Python Course
Test Representation

International Programming School Algorithmics

Course Instructor:

Associate Engineer, Andrii Voznesenskyi

Mathematics and Information Systems Faculty
Warsaw University of Technologies

May 20, 2023

Restrictions:

This document is intended for use only by individuals associated with Algorithmics and the author. Unauthorized distribution or sharing of this document, in whole or in part, is strictly prohibited.

Copyright Information:

Purpose:

This document is intended for the exclusive use of colleagues at Algorithmics International School or students seeking to enhance their individual understanding of the course and engage in self-education. It is of utmost importance to strictly adhere to the restrictions and copyright information provided within this document.

”Education is the kindling of a flame, not the filling of a vessel.”
Socrates, Ancient Greek philosopher

Зміст

1 Tasks S

1.1 Task S1

1.1.1 Task 1 (10 points)

Write a function `construct_course_list()` that constructs and returns a list of MIT classes in the following format:

- Course 1 - Civil and Environmental Engineering
- Course 2 - Mechanical Engineering
- Course 3 - Materials Science and Engineering
- Course 4 - Architecture
- Course 5 - Chemistry
- Course 6 - Electrical Engineering and Computer Science
- Course 7 - Biology
- Course 8 - Physics
- Course 9 - Brain and Cognitive Sciences
- Course 10 - Chemical Engineering

Returns:

- `course_list` (list): A list of strings representing the MIT course names.

Example:

```
1 construct_course_list()
```

This will return the following list:

```
[
"Course 1 - Civil and Environmental Engineering",
"Course 2 - Mechanical Engineering",
"Course 3 - Materials Science and Engineering",
"Course 4 - Architecture",
"Course 5 - Chemistry",
"Course 6 - Electrical Engineering and Computer Science",
"Course 7 - Biology",
"Course 8 - Physics",
"Course 9 - Brain and Cognitive Sciences",
"Course 10 - Chemical Engineering"
]
```

1.1.2 Task 2 (10 points)

Write a function `get_course_name(number)` that takes a number as input and returns the name of the MIT course corresponding to the given number.

Function Input:

- `number` (int): The course number to retrieve the name for.

Returns:

- `course_name` (str): The name of the MIT course corresponding to the given number.

Example:

```
get_course_name(1)
```

This will return the following string:

```
"Course 1 - Civil and Environmental Engineering"
```

1.1.3 Task 3 (10 points)

Create a function that takes a day of the year (an integer from 1 to 365) as input and returns the corresponding month. You can assume a non-leap year. For simplicity, you can consider each month to have a fixed number of days as it has in the callendar (January: 31 days, February: 28 days(you has to consider this number in your solution) and 29 in every leap year, March: 31 days, April: 30 days, May: 31 days, June: 30 days, July: 31 days, August: 31 days, September: 30 days, October: 31 days, November: 30 days, December: 31 days,). The function should return the month as a string. You can choose the month names as per your preference. **Example:**

```
print(get_month(75))  # Output: March
```

1.1.4 Task 4 (10 points)

Create a function `perform_operation(a, b, c=0, d=0, e=0, f=0, g=0, h=0, i=0, operator='+')` that takes two numbers (`a` and `b`) and up to eight additional numbers (`c` to `i`) as arguments. The function should also accept an operator (`+`, `-`, `*`, `/`) as a keyword argument. The function should perform the corresponding operation on all the provided numbers and return the result. If the operator is division (`/`) and the second number is zero, the function should return an error message.

Function Inputs:

- `a` (numeric): The first number.
- `b` (numeric): The second number.
- `c` to `i` (numeric, optional): Up to eight additional numbers.
- `operator` (str, optional): The operator to perform the operation. Default is `'+'`.

Returns:

- `result` (numeric or str): The result of the operation. If the operator is division (/) and the second number is zero, return the error message: `"Error: Division by zero"`.

Example:

```
print(perform_operation(5, 3, 2, 4, operator='+')) # Output: 14
print(perform_operation(10, 2, 3, operator='*')) # Output: 60
print(perform_operation(5, 0, operator='/')) # Output: "Error: Division by zero"
```

Note: The specific implementation details and variable names may vary.

1.1.5 Task 5 (10 points)

Consider the fraction $\frac{a}{b} = \frac{nominator}{denominator}$, where a represents the numerator and b represents the denominator. In this task, ask the user to input the numerator and denominator values. The program should calculate the result of the division and (if possible) the remainder of the division operation.

Create a function `divide_fraction(numerator, denominator)` that takes the numerator and denominator as input and returns the division result and remainder (if applicable).

Within the function, create a variable `i` and set it equal to `numerator`, and create another variable `answer` and set it to 0.

Use a while loop to iterate while `i` is greater than 0. In each iteration, subtract the `denominator` from `i` and increment `answer` by 1.

If `i` becomes 0 after the loop, return a string in the format: "Result: `answer`".

If `i` is not 0, return a string in the format: "Result: `answer-1` \n Reminder: `i+denominator`".

