contem data)
################################################################################


hoje=datetime.date.today() #Cria Objeto date com a data atual
aniversario=datetime.date(1992, 5, 1) #Cria Objeto date com a data personalizado

#Aceder a cada valor do objeto date
print(hoje.year)
print(hoje.month)
print(hoje.day)

#Fazer calculos com datas
dif=hoje-aniversario
print(dif)

#Cria uma string com data num formato especifico
data_str=hoje.strftime("%d:%m:%Y")
print(type(data_str))
print(data_str)

#Cria um objeto date através de uma string Usando um formato especificado.
festival="2023-10-25"
festival_dtime=datetime.date.fromisoformat(festival)
```

```
print(type(festival_dtime))
print(festival_dtime)


################################################################################
#             O Objeto time (Contem Hora)
################################################################################

data=datetime.datetime.now() #Cria Objeto data e hora com a data atual
agora=data.time() #Cria Objeto time e hora atual

almoco=datetime.time(12, 45, 50, 0) #Cria Objeto date com a data personalizado

#Aceder a cada valor do objeto time
print(agora.hour)
print(agora.minute)
print(agora.second)
print(agora.microsecond)

#Cria uma string com data num formato especifico
time_str=agora.strftime("%H:%M:%S")
print(type(time_str))
print(time_str)


#Cria um objeto time através de uma string
jantar="20:10:20"
jantar_time=datetime.time.fromisoformat(jantar)
print(type(jantar_time))
print(jantar_time)
```

Biblioteca Math

```
import math

# Constants
print(math.pi)              # Prints the value of pi (3.141592653589793)
print(math.e)               # Prints the value of Euler's number (2.718281828459045)

# Basic arithmetic functions
print(math.ceil(4.2))       # Rounds up to the nearest integer (5)
print(math.floor(4.9))      # Rounds down to the nearest integer (4)
print(math.trunc(4.9))      # Truncates the decimal part of a number (4)
print(math.fabs(-4.2))      # Returns the absolute value of a number (4.2)

# Exponential and logarithmic functions
print(math.exp(2))          # Returns e raised to the power of x (7.3890560989306495)
print(math.log(10))         # Returns the natural logarithm of x (2.302585092994046)
print(math.log10(100))      # Returns the base-10 logarithm of x (2.0)

# Trigonometric functions
print(math.sin(math.pi/2))  # Returns the sine of x (1.0)
print(math.cos(math.pi))    # Returns the cosine of x (-1.0)
print(math.tan(0))          # Returns the tangent of x (0.0)
print(math.radians(180))    # Converts degrees to radians (3.141592653589793)

# Hyperbolic functions
print(math.sinh(1))         # Returns the hyperbolic sine of x (1.1752011936438014)
print(math.cosh(1))         # Returns the hyperbolic cosine of x (1.5430806348152437)
print(math.tanh(1))         # Returns the hyperbolic tangent of x (0.7615941559557649)

# Angular conversion functions
print(math.degrees(math.pi/2))  # Converts radians to degrees (90.0)
print(math.radians(180))        # Converts degrees to radians (3.141592653589793)

# Power and square root functions
print(math.pow(2, 3))       # Returns x raised to the power of y (8.0)
print(math.sqrt(9))         # Returns the square root of x (3.0)

# Other functions
print(math.factorial(5))    # Returns the factorial of x (120)
print(math.gcd(24, 36))     # Returns the greatest common divisor of x and y (12)
print(math.isclose(0.1 + 0.2, 0.3))  # Checks if two values are close (True)

# Additional functions
print(math.acos(0.5))       # Returns the arc cosine of x in radians (1.0471975511965979)
print(math.asin(0.5))       # Returns the arc sine of x in radians (0.5235987755982989)
print(math.atan(1))         # Returns the arc tangent of x in radians (0.7853981633974483)
print(math.atan2(1, 1))     # Returns the arc tangent of y/x in radians (0.7853981633974483)
print(math.hypot(3, 4))     # Returns the Euclidean norm, sqrt(x*x + y*y) (5.0)
print(math.degrees(math.atan(1)))  # Converts radians to degrees (45.0)
print(math.radians(45))           # Converts degrees to radians
```

Biblioteca Random

```
import random

# Random number generation
print(random.random())              # Generates a random float between 0 and 1 (exclusive)
print(random.uniform(1, 10)) # Generates a random float within a specified range (inclusive)
print(random.randint(1, 6))  # Generates a random integer within a specified range (inclusive)
print(random.randrange(0, 101, 5)) #Generates a random integer in a specified range with a step

# Random sequences
numbers = [1, 2, 3, 4, 5]
random.shuffle(numbers)             # Shuffles the elements in a list randomly
print(numbers)

random_sample = random.sample([1, 2, 3, 4, 5], 3)  # Generates a random sample without
replacement
print(random_sample)

random_choice = random.choice(['apple', 'banana', 'orange'])  # a random element on s sequence
print(random_choice)

# Random distributions
print(random.normalvariate(0, 1))
# Generates a random float from a normal distribution with mean 0 and variance 1
print(random.gauss(0, 1))            # Alias for random.normalvariate

print(random.expovariate(1))
 # Generates a random float from an exponential distribution with rate 1
print(random.uniform(1, 10))  # Generates a random float within a specified range (inclusive)

# Random boolean values
print(random.choice([True, False]))  # Chooses a random boolean value (True or False)

# Setting the random seed
random.seed(42)                     # Sets the random seed to ensure reproducibility
print(random.random())              # Generates a random float using the same seed

random.seed()                       # Sets the random seed based on the current time
print(random.random())              # Generates a new random float

# Random bytes
random_bytes = random.randbytes(5)   # Generates a random bytes object of a specified length
print(random_bytes)

# Other functions
print(random.getrandbits(16))        # Generates a random integer with a specified number of
bits
print(random.choices(['red', 'green', 'blue'], k=3))
# Chooses multiple random elements from a sequence with replacement
```