PHOT 110: Introduction to programming

Project topics: project 1

Michaël Barbier, Spring semester (2023-2024)

Introduction

There are two projects to be completed for the PHOT 110 course during this semester. This file contains the project topics for the first project. The projects are strongly focused on one application and will force you to think not only about the Python code itself, but also about the problem-solving aspect.

You can and are encouraged to work together on projects, further, you can ask help from me, Hazan, and Metin (asking help will not influence your project grade). However your project report and corresponding script should be made individually and not copied from others or online resources.

Type of report for project 1

You need to send in both the script(s) used and a report. In the report you describe in a concise manner what the result was of your project and how you obtained it. Describe the crucial steps that you undertook to tackle the project task. Support your ideas with plots or schematic drawings. The report should be minimum 1 page, and maximum two pages including figures. The report should be in pdf-format, font-size 12.

Grading of the project

This project will count for 20% of your grade. Points are given on the combined effort of the project and the oral explanation of it during the exam. This means that you will know your final points only after the exam.

Please understand the code/report that you send in, you will be asked questions about it on the final exam. It is better to have a less "fancy" script than not being able to explain the code/report on the exam. The points will be calculated according to the geometric mean of (1) your grade of the code/report and (2) your explanation at the exam. Please ask help to your instructors on time, we might have not enough time to help you at the last day before the deadline of the report.

During this semester one more project will be made and total of projects made during the semester counts for 40% of your total grade for the course.

Project topics

Next is a list of 5 topics you can choose out for your project together with their task description.

Roller coaster g-forces

Main topics: **Mathematical functions**, **plots**

Roller coaster rides distort our feeling of balance, which people often consider pleasant as a challenge. However, roller coasters need to be designed carefully such that the people riding it will not get (too) uncomfortable or sick. A first check is to calculate the g-forces that a person will experience during the ride. See also the Publication on looping curvature by Liseberg (2005) and the separately provided reference: "Roller coasters without differential equations — a Newtonian approach to constrained motion" by Müller et al. (especially section 11).

In this project you will simulate the g-forces on the people inside a roller coaster cart of a ride which includes a looping. You can assume the roller coaster ride (and g-forces) to be approximately in 2D.

The task can be divided in following parts:

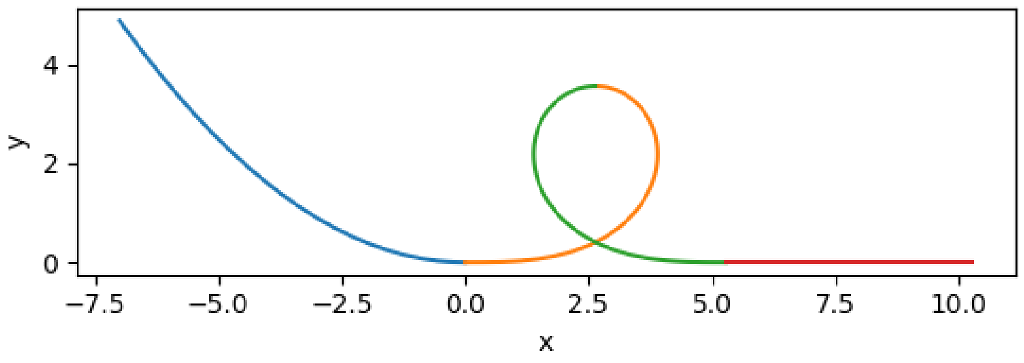


Figure 1: Roller coaster track with a Clothoid loop. The track is constructed piece-wise. For each piece we know the curvature radius.

- Create a parametric curve for the track in 2D. Take the first part a parabolic downhill part, the next a clothoid looping, and the last part a flat track. Take (x, y) as horizontal and vertical coordinates.
- Compute the velocity along the track for an object sliding on it (moving resistance free, point mass) using conservation of energy: $E_{total} = E_{kin} + V = mv^2/2 + mgh$ should be a constant. This gives us:

$$v_0^2 = 2\,g\,h_{\text{max}}, \qquad v_{\text{top}}^2 = 2\,g\,(h_{\text{max}} - h_{\text{loop}})$$

- Derive the g-force along the track from the velocity by $a_g = v^2/\rho$ where v is the velocity and ρ is the local curvature of the track.