Feel free to choose your own variable names for this task.

Function Inputs:

- `numerator` (int): The numerator of the fraction.
- `denominator` (int): The denominator of the fraction.

Returns:

- `result` (str): A string containing the division result and remainder (if applicable) in the format mentioned above.

Example:

```
divide_fraction(7, 3)
```

This will return the following string:

```
"Result: 2
 Reminder: 1"
```

Maybe the usage of the `\n` will help.

Note: The specific implementation details and variable names may vary.

1.2 Task S2

1.2.1 Task 1 (10 points)

Write a function `calculate_the_sum_of_n_numbers(n)` which calculates the sum of n numbers, where n is a natural finite number.
Example:

```
calculate_the_sum_of_n_numbers(100)
```

This will be used to calculate the sum $1 + 2 + 3 + \cdots + 100 = \frac{100*101}{2} = 50 * 101 = 5050$.

1.2.2 Task 2 (10 points)

Write a function `nums_to_n(n)` to print all the odd numbers from 1 to n, where n is a natural finite number, and if the number is divisible by 5, the function should draw the user's attention to it. The function does not return anything, it prints all odd numbers from 1 to n.
Example:

```
nums_to_n(10)
```

This code should print:

```
1;
3;
5 is divisible by 5;
7;
9;
```

Remember to include a semicolon at the end of each line!

1.2.3 Task 3 (10 points)

Write a function named `analyse_the_number(x, a_less, b_greater)` which will:

- Inform the user if the number is odd or even;
- Inform the user if the number is greater than or equal to `"b_greater"`;
- Inform the user if the number is less than or equal to `"a_less"`;
- Tell the user what is the factorial of x: if $x = 6$, it will print $6 * 5 * 4 * 3 * 2 * 1$;
- Print the number$(x^{b_greater})^{a_less}$;
- If the number is from the set $[-2, 2]$ (the number x may be 2, -1, 0, 1, 2), the function will print the number as follows: if the number x is 2, the program will print: "two".

1.2.4 Task 4 (10 points)

Solve the following system of linear equations:

$$2x + 3y = 7$$
$$4x - 2y = 10$$

Write a function `solve_system(a1, b1, c1, a2, b2, c2)` that takes the coefficients of two linear equations of the form $a_1x + b_1y = c_1$ and $a_2x + b_2y = c_2$ as inputs and solves the system of equations. The function should return the solution as a tuple (x, y) representing the values of x and y that satisfy both equations. If the system of equations has no solution or infinite solutions, return "No unique solution".

Example:

```
1 solve_system(2, 3, 7, 4, -2, 10) # Output: (2.0, 1.0)
```

```
1 solve_system(1, -2, 3, 2, -4, 6) # Output: "No unique
    solution"
```

```
1 solve_system(1, 2, 3, 2, 4, 6) # Output: "No unique solution"
```

```
1 solve_system(0, 0, 0, 0, 0, 0) # Output: "No unique solution"
```

Note: Show all the necessary steps and explanations to justify your solutions. You may use any appropriate methods or techniques taught in secondary school mathematics.

1.2.5 Task 5 (10 points)

Write a Python function `is_palindrome(s)` that takes a string `s` as input and returns `True` if the string is a palindrome and `False` otherwise. A palindrome is a word, phrase, number, or other sequence of characters that reads the same forward and backward, ignoring spaces, punctuation, and capitalization.

Example:

```
1 is_palindrome("racecar") # Output: True
```

```
1 is_palindrome("Hello, World!") # Output: False
```

Write the function `is_palindrome` to solve the task and test it with different strings.

1.3 Task S3

1.3.1 Task 1 (10 points)

Write a function `calculate_factorial(n)` that calculates and returns the factorial of a natural number n. This function should raise an exception if n is negative or if it's not an integer.

Example:

```
calculate_factorial(5) # Output: 120
```

This will return $5 * 4 * 3 * 2 * 1 = 120$.

1.3.2 Task 2 (10 points)

Write a function `print_fibonacci(n)` that prints the first n numbers in the Fibonacci sequence. The function should print each number on a new line. The function does not return anything, it only prints the Fibonacci numbers.

Example:

```
print_fibonacci(7)
```

This code should print:

```
0
1
1
2
3
5
8
```

1.3.3 Task 3 (10 points)

Write a function `is_prime(n)` that returns `True` if a number is prime and `False` otherwise. A prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself.

Example:

```
is_prime(7)
```

This will return `True` because 7 is a prime number.

1.3.4 Task 4 (10 points)

Write a function `find_palindromes(words)` that takes a list of words and returns a new list containing only the palindromes from the original list. A palindrome is a word that reads the same forwards and backwards. The function should ignore case sensitivity, meaning "Mom"and "mOm"should be considered palindromes.

Example:

```
words = ["level", "deed", "hello", "Madam", "world"]
find_palindromes(words)
```

This will return `["level "deed "Madam"]`, as these words are palindromes.

