Programming in Python[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

Academic year 2023/24, I semester

Lecture XIII: Random numbers

Random numbers

Pseudorandomness: the sequence of numbers is not predictable. . .

```python
from random import randint

# To get a random integer x in the set [1..10]
x = randint(1, 10)
```

```python
from random import randint

for _ in range(0,10):
   print(randint(1, 100))
```

unless you know the seed.

```python
from random import seed, randint

seed(292)
for _ in range(0,10):
   print(randint(1, 100))
```

Exercise

Write a Python program which chooses an integer 1–10 and asks to the user to guess it

- if the number given by the user is not 1–10, it prints "Invalid";
- if the number is the chosen one, it prints "Yes!";
- otherwise "You didn't guess it...".

Exercise

Write a Python program which chooses an integer 1–10 and asks to the user to guess it

- if the number given by the user is not 1–10, it prints “Invalid”;
- if the number is the chosen one, it prints “Yes!”;
- otherwise “You didn’t guess it...”.

Evolve the program: it should now ask until the user guess the number correctly, giving hints (“higher...”, “lower...”).

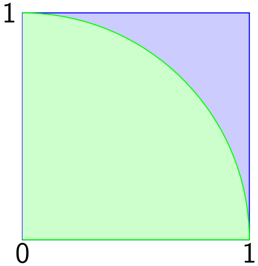
Exercise

Write a Python program which chooses an integer 1–10 and asks to the user to guess it

- if the number given by the user is not 1–10, it prints “Invalid”;
- if the number is the chosen one, it prints “Yes!”;
- otherwise “You didn’t guess it...”.

Evolve the program: it should now ask until the user guess the number correctly, giving hints (“higher...”, “lower...”).
How many tries in the worst case? Can you write a program guessing a number between 1 and `int(1e32)`

Example



- Blue square: 1
- Green area: $\frac{\pi}{4}$

The Monte Carlo method consists of choosing sample experiments at random from a large set and then making deductions on the basis of the probabilities estimated from frequency of occurrences.

Lecture XV: Using Third-party libraries

Third-party libraries

Python is “sold” *batteries included* (with many useful built-in libraries). Moreover, like many modern programming environments, it has standard online package directories that list libraries produced by independent developers.
`https://pypi.org/`
The Python package index currently lists almost 300K libraries!

Installing a library

The details are explained here: https://packaging.python.org/tutorials/installing-packages/

- In most cases it is very easy, the `pip` program does all the magic
- It is very important to understand the difference between a system-wide and a project-specific installation.

System-wide vs. Project-specific

If you don’t take special precautions, a package is installed in a way that makes it available to your Python system: every Python interpreter you launch sees them.

- In many cases, this is not what you want
- Different projects/programs might depend on different versions of the libraries
- Libraries themselves depend on other libraries, you want to understand exactly which packages your program is using in order to reproduce the settings on other machines

Virtual environments

Python provides the idea of virtual development environments (`venv`)

- You can create one with: `python -m venv CHOOSE_A_NAME`
- You must activate it (syntax depends on your OS): `CHOOSE_A_NAME\Scripts\activate.bat`
- In an active virtual environment all the installation are confined to it
- You can get the list of installed packages with `pip freeze`

Simplified venv administration

Virtual environments are key to avoid messing up your system. Many tools simplify their administration.

- pipenv (my preferred one, we will use this)
- poetry (similar to pipenv, currently less popular, but it has a better dependency control, a bit more complex)
- conda (uses its own package index, great flexibility and complexity, manage different python versions)

Virtual environments caveats

When you are working in a Python virtual environment, remember to launch all your development tools “inside” the virtual space.
For example, to use IDLE don’t click on the main application launcher, instead: `python -m idlelib`.