The clothoid loop with "radius" R can be defined as:

$$\begin{cases} x(t) = R\,S(t) \\ y(t) = R\,C(t) \end{cases}$$

where $S(t)$ and $C(t)$ are the Fresnel integrals defined as:

$$S(t) = \int_0^t \sin^2\left(\frac{\pi}{2}u^2\right)du, \qquad C(t) = \int_0^t \cos^2\left(\frac{\pi}{2}u^2\right)du$$

To know the distance s along the track, that is, we can integrate along the parametric curve leading to $s = R\,t$. This allows us to calculate (x, y) as function of distance s:

$$\begin{cases} x(t) = R\,S(s/R) \\ y(t) = R\,C(s/R) \end{cases}$$

while s is in interval $[0, \sqrt{2}]$. Here we assumed s defined in half of the Clothoid loop (but the other half is its symmetric counterpart). The local curvature radius ρ is given by:

$$\rho = \frac{R^2}{\pi s}$$

The above Fresnel integrals $S(t)$ and $C(t)$ can be numerically calculated. See also the documentation for the Scipy implementation of the Fresnel integrals. You need to install the `scipy` package to make use of its functions.

Astroids game

Main topics: **keyboard input**, **Pygame library**, **coordinate transformations**

In an Asteroids arcade game you are the commander of a spaceship which got caught in an asteroid cloud. Luckily your spaceship is equipped with lasers so you can pulverize the asteroids before they hit you. For the Asteroids game see the Wikipedia entry on the original Asteroids game of Atari. For more information on real asteroid clouds see the NASA article on the Kuiper belt

Task description of the project:

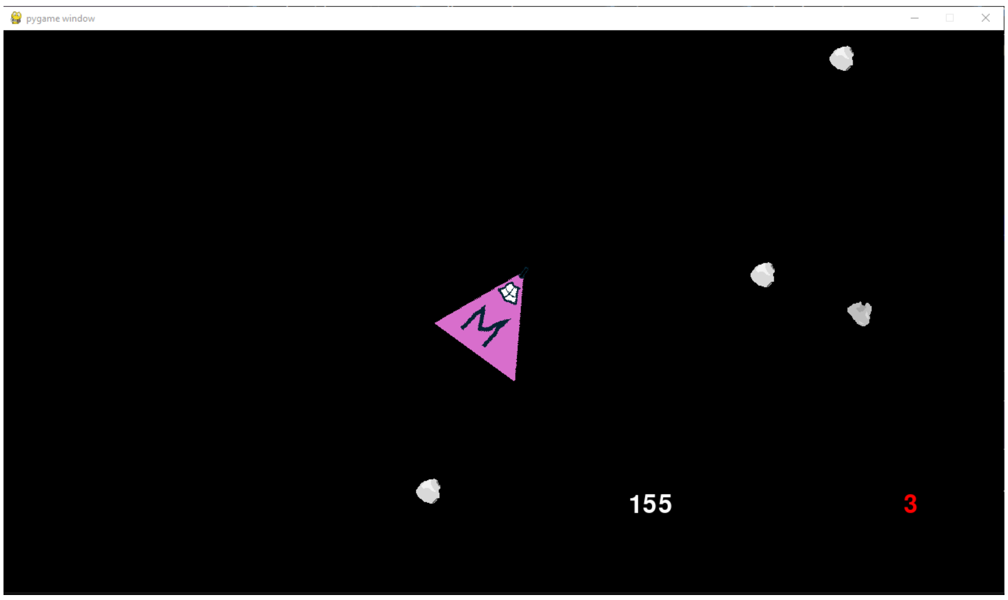
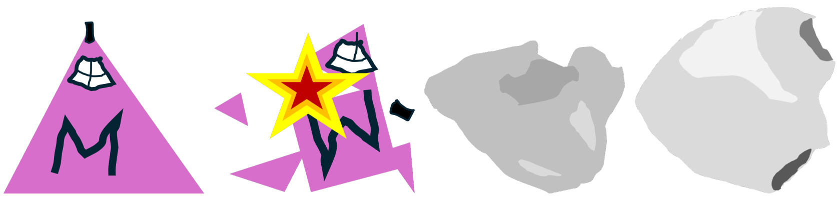


Figure 2: Screenshot asteroids game example



- Draw some images to use for your ship and asteroids
- During the game the spaceship is in the center of the screen
- The spaceship can rotate itself to shoot asteroids with a laser
- Asteroids fly in random directions (potentially harming the ship). Not all asteroids will hit the ship.

