**Viet Nam National University – Ho Chi Minh City**

**University of Information Technology**

**Faculty of Information System**

**--------------------🙢🕮🙠--------------------**



**STUDY ON EXAM INVIGILATOR ASSIGNMENT ALGORITHMS AND BUILD AN APPLICATION WITH PYTHON**

**Instructor:** Vu Minh Sang

**Student:** Nguyen Hoang Long – 16520688

***Ho Chi Minh City, 2020***

**Acknowledgement**

Throughout the completion of this project, I have received a great deal of support and assistance from the member of Information System Laboratory.

First, I would like to express my deepest gratitude to Dr. Nguyễn Thanh Bình for accepting me to work at Information System Laboratory. He always willing and enthusiastic to assist in any way he could. Also, I want to mention my instructor Mr. Vũ Minh Sang for providing guidance and feedback throughout the completion of this report.

Second, special thanks to Mr. Huỳnh Thiện Ý, whose support as part of his science paper allowed us to go extra miles and special thanks to Mr. Nguyễn Sơn lâm for being a good partner while we are working together.

I am also thankful to the Faculty of Information System and all its member’s staff for all the considerate guidance.

Kind regards,

Nguyen Hoang Long

*Ho Chi Minh City, 2020*

**Comments**

**Table of contents**

[Part 1: Introduction 5](#_Toc60162957)

[**1.1** **Company introduction** 5](#_Toc60162958)

[**1.2** **Role** 5](#_Toc60162959)

[Part 2: Internship content 6](#_Toc60162960)

[**2.1 Working time** 6](#_Toc60162961)

[**2.2 Working process** 6](#_Toc60162962)

[Part 3: Knowledge learned and accomplishment 7](#_Toc60162963)

[**3.1 Genetic algorithm** 7](#_Toc60162964)

[**3.1.1 Overview** 7](#_Toc60162965)

[**3.1.2 Why use Genetic algorithm?** 8](#_Toc60162966)

[**3.1.3 Basic functionality of Genetic algorithm** 9](#_Toc60162967)

[**3.2 Python language** 10](#_Toc60162968)

[**3.2.1 What is Python and why choose Python?** 10](#_Toc60162969)

[**3.2.2 Application to work with Python** 10](#_Toc60162970)

[**3.2.3 Genetic algorithm function in python** 10](#_Toc60162971)

[**3.3 Python libraries** 12](#_Toc60162972)

[**3.3.1 Tkinter** 12](#_Toc60162973)

[**3.3.2 Pandas** 15](#_Toc60162974)

[**3.3.3 NumPy** 15](#_Toc60162975)

[**3.3.4 threading** 15](#_Toc60162976)

[**3.4 Graphical User Interface (GUI)** 15](#_Toc60162977)

[Part 4: Summary 20](#_Toc60162978)

[**4.1 Knowledge** 20](#_Toc60162979)

[**4.2 Skills** 20](#_Toc60162980)

[Part 5: References 21](#_Toc60162981)

# **Part 1: Introduction**

## **Company introduction**

The Information Systems Laboratory (ISL) was established on 1 February 2012 to strengthen the capability of training, researching, transfer of technology for the benefit of Vietnam’s economic growth. ISL located at room E9.1 of the University of Information Technology. ISL has a dozen of high-end personal computers and many equipment that meet demand for building cloud computing solution located at Data Center, which is a tremendous help to postgraduate research students, researchers, and lecturers.

The research activities of ISL are structured around three core topics:

* Knowledge-based systems
* Mobile information systems
* Spatial information systems

## **Role**

I was grateful to Dr. Nguyễn Thanh Bình for giving me a chance to be able to work at the Information Systems Laboratory as an intern.

Most of my responsibilities at Information Systems Laboratory were self-learning and complete small tasks from Mr. Bình in order to finish my graduate thesis.

Detail:

* Self-learning, research about optimization algorithm, Python libraries.
* Run algorithms test cases to solve assignment problem.
* Design and implement an application to apply the algorithm based on Python language.

# **Part 2: Internship content**

## **2.1 Working time**

I started my internship at the IS labs from 20/08/2020 to 27/11/2020.

