Modules vs Packages

Module: are simply files with the **". py"** extension containing Python code that can be imported inside another Python Program. Module contains functions and global variables . It is an executable file and to organize all the modules we have the concept called Package in Python. E.g from math import pow, import math

Packages : A python package is a collection of modules. Modules that are related to each other are mainly put in the same package. When a module from an external package is required in a program, that package can be imported and its modules can be put to use. This directory contains Python modules also have __init__.py file by which the interpreter interprets it as a Package.

RA20 APTT

Math Module

•The **math module** is a standard module in Python and is always available.

•Python math module contains most famous mathematical functions, which includes trigonometric functions, representation functions, logarithmic functions, etc.

• Math module also defines two mathematical constants, i.e.Pie and Euler number.

•To use mathematical functions under this module, you have to import the module using import math.

import math or **import math as m**

math.sqrt(4) **m.sqrt(4)**

•Some funtions math.log10(20), math.pow(2,4), math.floor(10.4) , math.ceil(10.5), math.factorial() , math.modf(10.5)

RA20 APTT

List creation

Function	Description
ceil(x)	Returns the smallest integer greater than or equal to x.
copysign(x, y)	Returns x with the sign of y
fabs(x)	Returns the absolute value of x
factorial(x)	Returns the factorial of x
floor(x)	Returns the largest integer less than or equal to x
fmod(x, y)	Returns the remainder when x is divided by y
frexp(x)	Returns the mantissa and exponent of x as the pair (m, e)
fsum(iterable)	Returns an accurate floating point sum of values in the iterable
isfinite(x)	Returns True if x is neither an infinity nor a NaN (Not a Number)
isinf(x)	Returns True if x is a positive or negative infinity
isnan(x)	Returns True if x is a NaN
ldexp(x, i)	Returns x * (2**i)
modf(x)	Returns the fractional and integer parts of x
trunc(x)	Returns the truncated integer value of x
exp(x)	Returns e**x
expm1(x)	Returns e**x - 1
log(x[, b])	Returns the logarithm of x to the base b (defaults to e)
log1p(x)	Returns the natural logarithm of 1+x
log2(x)	Returns the base-2 logarithm of x
log10(x)	Returns the base-10 logarithm of x
pow(x, y)	Returns x raised to the power y

RA20 APTT

List creation

Function	Description
sqrt(x)	Returns the square root of x
acos(x)	Returns the arc cosine of x
asin(x)	Returns the arc sine of x
atan(x)	Returns the arc tangent of x
atan2(y, x)	Returns atan(y / x)
cos(x)	Returns the cosine of x
hypot(x, y)	Returns the Euclidean norm, sqrt(x*x + y*y)
sin(x)	Returns the sine of x
tan(x)	Returns the tangent of x
degrees(x)	Converts angle x from radians to degrees
radians(x)	Converts angle x from degrees to radians
acosh(x)	Returns the inverse hyperbolic cosine of x
asinh(x)	Returns the inverse hyperbolic sine of x
atanh(x)	Returns the inverse hyperbolic tangent of x
cosh(x)	Returns the hyperbolic cosine of x
sinh(x)	Returns the hyperbolic cosine of x
tanh(x)	Returns the hyperbolic tangent of x
pi	Mathematical constant, the ratio of circumference of a circle to it's diameter (3.14159...)
e	mathematical constant e (2.71828...)

Random Module

• Python offers **random module** that can generate **random numbers**.

• These are **pseudo-random number** & sequence of number generated depends on the seed.

•Python random module implements pseudo-random number generators for various distributions, including integer and float (real).

import random

print(random.random())

 # Output 0.24480512307264823 You may get a different number

•The **random.random()** is the most basic function of the random module.

•Almost all functions of the random module depend on the basic function **random().**

•random() return the next random floating-point number in the range [0.0, 1.0].

RA20 APTT

Random

```
import random

# random number from 0 to 1
print(random.random())
# Output 0.16123124494385477

# random number from 10 to 20
print(random.randint(10, 20))
# Output 18

# random number from 10 to 20 with step 2
print(random.randrange(10, 20, 2))
# Output 14

# random float number within a range
print(random.uniform(5.5, 25.5))
# Output 5.86390810771935

# random choice from sequence
print(random.choice([10, 20, 30, 40, 50]))
# Output 30
```

```
# random sample from sequence
print(random.sample([10, 20, 30, 40, 50], k=3))
# Output [50, 10, 20]

# random sample without replacement
print(random.choices([10, 20, 30, 40, 50], k=3))
# Output [30, 10, 40]

# random shuffle
x = [10, 20, 30, 40, 50, 60]
random.shuffle(x)
print(x)
# [60, 10, 30, 20, 50, 40]

# random seed
random.seed(2)
print(random.randint(10, 20))
# 10
random.seed(2)
print(random.randint(10, 20))
# 10
```

RandInt and Randrange

• **randint()** Generate random integer number from the inclusive range. E.g random.randint(0, 10) will return random number from **[0, 1, 2, 3, 4, 5, 6, 7, 8 ,9, 10].**

•**randint** considers both the start and stop numbers while generating random integers

•**randrnage()** is used to generate random integer from the given **exclusive range** by specifying the increment. For example, **random.randrange(0, 10, 2)** will return any random number between 0 and 10 (like 0, 2, 4, 6, 8).