To obtain a graphical window, and facilitate keyboard interactions, you will use Pygame, a Python package dedicated to making games in Python, see the Pygame website.

A few more hints to get you started:

- To start off with your project have a look at the Pygame documentation: start with the example at the beginning and then add the functionality for your game.
- Loading an image (asteroid image, spaceship, etc.) and scaling it to the right size (to use inside the game) can be done using:

```python
SHIP_IMAGE_SIZE = (128, 128)
im_ship_original = pygame.image.load("images/spaceship.png").convert()
im_ship = pygame.transform.scale(im_ship_original, SHIP_IMAGE_SIZE)
```

- Rotation of images can be done using:

```python
im_ship_rot = pygame.transform.rotate(surface=im_ship, angle=angle)
```

- The y-axis of the screen is pointing down, you can create functions to convert between screen coordinates and the game coordinates (where the ship is in the origin).
- To put an image on the screen you use (realize that the destination parameter uses the location of the upper-left corner of the image, you need to correct for this for the image of ship or asteroid to be in the middle)

```
screen.blit(the_image, dest=(x,y))
```

- A rotated image is larger than the original, so the center point is different, so every time you rotate your ship you need to recalculate its center point.
- Check for collisions between ship and asteroids at each time point. Calculate the distance between the spaceship and the asteroids and verify whether this distance is large enough.

Fractals: the Dragon curve

Main topics: **recursive functions**, **plots**

Fractals are geometric shapes which are highly detailed and look similar at different scales. Typically you will see similar shapes when you zoom in. For more information on fractals see: Wikipedia page on fractals.

In this project the idea is to make a script to generate the Dragon fractal curve: Wikipedia page on Dragon_curve. Especially have a look at the construction on this Wikipedia page. The output would be something as in the figure below:

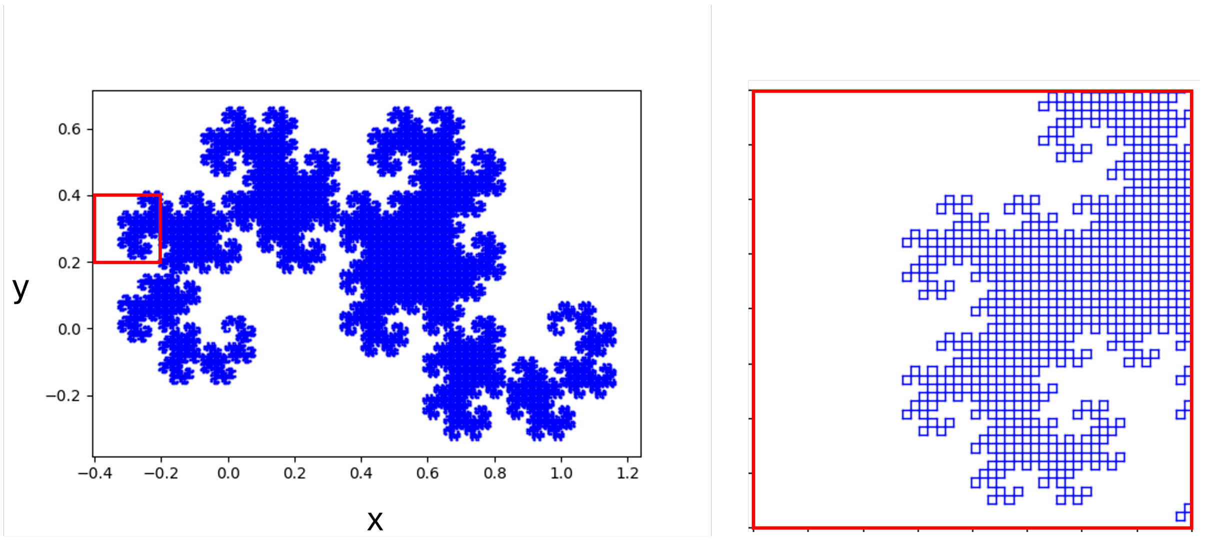


Figure 3: Example output of the Dragon fractal for the 17th iteration.

