

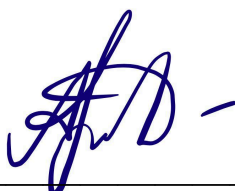
NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science
Bachelor's Programme "Applied Mathematics and Informatics"

Software Project Report on the Topic:
Development of Materials for Teaching Programming in Python

Fulfilled by:

Student of the Group
БПА/212
Dekkusheva Alina
Vladimirovna



(signature)

May 31st, 2024
(date)

Checked by the Project Supervisor:

Voznesenskaya Tamara Vasilievna
Associate Professor
Big Data and Information Retrieval School, Faculty of CS, HSE University

(signature)

June 2nd, 2024
(date)

Moscow 2024

Table of contents

Table of contents.....	2
Abstract.....	3
Introduction.....	4
Relevance.....	5
Expected outcomes.....	6
Instruments used.....	6
Subject area.....	7
General process.....	8
Basic Terms and Definitions.....	9
Working process.....	11
Comparative analysis of sources and analogues.....	18
Results and conclusions.....	21
Main results.....	21
Prospects of further work.....	22
Conclusions.....	23
List of sources.....	24

Abstract

This project report details the development of educational materials for teaching Python programming to students. The materials include instructional content and interactive Jupyter Notebooks covering key topics such as NumPy, strings, and lists. Emphasizing practical, hands-on learning, the project incorporates engaging exercises and Yandex Contest modules.

Informed by a thorough literature review and under supervision, the materials adopt best educational practices.

Key outcomes include the creation of comprehensive educational content, interactive learning modules, and increased student engagement. Future work includes adding multilingual support. In addition, continuous updates based on feedback and Python developments will keep the materials relevant and effective.

This project enriches educational resources for teaching Python, providing a foundation for developing essential programming skills.

Introduction

In the ever-evolving landscape of software development, the ability to program in multiple languages is becoming more and more of a necessity. Python is one of the easiest programming languages to learn given its straightforward approach and versatility.

The project aims to teach students the crucial skills and confidence to tackle software development and data collection and analysis. With plans to engage students through introductory Python programming educational resources and application guides.

This project aims to provide the Python programming language with a set of educational materials which will guide beginner programmers from their very first steps through to the level of high skills. It also aims to provide a comprehensive set of materials for learners in all phases.

Relevance

The development of educational materials for teaching Python programming is highly relevant in today's world due to several key factors:

1. **Growing demand for programming skills:** The need for programming knowledge is expanding rapidly across many industries. At the same time, Python, due to its versatility, is a language of choice for both the beginners and professional workers.
2. **Interactive learning:** The project emphasizes interactive learning through Jupyter Notebooks, which allows students to engage actively with the material. This method enhances understanding of complex concepts.
3. **Support for diverse learning outcomes:** The detailed plan for future work demonstrates a commitment to structured and phased learning outcomes. This approach ensures that students can progress from basic to advanced levels in a systematic manner.

Expected outcomes

The expected product of this project is a set of educational material on Python programming on the following topics:

1. Lists
2. Strings
3. Numpy

Instruments used

During the production, the following tools were used:

1. Jupyter notebook
2. Yandex Contest
3. Google Drive
4. Google Slides
5. Google Docs
6. GitHub

Subject area

The subject area of this project is conjugate with the development of educational resources for teaching Python programming: the creation of comprehensive instructional materials and interactive learning modules dedicated to beginner and intermediate learners. It is an intersection of a range of areas, the key ones of which include:

1. **Python programming:** The project introduces Python's fundamental concepts and syntax to new learners, along with a more complex topic NumPy.
2. **Data structures:** The project covers essential data structures in Python: strings and lists. These are fundamentals in programming which enable efficient data storage and manipulation. Understanding these data structures is crucial for writing efficient and effective code.
3. **NumPy:** a powerful library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. This is particularly relevant for students interested in data science, as NumPy is a foundational tool for data analysis and computational tasks.
4. **Interactive learning:** A significant aspect of the project is the use of Jupyter Notebooks to create an interactive learning environment. This interactive format helps students better understand.
5. **Educational pedagogy:** The project integrates modern educational techniques, such as step-by-step tutorials and coding exercises.