Working time:

* Morning: from 8 a.m. to 12 a.m.
* Afternoon: from 13 p.m. to 4 p.m.

Working session during the week:

* I go to the labs 3 days a week from Tuesday to Thursday.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Monday | Tuesday | Wednesday | Thursday | Friday |
| Morning |  | X | X | X |  |
| Afternoon |  | X | X | X |  |

Table 2.1. Training time per week

## **2.2 Working process**

My objectives in the internship period are to learn about optimization algorithms, learn Python language and build a window application with Python in order to finish my graduate thesis.

Below is a timeline and brief description of my working process at ISL.

|  |  |
| --- | --- |
| **Time** | **Contents** |
| 20/08 – 27/08 | Research optimization algorithms. |
| 01/09 – 24/09 | Research Genetic algorithm. |
| 25/09 – 15/10 | Research Python language, libraries   * Tkinter: for UI. * Pandas: for working with data. * NumPy: for working with number, matrix. * Threading: for working with multiple thread. |
| 15/10 – 22/10 | Create UI for welcome window. |
| 22/10 – 29/10 | Create UI for main window. |
| 29/10 – 05/11 | Create UI for review data window. |
| 05/11 – 11/11 | Build a small database. |
| 12/11 – 24/11 | Apply Genetic algorithm and run test data. |
| 24/11 – 27/11 | Test run and make modifications to the application. |

Table 2.2. Timeline and description

# **Part 3: Knowledge learned and accomplishment**

## **3.1 Genetic algorithm**

### **3.1.1 Overview**

Genetic algorithms are a subset of a larger class of evolutionary algorithms that describe a set of techniques inspired by natural selection such as inheritance, mutation, and crossover. Genetic algorithms require both a genetic representation of the solution domain and a fitness function to evaluate the solution domain. The technique generates a population of candidate solutions and uses the fitness function to select the optimal solution by iterating with each generation. The algorithm terminates when the satisfactory fitness level has been reached for the population or the maximum generations have been reached.

Four main steps of Genetic algorithm:

**Step 1:** Initialize the population (set of initial solutions to the problem)

**Step 2:** Assign fitness score to each individual and create a new population using genetic operations. (crossover, selection, mutation)

**Step 3:** Assess adaptability of each individual in the population through the fitness function. Individuals in newly born population replace those in the previous population based on “fit” score. (Individual with higher score has more chance to be selected than those lower score)

**Step 4:** Check the stop conditions. If the stop condition is satisfied then the algorithm stops and returns the best possible solution, otherwise go back to step 2.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 3.1. Genetic algorithm workflow

### **3.1.2 Why use Genetic algorithm?**

Genetic algorithms have proven to be an enormously powerful and successful problem-solving strategy. Genetic algorithms have been used in a wide variety of fields to find solutions to problems that are more difficult than those faced by human designers. Thus, the solutions they come up with are often more efficient, more elegant, or more complex than anything comparable a human engineer would produce. [1]

### **3.1.3 Basic functionality of Genetic algorithm**

* **Initialization:**

The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. The initial population is generated randomly.

* **Selection**

During each successive generation, a portion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions ( as measured by a fitness function) are typically more likely to be selected.

There are a lot of techniques to select the individuals to be copied over into the next generation, some of them are: roulette wheel selection, rank selection, tournament selection, elitism selection,…

* **Crossover**

Crossover is a genetic operator used to combine the genetic information of two parents to generate new offspring.

* **Mutation**

After selection and crossover, we get new population full of individuals. Some are directly copied, and others are produced by crossover operation. Mutation occurs to maintain richness within the population and prevent premature convergence.

## **3.2 Python language**

### **3.2.1 What is Python and why choose Python?**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast.

### **3.2.2 Application to work with Python**

**PyCharm:** is an integrated development environment (IDE) used in computer programming, specifically for the Python language.

**Jupyter Notebook:** is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text.

### **3.2.3 Genetic algorithm function in python**

Below are some of the main functions of genetic algorithm that I have implemented.