•The randrange() **doesn't consider the stop number** while generating a random integer. It is an exclusive random range.

•In **rangrange** Out of three, **two parameters are optional**. i.e., start and step

RA20 APTT

Random number of a specific length

```
import random
# random number of length 4
num1 = random.randint(1000, 9999)
# random number of length 4 with step 2
num2 = random.randrange(1000, 10000, 2)
print(num1, num2)

# Output 3457 5116
```

RA20 APTT

Random negative integer

```
import random

singed_int = random.randrange(-60, -5)
print(singed_int)
# Output -16
```

Random positive or negative integer

```
import random

for i in range(5):
    print(random.randint(-10, 10), end=' ')
# Output 10 -1 5 -10 -7
```

RA20 APTT

List of Random Integers

```
import random

random_list = []
# Set a length of the list to 10
for i in range(0, 10):
    # any random numbers from 0 to 1000
    random_list.append(random.randint(0, 1000))
print(random_list)
# Output [994, 287, 65, 994, 936, 462, 839, 160, 689, 624]
```

RA20 APTT

List of Random Integers Without Duplicates

•To make sure each number in the list is unique, use the **random.sample()** method to generate a **list of unique random numbers.**

•The **sample() returns** a sampled list of selected random numbers within a range .

•**Sample () never repeats the element** so we get a list of random numbers **without duplicates.**

```
import random

# Generate 10 unique random numbers within a range
num_list = random.sample(range(0, 1000), 10)
print(num_list)
# Output [499, 580, 735, 784, 574, 511, 704, 637, 472, 211]
```

RA20 APTT

Generate a secure random integer

• Till now all examples of random numbers are not cryptographically secure.

• The **cryptographically secure random generator** generates random numbers using **synchronization methods** to ensure that no two processes can obtain the same number simultaneously.

•For a security-sensitive application use **secret module** (Available in Python version higher than 3.6.)

```
import secrets

# secure random integer
# from 0 to 10
secure_num = secrets.randbelow(10)
print(secure_num)
# Output 5
```

RA20 APTT

Summaryabout randint() and randrange()

• **Use randint()** when you want to generate a random number from an inclusive range.

•Use **randrange()** when you want to generate a random number from an exclusive range by specifying the increment.

•The randint() rand randrange() works only with integers not with float numbers.

•In randrange() step must not be 0, or else you will get a ValueError.

• The start should not be greater than stop if you are using all positive numbers.

• **If start is greater than stop**, you will get a **ValueError in** randrange(), But, you **can set a start value greater than stop** if you are using a **negative step value**.

```
import random

# ValueError: empty range for randrange()
print(random.randrange(100, 10, 2))
```

```
import random

print(random.randrange(100, 10, -2))
# output 60
```

Random String in Python

• To generate **random strings** in Python, the **string and random modules** are used.

•random.choice() is used to generate strings in **which characters may repeat**, while . **random.sample() is used for non-repeating characters**

Method	Description
string.ascii_uppercase	Returns a string with uppercase characters
string.ascii_lowercase	Returns a string with lowercase characters
string.ascii_letters	Returns a string with both lowercase and uppercase characters
string.digits	Returns a string with numeric characters
string.punctuation	Returns a string with punctuation characters

RA20 APTT

Random String in Python

```
import random
import string

# printing lowercase
letters = string.ascii_lowercase
print ( ''.join(random.choice(letters) for i in range(10)) )

# printing uppercase
letters = string.ascii_uppercase
print ( ''.join(random.choice(letters) for i in range(10)) )

# printing letters
letters = string.ascii_letters
print ( ''.join(random.choice(letters) for i in range(10)) )

# printing digits
letters = string.digits
print ( ''.join(random.choice(letters) for i in range(10)) )

# printing punctuation
letters = string.punctuation
print ( ''.join(random.choice(letters) for i in range(10)) )
```

RA20 APTT

Random String in Python

```
import random
import string
# printing lowercase
letters = string.ascii_lowercase
print(''.join(random.choice(letters) for i in range(10)))
```

```
import random
import string

def get_rand_string(length):
    letters = string.ascii_lowercase
    print(''.join(random.choice(letters) for i in range(length)))

get_rand_string(8)
get_rand_string(6)
get_rand_string(4)
```