General process

The development of the educational materials followed a structured and systematic approach to ensure thoroughness and effectiveness. Let me outline the key stages:

1. Initial planning and research:

- Conducted a comprehensive review to identify existing Python programming courses and educational best practices.
- Defined clear learning outcomes and objectives for each topic.

2. Content development:

- Created detailed instructional materials for each topic (NumPy, strings, lists). This included writing explanations, designing examples, and developing interactive exercises.
- Ensured that the content was structured progressively.

3. Interactive Jupyter Notebooks/Yandex Contest:

- Developed Jupyter Notebooks for each topic, incorporating the instructional content and interactive exercises.
- Developed Yandex Contests for topics.

Basic Terms and Definitions

- **Python** - high-level, general-purpose programming language known for its readability and versatility. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.
- **Data type** - a classification which specifies the type of value a variable holds, such as integers, strings, or lists.
- **Function** - a block of reusable code designed to perform a specific task.
- **Method** - function that is associated with an object, it is defined within a class.
- **Immutability** - property of certain objects that prevents their state from being modified after they are created. Immutable objects include types such as strings, tuples, and integers.
- **Concatenation** - the process of joining two or more strings to form a new string.
- **Indexing** - accessing individual elements of a sequence (string, list, or tuple) using their position (index) in the sequence.
- **Unicode** - a standardized character encoding system designed to support text representation and manipulation.
- **Slicing** - extracting a subset of elements from a sequence (list, string, or tuple) using a specific range of indices.
- **Case sensitivity** - the language distinguishes between uppercase and lowercase letters.
- **Regular expressions** - sequences of characters that define a search pattern, primarily used for string matching and manipulation.
- **Encoding and decoding** - processes of converting data from one format to another.

- **Dynamic datatype** - nature where the type of a variable is determined at runtime rather than at compile-time.
- **Iteration** - process of executing a set of statements repeatedly, typically using loops.
- **Stack** - a data structure that follows the Last In, First Out (LIFO) principle, where the last element added to the stack is the first one to be removed.
- **Queue** - a data structure that follows the First In, First Out (FIFO) principle, where the first element added to the queue is the first one to be removed.
- **Attribute** - a value or a method associated with an object. Attributes are used to store data or functionalities related to an object
- **Aggregation functions** - functions that perform calculations on multiple values to return a single summarizing value.
- **Library** - a collection of modules that provide implementations for many tasks, such as web programming, date and time manipulation, and mathematical operations. Libraries are used to extend Python's functionality.
- **NumPy** - library for Python, providing support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.
- **Interactive learning** - incorporate interactive elements such as coding exercises and projects. This approach encourages active learning and helps students solidify their understanding.
- **Student guideline** - a set of instructions and rules provided to learners to help navigate their educational course.

Working process

The work on the project, as mentioned above, consisted of several major stages with multiple substeps on each of them. The particular tasks done by me were as follows:

1. Literature review and analysis:

- Conduct a review of existing Python programming courses and educational materials (see ch. Comparative analysis of sources and analogues)
- Identify best practices and effective teaching techniques used in similar courses.

2. Define learning outcomes:

- Establish clear learning outcomes for each topic.
- Define specific skills and knowledge students should acquire upon completing each module (NumPy, Strings, Lists).

3. Develop educational content for NumPy:

- Create detailed instructional materials covering:
 - array creation
 - array attributes
 - array indexing
 - array slicing
 - array mathematics
 - aggregation functions
 - random number generation
 - linear algebra operations
 - reshaping and transposing
- Include practical exercises within the Jupyter Notebook.

4. Develop educational content for Strings:

- Develop educational content for this topic including:

- basic properties
- inner implementation
- slices
- in-built methods
- regular expressions
- encoding and decoding
- Include practical exercises within Jupyter Notebook and Yandex Contest

A. Head over heels

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

You have been given a word. Make its last letter the first. All the subsequent letters will be shifted to the right by one position in this case.

Input format

A line consisting of one word (a sequence of lowercase Latin letters without spaces) of no more than 200 characters.