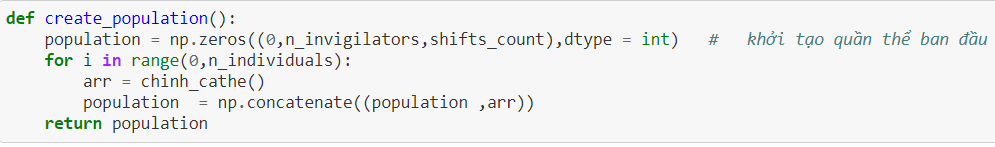


Figure 3.2 Function to create initial population

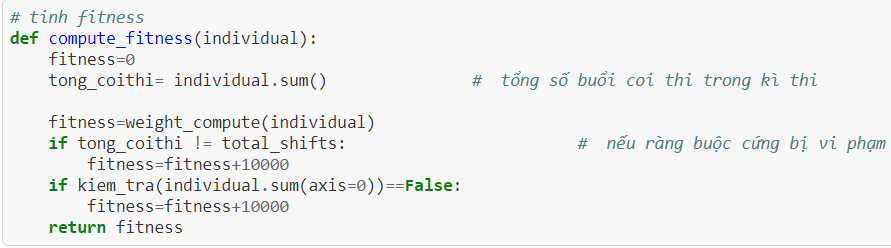


Figure 3.3 Fitness function

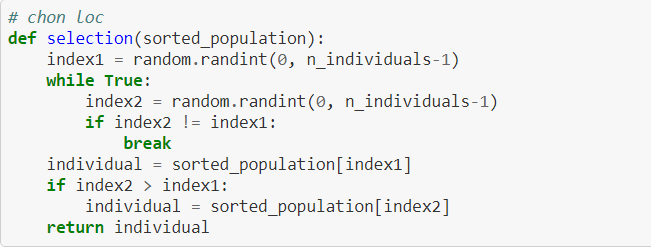


Figure 3.4 Selection function

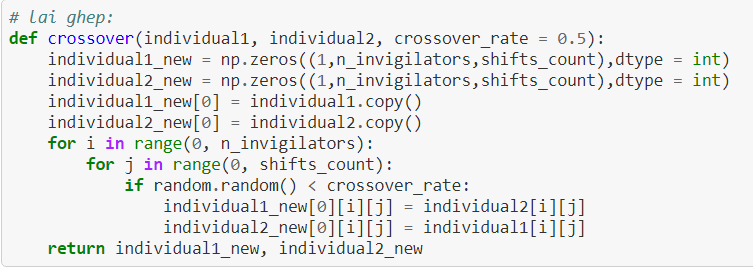


Figure 3.5 Crossover function

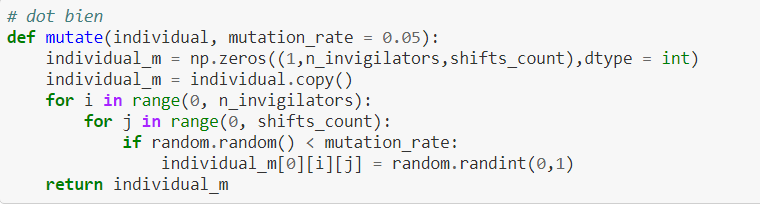


Figure 3.6 Mutation function

## **3.3 Python libraries**

### **3.3.1 Tkinter**

Tkinter is the standard Graphical User Interface (GUI) library for Python. It provides a powerful object-oriented interface to the Tk GUI toolkit. Python when combined with Tkinter provides a fast and easy way to create GUI applications.

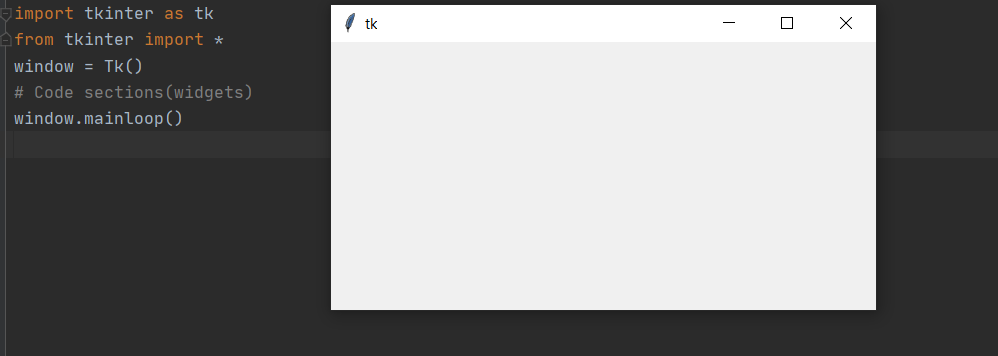


Figure 3.7 Example code to create a basic tkinter GUI window.