Random String of mix Case and Digits

```
import random
import string

def get_rand_string(length):
    letters = string.ascii_letters
    print(''.join(random.choice(letters) for i in range(length)))

get_rand_string(8)
get_rand_string(6)
get_rand_string(4)
```

```
import random
import string
def get_rand_string(length):
    letters = string.digits
    print(''.join(random.choice(letters) for i in range(length)))
get_rand_string(8)
get_rand_string(6)
get_rand_string(4)
```

Random String of Specific letters & Without Repeating Characters

```
import random
import string

print(''.join(random.choice("abcdxyzpqr") for i in range(5)))
```

```
import random
import string

for i in range(3):
    # get random string of length 6 without repeating letters
    result_str = ''.join(random.sample(string.ascii_lowercase, 5))
    print(result_str)
```

RA20 APTT

Create Password

```
import random
import string

# get random password pf length 8 with letters, digits, and symbols
mixed_chars = string.ascii_letters + string.digits + string.punctuation
password = ''.join(random.choice(mixed_chars) for i in range(8))
print("Random password is:", password)
```

```
import random
import string

password = ''.join(random.choice(string.printable) for i in range(8))
print("Random password is:", password)
```

import string

Storing the sets of punctuation, digits, ascii_letters and spaces in variable result . It's a Pre-initialized string used as string constant

result = string.printable
print(result)

RA20 APTT

Practice Questions

1. Produce 10 random integers between 100 and 999 .
2. Produce 20 random lottery tickets and pick two lucky tickets as a winner. The lottery number must be 10 digits long All 20 ticket number must be unique.
3. Generate 6 digit random secure OTP
4. Choose a random character from a given String.
5. Generate a random Password which is 10 characters long and have at least 1 digit, and 1 special symbol.
6. Dice is Rolled in such a way that every time we get the same number
7. Multiplication of two random float numbers num1 between 0.1 and 1, num2 between 9.5 and 99.5

RA20 APTT

Practice Questions

Q 1

```
import random
print("Generating 10 random integer number between 100 and 999)
for num in range(10):
    print(random.randrange(100, 999   ), end=', ')
```

Q 2

```
import random
lot_tickets_list = []
print("creating 20 random lottery tickets")
# to get 100 ticket
for i in range(20):
    lot_tickets_list.append(random.randrange(1000000000, 9999999999))
# pick 2 luck tickets
winners = random.sample(lot_tickets_list, 2)
print("Lucky 2 lottery tickets are", winners)
```

RA20 APTT

Practice Questions

Q 3

```
import secrets
#Getting systemRandom class instance out of secrets module
sec_generator = secrets.SystemRandom()

otp = sec_generator.randrange(1000, 9999)
print("Secure Random OTP = ", otp)
```

Q 4

```
import random

name = 'Welcome to Python'
char = random.choice(name)
print("random char is ", char)
```

RA20 APTT

Practice Questions

Q 5

```
import random
import string

def randomPassword():
    randomSource = string.ascii_letters + string.digits + string.punctuation
    password = random.sample(randomSource, 6)
    password += random.sample(string.ascii_uppercase, 2)
    password += random.choice(string.digits)
    password += random.choice(string.punctuation)

    passwordList = list(password)
    random.SystemRandom().shuffle(passwordList)
    password = ''.join(passwordList)
    return password

print ("Password is ", randomPassword())
```

RA20 APTT

Practice Questions

Q 6

```
import random

dice = [1, 2, 3, 4, 5, 6]
print("Randomly selecting same number of a dice")
for i in range(5):
    random.seed(25)
    print(random.choice(dice))
```

Q 6

```
import random

print('Random number with seed 30')
for i in range(3):
    # Random number with seed 30
    random.seed(30)
    print(random.randint(25, 50))
```

Q 7

```
import random

num1  = random.random()
print("First Random float is ", num1)
num2 = random.uniform(9.5, 99.5)
print("Second Random float is ", num1)

num3 = num1 * num2
print("Multiplication is ", num3)
```

RA20 APTT

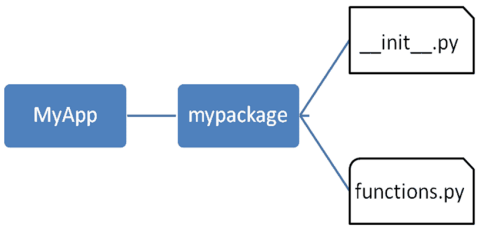
PACKAGES

- A package in Python takes the concept of the modular approach.
- A package can contain one or more relevant modules.
- A module can contain multiple objects, such as classes, functions, etc.
- Let us create a package named mypackage, using the following steps:
  - Create a new folder named e.g. C:\myapp
  - Inside myapp, create a subfolder with the name 'mypackage'.
  - Create an empty __init__.py file in the mypackage folder.
  - Using a Python-aware editor like Pycharm, IDLE, create modules greet.py and functions.py

RA20 APTT

PACKAGES

- A package in Python takes the concept of the modular approach.
- A package can contain one or more relevant modules.
- A module can contain multiple objects, such as classes, functions, etc.



RA20 APTT