Output format

Output the resulting string

Picture 1. “Head over heels” task.

B. А роза упала на лапу Азора...

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

Write a program that would determine whether the entered word is a palindrome. Please note that your programme must be case-insensitive.

Input format

The input to the program is a string of no more than 20 characters, containing no characters other than numbers and letters (both uppercase and lowercase).

Output format

If the input string is a palindrome, output "YES". Otherwise, output "NO".

Picture 2. “А роза упала на лапу Азора” / “Stressed desserts” task.

C. IP?

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

In order to access the Internet, each computer is assigned a so-called IP address. It consists of four integers ranging from 0 to 255, separated by points.

Write a programme that would determine whether a given string is a valid IP address.

Input format

The input to the programme is a string of no more than 20 characters.

Output format

If the input string is a correct IP-address, output "YES". Otherwise, output "NO".

Picture 3. "IP?" task.

D. University of Mars

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

The National University of Mars has introduced additional standards for academic writing. Since this moment, all the students in their course works must write the numbers from 0 to 9 (1-digit non-negative) as words. (E.g. "one" instead of "1").

Write the programme that will help a student to rewrite his course paper so that it satisfies this novelty.

It is guaranteed that the paper does not contain any other numbers except for those on the interval [0; 9]

Picture 4. "University of Mars" task.

5. Develop educational content for Lists:

- Design instructional modules for understanding and using lists in Python, including:
 - basic properties
 - inner implementation
 - slices
 - in-built methods
 - multi-dimensional spaces
 - list as stack and queue
 - iterations
- Include practical exercises within the Jupyter Notebook and Yandex Contest

A. Interval

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

You are given an array A of indefinite length and two numbers - i and j . It is necessary to display all elements of the array from the i -th to the j -th (both inclusive).

Input format

The first line of the input file contains an array A , the elements of which are separated by spaces. The second line contains 2 numbers separated by a space - i and j .

Output format

Print all elements of the array from the i -th to the j -th inclusive. If this cannot be done, do not output "NO".

Picture 5. "Interval" task.

B. Grasshopper

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

You are given an array of indefinite length and a natural number N . Output all the elements of the array whose indices are a multiple of N .

Input format

The first line contains an array whose elements are separated by spaces. The second line contains the number N not greater than 10.

Output format

Print the corresponding array elements separated by spaces

Picture 6. “Grasshopper” task.

C. Pinta

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

Write a programme that would determine whether the sum of the squares of all the elements of an array of size N is a 5-digit number.

Input format

The first line of the input file contains a natural number N — the number of elements in the array. The next line contains N integers separated by spaces - these are array elements.

Output format

If the input data satisfies the condition of the problem, the programme should output "YES", otherwise "NO".

Picture 7. “Pinta” task.

D. Competition

Time limit	1 second
Memory limit	64.0 MB
Input	stdin or input.txt
Output	stdout or output.txt

Once upon a time, a competition was held at a school for young programmers. Now, it's time to see the results.

Each participant has his own identification number assigned and also a certain number of points he/she managed to gain. Display the tournament table - a list of competition participants sorted by the number of points scored.

Input format

The first line contains the number n - the number of participants. Each next line contains a pair of numbers separated by a space: the number of points scored and the identification number of the corresponding participant. All numbers in the input file are non-negative integers and do not exceed 1000.

Output format

Output the original list in descending order of points. If some participants have the same points, then they need to be sorted among themselves by the identification number in descending order.

Picture 8. "Competition" task.

Comparative analysis of sources and analogues

In developing the educational materials for teaching Python programming, it is essential to compare our initiative with the already-existing similar resources and materials. This comparative analysis helps identify strengths, weaknesses, and unique contributions of our project.

1. Codim (Кодим)

Codim is an online platform on coding courses for school students, which also offers Python.

Structure:

- **Basic Python:** Covers variables, data types, and simple operations.
- **Advanced Python:** Introduces functions, loops, and complex data structures.
- **Project-based Learning:** Includes practical projects such as game development and data analysis.

Comparison with our case:

- **Strengths:** Codim's project-based approach and real-time feedback system are highly effective in engaging students and reinforcing learning.
- **Weaknesses:** The content is less comprehensive in advanced topics compared to our materials. Our project offers more in-depth coverage of libraries like NumPy, which are crucial for data science applications.