To generate the Dragon curve you should use recursive functions, i.e. a function which calls itself (this is just an example of a recursive function):

```python
def a_recursive_fct(generation, max_generation):
  # This function calls itself until certain depth/generation
  if generation < max_generation:
    a_recursive_fct(generation+1, max_generation)
  else:
    # do something
```

Realize that the stopping criterium for the recursive function is necessary, otherwise it would keep calling itself an infinite amount of times. Let's see another example of a recursive function:

```python
def recursive_print(n, max_n):
  # Performing actions before or after calling itself
  print("before_" + str(n))
  if n < max_n:
    recursive_print(n + 1, max_n)
  print("after_" + str(n))

recursive_print(0, 3)
```

```
before_0
before_1
before_2
before_3
after_3
after_2
after_1
after_0
```

Notice the order of the printed lines, it depends on whether we print before or after the function calls itself. We can also make multiple calls within a recursive function:

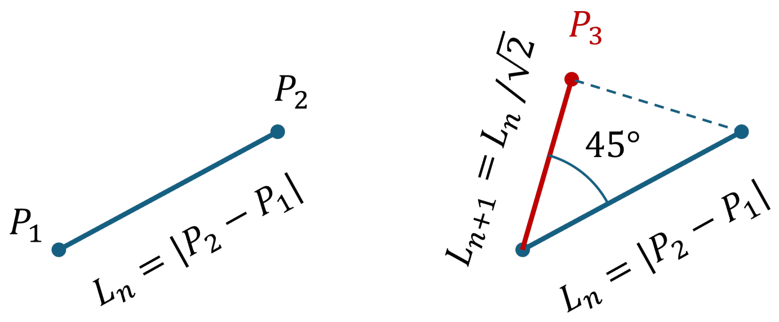
```python
def recursive_fct(n, max_n, text):
  if n < max_n:
    str1 = recursive_fct(n + 1, max_n, "0")
    str2 = recursive_fct(n + 1, max_n, "1")
    text = text + str1 + str2
  return text

# Start with an empty list
my_list = recursive_fct(0, 3, "")
print(my_list)
```

```
00011011001101
```

Hints to get started:

- The dragon curve starts with a single line connecting two points.
- choose your maximum depth relatively small: a depth equal to 16 is already a lot, start with less.
- At each step you divide a line in two different line pieces, which are either becoming a corner pointing above the line or below. So at each step you would call the function twice. To make a “corner” of a line piece you can use the following idea:



- You can perform rotations using complex numbers, or polar coordinates, or using the rotation matrix (see e.g. the Wikipedia page mentioned above).

Website crawling

Main topics: **urls**, **html inspection**, **Selenium/webdriver**

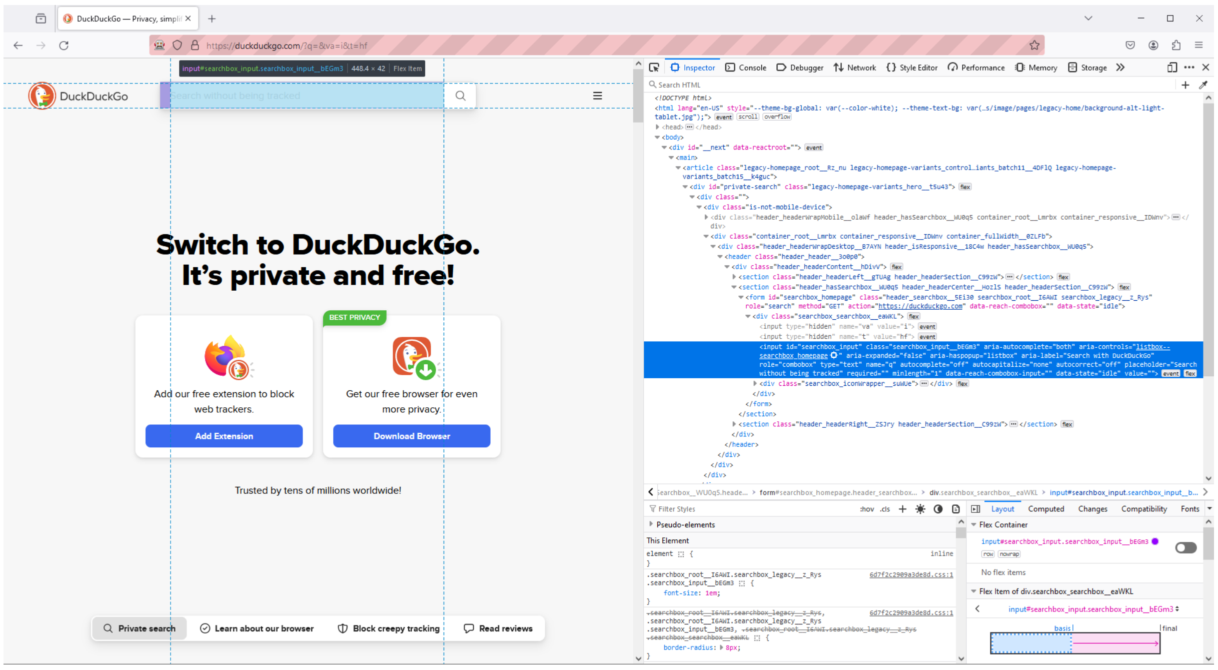
Applications will often connect to websites or web-services online to be able to provide their own functionality. Examples are:

- An application which combines a trajectory in Openstreetmaps with local weather forecasts to help you find an ideal time-slot to walk to/from work or school.
- An application which allows you to compare the prices of clothes from the main online shops.
- Application which collects information on chemical compounds from multiple databases, allowing optimal choice when designing e.g. a new pharmaceutical drug.

Within this project you will perform the automated collection of data of a major online webstore: https://www.n11.com/. Specifically you will automatically download the images of the first 10 sport shoes in the category.

To perform the task you will use the python package urllib3 to make http requests (to automatically download images) and Selenium (to mimick actual browsing). The task can be divided in parts as follows:

- Use your webbrowsers developer tool to find the HTML elements describing shoes. Therefore: browse to https://www.n11.com/, and search for the website elements via the inspection tool. An example of using the inspection tool can be found in the image below.



```
<!DOCTYPE html>
<html lang="en-US" style="--theme-bg-global: var(--color-white); --theme-text-bg: var(…s/image/pages/legacy-home/background-alt-light-tablet.jpg");"> event scroll overflow
  <head> ⋯ </head>
  <body>
    <div id="__next" data-reactroot=""> event
      <main>
        <article class="legacy-homepage_root__Rz_nu legacy-homepage-variants_control…iants_batch11__4DFlQ legacy-homepage-variants_batch15__k4guc">
          <div id="private-search" class="legacy-homepage-variants_hero__t5u43"> flex
            <div class="">
              <div class="is-not-mobile-device">
                <div class="header_headerWrapMobile__olaWf header_hasSearchbox__WU0q5 container_root__Lmrbx container_responsive__IDWnv"> ⋯ </div>
                <div class="container_root__Lmrbx container_responsive__IDWnv container_fullWidth__0ZLFb">
                  <div class="header_headerWrapDesktop__B7AYN header_isResponsive__18C4w header_hasSearchbox__WU0q5">
                    <header class="header_header__3o0p0">
                      <div class="header_headerContent__hDivV"> flex
                        <section class="header_headerLeft__gTUAg header_headerSection__C99zW"> ⋯ </section> flex
                        <section class="header_hasSearchbox__WU0q5 header_headerCenter__HozlS header_headerSection__C99zW"> flex
                          <form id="searchbox_homepage" class="header_searchbox__5Ei30 searchbox_root__I6AWI searchbox_legacy__z_Rys" role="search" method="GET" action="https://duckduckgo.com" data-reach-combobox="" data-state="idle">
                            <div class="searchbox_searchbox__eaWKL"> flex
                              <input type="hidden" name="va" value="i"> event
                              <input type="hidden" name="t" value="hf"> event
                              <input id="searchbox_input" class="searchbox_input__bEGm3" aria-autocomplete="both" aria-controls="listbox--searchbox_homepage" aria-expanded="false" aria-haspopup="listbox" aria-label="Search with DuckDuckGo" role="combobox" type="text" name="q" autocomplete="off" autocapitalize="none" autocorrect="off" placeholder="Search without being tracked" required="" minlength="1" data-reach-combobox-input="" data-state="idle" value=""> event flex
                              <div class="searchbox_iconWrapper__suWUe"> ⋯ </div> flex
                            </div>
                          </form>
                        </section>
                        <section class="header_headerRight__ZSJry header_headerSection__C99zW"> ⋯ </section> flex
                      </div>
                    </header>
                  </div>
                </div>
              </div>
```

```
earchbox__WU0q5.heade… > form#searchbox_homepage.header_searchbox… > div.searchbox_searchbox__eaWKL > input#searchbox_input.searchbox_input__b…
```

```
Pseudo-elements
This Element
element {                                                          inline
}
.searchbox_root__I6AWI.searchbox_legacy__z_Rys               6d7f2c2909a3de8d.css:1
.searchbox_input__bEGm3 {
    font-size: 1em;
}
~~.searchbox_root__I6AWI.searchbox_legacy__z_Rys~~,          6d7f2c2909a3de8d.css:1
.searchbox_root__I6AWI.searchbox_legacy__z_Rys
.searchbox_input__bEGm3, ~~.searchbox_root__I6AWI.searchbox_legacy__z_Rys~~
~~.searchbox_searchbox__eaWKL~~ {
    border-radius: 8px;
}
```

```
input#searchbox_input.searchbox_input__bEGm3
row nowrap
input#searchbox_input.searchbox_input__bEGm3
```