Tkinter provides various controls, such as buttons, labels, text boxes… used in a GUI application. These controls are commonly called **Widgets**.

Below is brief description of the widgets that I have used:

Note:

* **Master** – This represents the parent window.
* **Options** – This represent commonly used options for this widget, separated by commas.

**Frame:** the frame widget is very important as it works like a container, which is responsible for arranging the position of other widgets.

Syntax: x = Frame(master, options, …)

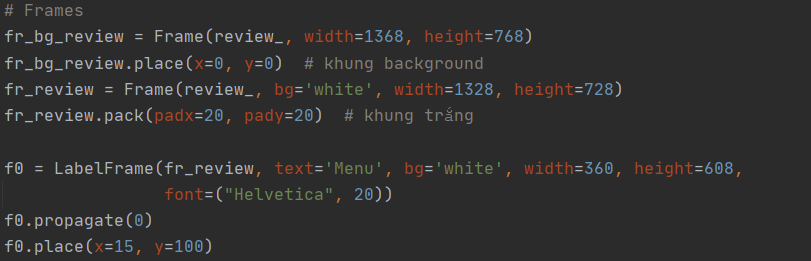


Figure 3.8 Frame widgets

**Button:** the button widget is used to display buttons in your application.

Syntax: x = Button(master, options, …)

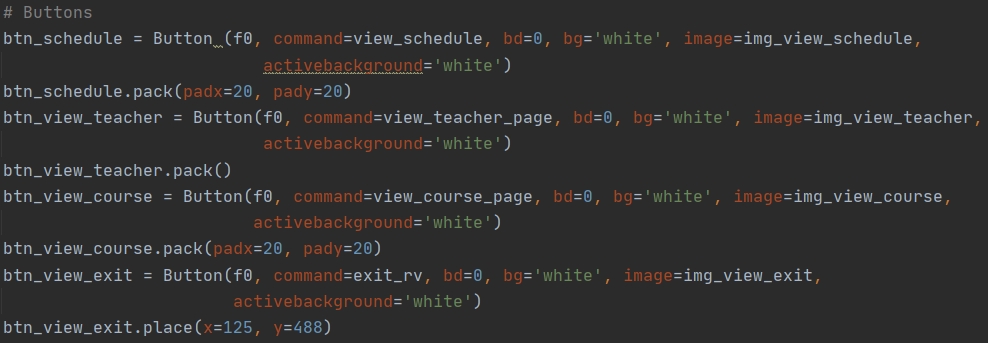


Figure 3.9 Button widgets

**Label:** the label widget implements a display box where you can place text or images.

Syntax: x = Label(master, options, …)

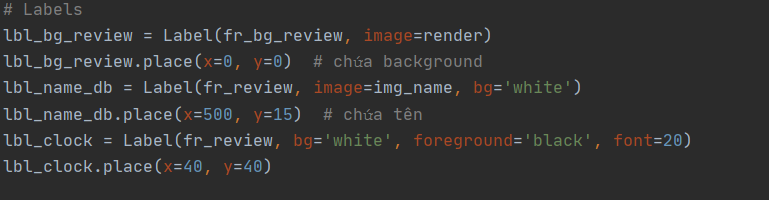


Figure 3.10 Label widgets

**ttk.Progressbar:** the ttk.Progressbar widget shows the status of a long-running operation. It has 2 modes: 1) the determinate mode which shows the amount of completed relative to the total amount of work to be done and 2) the indeterminate mode which provides an animated display to let the user know that work is progressing.

