



CODE CHALLENGE
CHAMPIONSHIP

PYTHON TRACK

RULES & GUIDELINES

2026



pythonTM



Table of Contents

1. Introduction	1
1.1. What's Python?	1
1.2. Why learn Python?	1
1.3. Does that track suit you?	1
1.4. Track Guidelines	2
2. Categories	2
2.1. Junior Category	2
2.2. Senior Category	3
3. Judging Criteria	3
3.1. Code Completeness & Correctness	3
3.2. Code Readability & Organization	3
3.3. Judging & Code Explanation	3
3.4. Bonus Part	4

1. Introduction

1.1. What's Python?

Python is one of the **easiest programming languages** as it is quite close to our natural language. It's widely used all over the world and considered to be the most popular programming language due to its **huge community** as well as it's well recognized in all problem-solving competitions, and used in many programming specializations including automation, AI, Data Science, creating websites and games, and many more.

1.2. Why learn Python?

- ✓ Easy syntax and beginner-friendly programming language
- ✓ Used in almost all programming specializations
- ✓ The top programming language for AI
- ✓ Strong support and huge community all over the world
- ✓ High demand in the job market with high salaries

1.3. Does that track suit you?

If you are interested in learning how to solve real-world industrial challenges, elevating your problem-solving skills, or entering the world of AI in the future. This will be the ideal track for you to start your unique professional journey!

1.4. Track Guidelines

1. You can use any code editor to write your Python code, but we recommend VS Code.
 2. Using AI coding assistant extensions (eg. GitHub Copilot) are not allowed.
 3. Teams can use any Python built-in functions to aid them in solving the challenges.
 4. Copying code from external sources or using pre-written solutions is strictly prohibited and will lead to team disqualification.
-

2. Categories

New update: All missions will require **GUI output** rather than CLI output as illustrated below.

2.1. Junior Category

This category is designed for students aged **10–13** years. Topics covered are:

Core Python Requirements:

- Basics → (Variables, comments, type casting)
- Data types → (integers, floats, booleans, strings, lists, dictionaries)
- Operations → (Arithmetic, Comparison, Logical, Membership, Assignment)
- Conditional statements → (if, elif, else, match statements)
- For and while loops
- Indexing & slicing
- User-defined functions
- Context managers (with statement)
- Random module → (random, randint, choice)
- String module → (ascii_letters, lowercase, uppercase, digits, punctuation)
- Datetime module → (creating dates, today, extracting day or month or year from dates)

GUI Development (Streamlit) Requirements:

- Displaying text functions → (write, title, header, subheader, code, divider)
- Status elements → (info, success, error, warning, spinner)
- Input widgets → (number_input, text_input, date_input, button, slider, selectbox)
- Layouts & containers → (forms, columns, sidebar, tabs)



2.2. Senior Category

This category is designed for students aged **14–17** years. Topics covered are:

Core Python Requirements:

- All topics covered in the junior category
- Exception handling → (try-except blocks)
- Statistics module → (mean, median, mode, quantiles)
- Pandas library → (reading csv, excel files; creating dataframes)
- google-genai library → (Configuring API keys, choosing models, generating responses)

GUI Development (Streamlit) Requirements:

- All topics covered in the junior category
- Multi-pages development → (st.Page, st.navigation)
- Chat elements → (chat_input, chat_message)
- Chart elements → (line chart, bar chart, area chart, dataframe)
- Custom color themes using config.toml

3. Judging Criteria

3.1. Code Completeness & Correctness

This section is worth **50 points** and aims to evaluate how effectively your code solves the given problem based on 2 questions as follows:

1. Does the code solve the provided problem? (40 points)

- Our scoring system will generate 8 input and output test cases then run the team's code to compare the output of their code with the expected output.
- If the expected output matches the team's output, the team will receive 5 points for each correct match.

2. Does the code handle potential runtime errors? (10 points)

- Our scoring system will examine the code carefully to see if it contains any runtime errors.
- For each potential runtime error found as "ValueError", "ZeroDivisionError", etc. The score will be decreased by 2.

3.2. Code Readability & Organization

This section is worth **15 points** and assesses the quality and cleanliness of your code based on the following 3 questions:

1. Are well-named (variables, functions, etc.) used? (5 points)

- 
- Our scoring system will examine the names of variables, functions, classes (if exist) to see whether they are descriptive and relative to the proposed mission.
 - For each non-descriptive or non-meaningful name found, the score will be decreased by 1.
- 2. Contain comments for clarity and explanation? (5 points)**
- Our scoring system will search for the team's code for comments and function docstrings (if a function exists) that must describe the code written.
 - For each 5-8 lines, a comment must be found explaining what's happening, otherwise, the score will be decreased by 1.
- 3. Is the code following PEP8, PEP20 principles? (5 points)**
- Our scoring system will check if the code obeys PEP8 and PEP20 guidelines that improve code readability including:
 - Limiting lines to 79 characters for code and 72 characters for comments.
 - Writing comments that explain why it is done, not just what it does.
 - If the implementation is hard to explain, it's a bad idea.
 - Using 4 spaces per indentation level.
 - Flat is better than nested.
 - For each guideline not followed, the score will be decreased by 1.

3.3. Judging & Code Explanation

This section is worth **25 points** and will be evaluated by a panel of 2 experienced judges based on the following 2 questions:

- 1. Is the team able to explain their provided solution? (10 points)**
- Judges will evaluate the explanation of the team to the solution provided to ensure they understand every part of it.
 - Each judge will give the team a score of 10 then the final score will be the average of the 2 judges' scores.
- 2. Is the team able to answer the judges' questions? (15 points)**
- Judges will ask 3 technical questions relevant to the mission provided.
 - Each correct answer, the team will get 5 points.

3.4. Bonus Part

This section is worth **10 points** and will be evaluated as follows:

- 1. Did the team make the bonus part of the mission? (10 points)**
- Our scoring system will examine whether the team made the bonus part correctly or not.
 - If they solve the bonus part correctly, they will receive 10 points, otherwise, they will get 0.



**CODE CHALLENGE
CHAMPIONSHIP**

**Click On Logos
And Follow Us**



Code Challenge Championship

**Join our WhatsApp Channel
to receive all updates.**



Channel



+ 2 01500 333 864



codechallenge-championship.org



info@codechallenge-championship.org