Figure 4: Using the inspection tool of the web developer tools in Firefox. In this case we identified the id of the search bar in the website of the search engine DuckDuckGo

- Afterward use this id to automatically fill in a search keyword by using Selenium, which is a library to automate a webbrowser. For this, test up front which keyword would go to the correct category (in this project sport shoes).
- Find the list of shoes and their corresponding images, within that page by again using the inspection tool of the webbrowser.
- Use the functions of Selenium to select for each sport shoe in the list the corresponding image.
- Use urllib3 to download the images from the website.

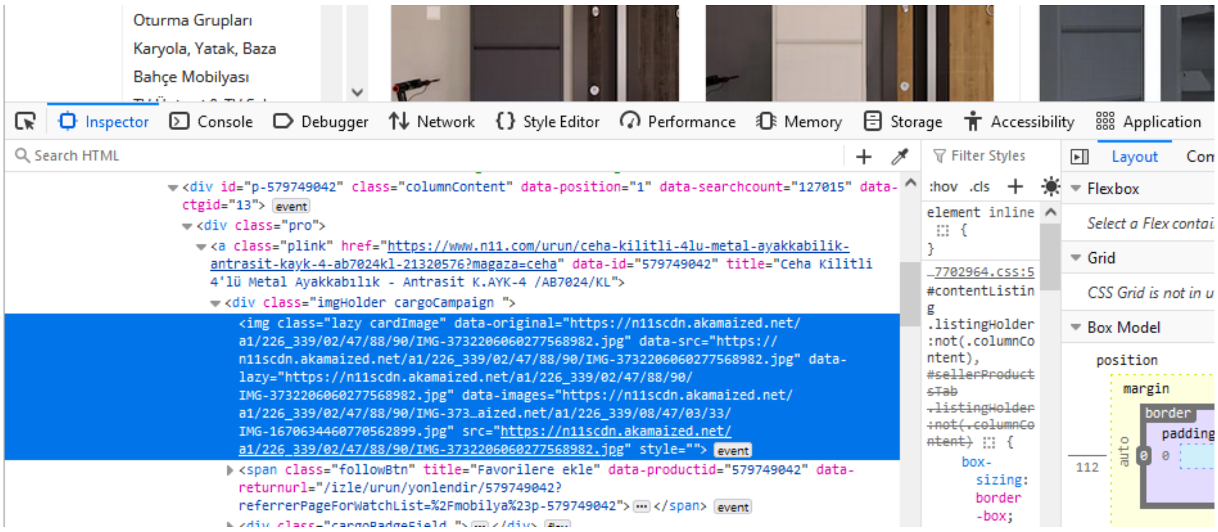
Hints to get started:

- To download an image from a website you need to know use the "src" entry of an "img" (html) element.

- If you have the "src" link, you can download the image via a GET request by employing urllib3:

```
# create a PoolManager instance
http = urllib3.PoolManager()
# Load the image
response_img = http.request('GET', url=img_link)
image = response_img.data
```

- Once downloaded you can save the image from the response data:



```
<div id="p-579749042" class="columnContent" data-position="1" data-searchcount="127015" data-ctgid="13"> event
  <div class="pro">
    <a class="plink" href="https://www.n11.com/urun/ceha-kilitli-4lu-metal-ayakkabilik-antrasit-kayk-4-ab7024kl-21320576?magaza=ceha" data-id="579749042" title="Ceha Kilitli 4'lü Metal Ayakkabılık - Antrasit K.AYK-4 /AB7024/KL">
      <div class="imgHolder cargoCampaign ">
        <img class="lazy cardImage" data-original="https://n11scdn.akamaized.net/a1/226_339/02/47/88/90/IMG-3732206060277568982.jpg" data-src="https://n11scdn.akamaized.net/a1/226_339/02/47/88/90/IMG-3732206060277568982.jpg" data-lazy="https://n11scdn.akamaized.net/a1/226_339/02/47/88/90/IMG-3732206060277568982.jpg" data-images="https://n11scdn.akamaized.net/a1/226_339/02/47/88/90/IMG-373…aized.net/a1/226_339/08/47/03/33/IMG-1670634460770562899.jpg" src="https://n11scdn.akamaized.net/a1/226_339/02/47/88/90/IMG-3732206060277568982.jpg" style=""> event
        <span class="followBtn" title="Favorilere ekle" data-productid="579749042" data-returnurl="/izle/urun/yonlendir/579749042?referrerPageForWatchList=%2Fmobilya%23p-579749042"> … </span> event
```