Syntax: x = ttk.Progressbar(master, options, …)

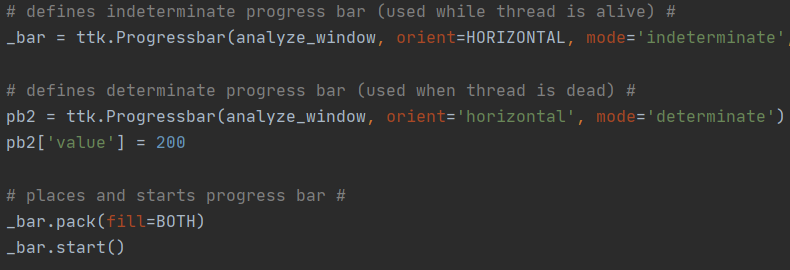


Figure 3.11 ttk.Progressbar widgets

**ttk.Treeview:** the ttk.Treeview widget displays a hierarchical collection of items.

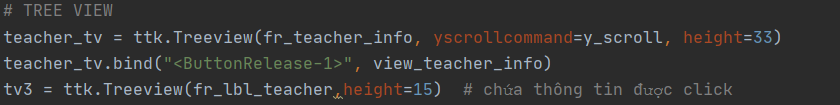


Figure 3.12 ttk.Treeview widgets

### **3.3.2 Pandas**

Pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive.

### **3.3.3 NumPy**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

### **3.3.4 threading**

Python threading allows you to have different parts of your program run concurrently and can simplify your design.

## **3.4 Graphical User Interface (GUI)**

In this section, I will show some of the GUI that have been implemented with the above widgets, using PyCharm as a compiler.

Below are some of the GUI windows:

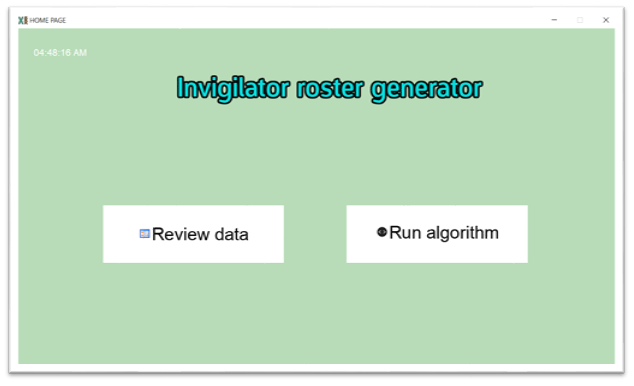


Figure 3.13 home UI

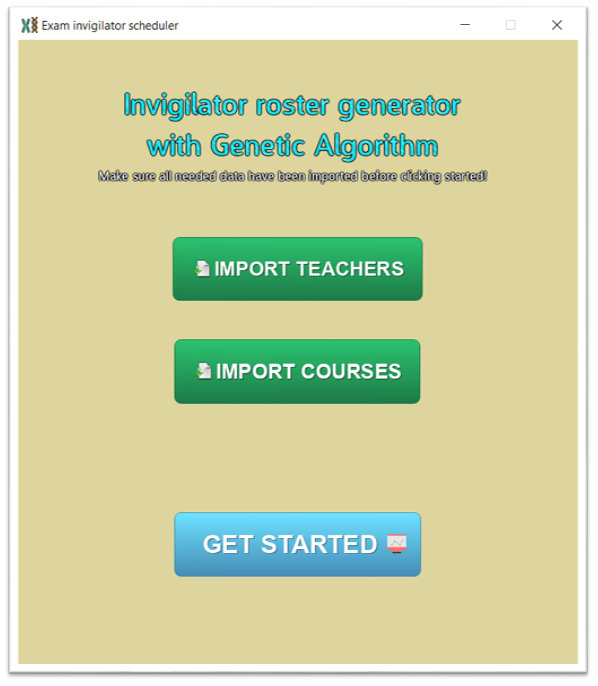


Figure 3.14 welcome UI

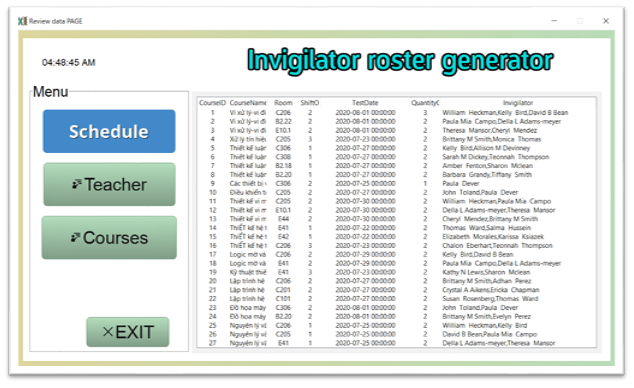


Figure 3.15 Review date UI

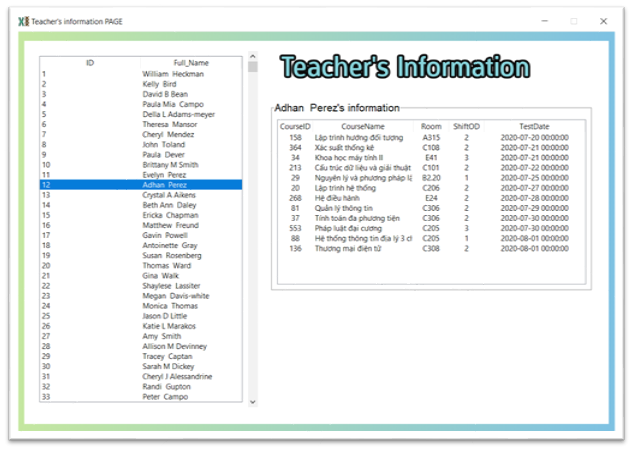


Figure 3.16 View teacher UI

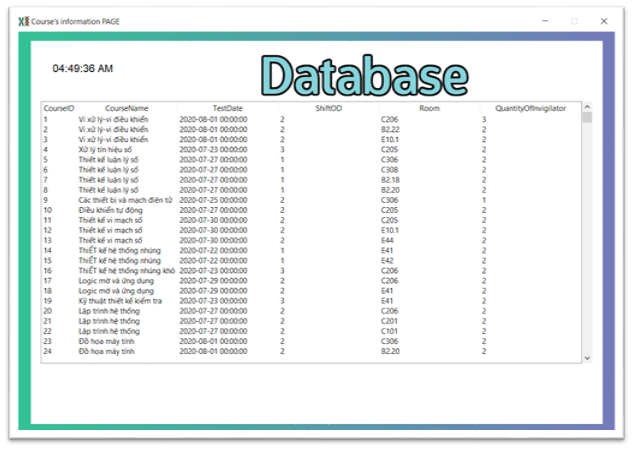


Figure 3.17 View exam schedule UI



Figure 3.18 Genetic algorithm UI

# **Part 4: Summary**

During all the time working at the Information System Laboratory, I have many chances to learn new things, chances to improve my knowledges, skills, and chances to learn some working experiences.

## **4.1 Knowledge**

* Learn more about Optimization algorithms and its applications, especially Genetic algorithm.
* Learn new programing language: Python
* Consolidate knowledge about application programming.
* Improve programing skills.
* Approach, research, and work with Tkinter, Pandas and NumPy.

## **4.2 Skills**

* Improve self-learning ability and learn new technology.
* Improve analysis problem, error detection and handling skill in coding process.
* Increase the ability to communicate when working in groups.
* Chances to work in a professional, dynamic environment.
* Have a basic understanding of the actual project workflow.

# 

# **Part 5: References**

[1] "Adaptive Learning: Fly the Brainy Skies." Wired, vol.10, no.3 (March 2002).

Introduction to genetic algorithm: <https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3>

Jupyter NoteBook: <https://jupyter.org/>

NumPy: <https://numpy.org/>

Python: <https://www.python.org/doc/>

PyCharm: <https://www.jetbrains.com/pycharm/>

Pandas: <https://pandas.pydata.org/>

Tkinter:

<https://docs.python.org/3/library/tkinter.html>

<https://www.tutorialspoint.com/python/python_gui_programming.htm>