2. Yandex Lyceum (Яндекс Лицей)

Overview: Yandex Lyceum provides a two-year educational program focused on Python programming for high school students.

Structure:

- **First Year:** Basics of Python: syntax, basic operations, and simple algorithms.
- **Second Year:** Advanced topics such as web development, data structures, and algorithmic thinking.

Comparison with our case:

- **Strengths:** Due to its long-term program it allows for deep learning and sustained engagement.
- **Weaknesses:** While comprehensive, the program's length and structure may not be as flexible for students looking for shorter, more intensive courses. Our project's approach with Jupyter Notebooks offers more flexibility and immediate application of concepts.

3. Algorithmics (Алгоритмика)

Overview: Algorithmics is an international coding school that originated in Russia, offering courses for children and teenagers. Their Python courses are tailored to different age groups and skill levels. This school uses game-based learning which attracts students.

Structure:

- **Beginners:** Introduction to Python, basic commands, and simple projects.
- **Intermediate:** More complex projects, including game development and basic data analysis.
- **Advanced:** Preparation for programming competitions, advanced algorithms, and data structures.

Comparison with our case:

- **Strengths:** Algorithmics' game-based learning is very effective in maintaining student interest and motivation.
- **Weaknesses:** Their focus on younger audiences might mean less depth in advanced topics. Our project targets a broader age range and provides a more thorough exploration.

Results and conclusions

Main results

The project includes the results in the development of educational materials for teaching Python programming, focusing on topics NumPy, strings, and lists.

The resulting products may be summarized as follows:

1. **NumPy**: A detailed notebook was created, covering fundamental concepts such as array creation, array attributes, indexing, slicing, mathematical operations, etc. Another notebook on this topic was created to provide students with the possibility to practice their knowledge and skills.
2. **Strings**: A notebook that explores the properties and manipulation of strings in Python. It covers basic operations, slicing, in-built methods, regular expressions, and encoding/decoding, with practical exercises embedded. The Yandex Contest on this topic for interactive usage was also created.
3. **Lists**: A comprehensive notebook focusing on the properties and uses of lists, including slicing, built-in methods, multi-dimensional lists, stacks, queues, and iterations, complemented by interactive coding tasks. The Yandex Contest on this topic for interactive usage was also prepared.

Prospects of further work

While the project has successfully developed a comprehensive set of educational materials for teaching Python programming, there are several prospects for future work:

1. Advanced topics and specialized modules:

- Develop additional modules covering advanced Python topics such as machine learning, data visualization and web development.

2. Enhanced interactive features:

- Implement automated grading of exercises, personalized feedback.
- Explore the use of interactive visualization tools and dashboards for a better understanding.

3. Multilingual support:

- Provide students with the opportunity to choose the language in studying the materials.

4. Continuous improvement based on feedback and Python updates:

- Regularly update the materials based on ongoing feedback from students.
- Update the materials based on the Python new versions.

Conclusions

The development of comprehensive educational materials for teaching Python programming is the main achievement. The project has created detailed instructional content, interactive Jupyter Notebooks, and Yandex Contests, providing an engaging learning experience.

The project successfully met its objective. The project emphasized practical learning through hands-on exercises and interactive coding tasks. The prospects for future work include expanding the contents by translating and updating the materials. The project has made a valuable contribution to the available educational resources for teaching Python programming.

List of sources

1. Python Software Foundation. Python Documentation:
<https://docs.python.org/>
2. Codim (Кодим):
<https://codim.ru/>
3. Yandex Lyceum (Яндекс Лицей):
<https://yandexlyceum.ru/>
4. Algorithmics (Алгоритмика):
<https://algorithmicschool.com/>
5. Jupyter Project. Project Jupyter:
<https://jupyter.org/>
6. String Documentation:
<https://docs.python.org/3/library/string.html>
7. Lists Documentation:
<https://docs.python.org/3/tutorial/datastructures.html>
8. NumPy Developers. NumPy Documentation:
<https://numpy.org/doc/>