Figure 5: An example "img" html element which contains a "src" (from the n11 website).

```python
# This image data can then be saved to a file
with open(file_name, "wb") as f:
    f.write(image)
```

- You need to install Selenium as a package but also the Firefox "Gecko" webdriver (or the Chrome webdriver).

```python
from selenium import webdriver
```

- Selenium can be used as in the following example:

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys

# This will open the browser
browser = webdriver.Firefox()

# Go to the website DuckDuckGo (similar to Google search)
browser.get(url="https://www.duckduckgo.com")
# Next command selects the search box (id required)
elem = browser.find_element(By.ID, "searchbox_input")
# Supply the search word and press enter
elem.send_keys("Michael" + Keys.RETURN)
# Waiting a short time to allow webpage to load
# (not the best way to do it)
time.sleep(2)
# Extract information from the query
elem_from_response = browser.find_element(By.CSS_SELECTOR,
                        'a[data-testid="about-image"]')
```

```
# Within the "a" (anchor) element find the image
img_link = elem_from_response.find_element(By.TAG_NAME,
               "img").get_attribute("src")
```

Electronic piano

In this project we will make a very primitive synthesizer using the pc-speaker of the computer. The pc-speaker can be steered by sending sinusoidal waves. Depending on the frequency of the wave, the tone will be different. We will use the Python library sounddevice to send the wave shapes to the pc-speaker. Then we will use the keys of our keyboard as the keys of a piano. To listen for keyboard “key-press” events we will use the pynput library

To mimick a classical piano we will incorporate the Pythagorian scale. In short we have 7 musical notes: C D E F G A B C’, with frequency ratios = 1, 9/8, 81/64, 4/3, 3/2, 27/16, 243/128, 2. Here A represents the famous 440 Hz tone in the 4th octave. C’ is the C of the next octave, and each octave doubles the frequency.

The project can be divided in two tasks:

1. Let the computer play the musical notes of a very simple song (e.g. read in the note letters from a file or a string)
2. Make a little piano of your computer keyboard by implementing a key-listener so that certain keys will correspond to let’s say just seven musical notes of an octave.

Hints to perform the above tasks:

- Make a function to compute the frequency f according to the note letter and octave.
- Generate a sinus wave (t) with frequency f over an interval T of about 0.5 second long, thereby take a high sampling rate $N > 10^4$ of time points per second. Take the amplitude $A \approx 0.2$ not have a too loud sound.

$$\Phi(t) = A\sin(2\pi f t), \quad \text{with } t \in [0..T]$$

- Feed the sinusoidal wave to the pc-speaker using the sounddevice library and wait for the sound to play (use the sleep funtion of sounddevice):

```
import sounddevice

# .. code to make a wave

wave = create_wave(amplitude, octave, note_index, T)
sounddevice.play(wave)
sounddevice.sleep(int(np.round(1000 * T)))
```

- Change the envelope shape of the sinusoidal wave form to something more smooth than a square.

- Automate playing musical notes by reading note letter from a file or a string using a loop. Provide your script with a simple sequence of notes: e.g. Frere Jacques.
- Make a key listener to provide the notes to be played via the computer keyboard. You can adapt the example script for a keylistener in the documentation for pynput:

```
from pynput import keyboard

def on_press(key):
    try:
        print('alphanumeric key {0} pressed'.format(
            key.char))
    except AttributeError:
        print('special key {0} pressed'.format(
            key))

def on_release(key):
    print('{0} released'.format(
        key))
    if key == keyboard.Key.esc:
        # Stop listener
        return False

# Collect events until released
with keyboard.Listener(
        on_press=on_press,
        on_release=on_release) as listener:
    listener.join()
```

- Connect seven keys of the keyboard to the musical notes.