1.3.5 Task 5 (10 points)

Write a function `calculate_mean(numbers)` that takes a list of numbers and returns the mean (average) value. The function should return the mean as a floating-point number.

Example:

```
numbers = [1, 2, 3, 4, 5]
calculate_mean(numbers) # Output: 3.0
```

This will return the mean of the numbers in the list, which is 3.0.

1.4 Task S4

1.4.1 Task 1 (10 points)

Write a function `calculate_poly_function_val(a, b, c, d, e, k, g)` that calculates the value of a function $f(a,b,c,d,e,k,g) = 10a^3+11b^{10}+12c^3+3d^2+6e^{18}+67k^{12}+22g^4+\frac{127}{168}a^4+4e+\frac{6}{7}g^2+\sqrt[3]{\frac{3}{2}a^2}-\frac{2}{5}d+10e^2-\pi k+\frac{(c+d+k+g)a^2}{b}$ at the given point (a,b,c,d,e,k,g).

Function Inputs:

- `a` (float): The value of the variable a.
- `b` (float): The value of the variable b.
- `c` (float): The value of the variable c.
- `d` (float): The value of the variable d.
- `e` (float): The value of the variable e.
- `f` (float): The value of the variable k.
- `g` (float): The value of the variable g.

Returns:

- `result` (float): The calculated value of the function $f(a,b,c,d,e,k,g)$ at the given point.

Example:

```
calculate_poly_function_val(1.5, 2.3, -0.7, 4.2, 0.8, -1.1,
   3.6)
```

This will return the calculated value of the function at the given point.

Note: Make sure to handle any necessary mathematical operations correctly according to the specified function.

1.4.2 Task 2 (10 points)

Write a function `calculate_factorial(n)` that calculates and returns the factorial of a number n. The function should raise an exception if n is negative or not an integer.
Example:

```
calculate_factorial(5)
```

This function will return $5*4*3*2*1 = 120$.

1.4.3 Task 3 (10 points)

Write a function `calculate_combination(n, r)` that calculates and returns the combination of n items taken r at a time, where n and r are natural numbers and $r \leq n$. More presize information about the Newtonian binomial symbol defined to tel how many ways exists to choose r different items from n is below
Example:

```
calculate_combination(5, 2)
```

This function will return 10, as there are 10 ways to choose 2 items from 5.

The combination, denoted as $\binom{n}{r}$, represents the number of ways to choose r items from a set of n distinct items, without regard to their order. It can be calculated using the formula:

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

1.4.4 Task 4 (10 points)

Write a function `analyze_apple_weights(apple_weights)` that takes a list of apple weights as input and performs the following analysis:

Calculate the average weight of the apples. Determine the maximum weight among the apples. Determine the minimum weight among the apples.

The function should return a dictionary containing the analysis results.

Function Input:

- `apple_weights` (list): A list of floating-point numbers representing the weights of the apples.

Returns:

- `analysis_results` (dictionary): A dictionary containing the following analysis results: - `"average_weight"`: The average weight of the apples. - `"max_weight"`: The maximum weight among the apples. - `"min_weight"`: The minimum weight among the apples.

Example:

```
apple_weights = [0.2, 0.5, 0.3, 0.6, 0.4]
analyze_apple_weights(apple_weights)
```

This will return the following dictionary:

```
{
"average_weight": 0.4,
"max_weight": 0.6,
"min_weight": 0.2
}
```

Note: This simplified task focuses on calculating the average, maximum, and minimum weights of the apples in the dataset. It omits the additional analysis requirements mentioned in the original task.

1.4.5 Task 5 (10 points)

Consider the list of MIT classes in the following format:

- Course 1 - Civil and Environmental Engineering

- Course 2 - Mechanical Engineering
- Course 3 - Materials Science and Engineering
- Course 4 - Architecture
- Course 5 - Chemistry
- Course 6 - Electrical Engineering and Computer Science
- Course 7 - Biology
- Course 8 - Physics
- Course 9 - Brain and Cognitive Sciences
- Course 10 - Chemical Engineering

Write a function `get_course_name(number)` that takes a number as input and returns the name of the MIT course corresponding to the given number.

Function Input:

- `number` (int): The course number to retrieve the name for.

Returns:

- `course_name` (str): The name of the MIT course corresponding to the given number.

Example:

```
1 get_course_name(1)
```

This will return the following string:

```
"Course 1 - Civil and Environmental Engineering"
```

1.5 Task S5

1.5.1 Task 1 (10 points)

Create a function `perform_operation(a, b, c=0, d=0, e=0, f=0, g=0, h=0, i=0, operator='+')` that takes two numbers (`a` and `b`) and up to eight additional numbers (`c` to `i`) as arguments. The function should also accept an operator (`+`, `-`, `*`, `/`) as a keyword argument. The function should perform the corresponding operation on all the provided numbers and return the result. If the operator is division (`/`) and the second number is zero, the function should return an error message.

Function Inputs:

- `a` (numeric): The first number.
- `b` (numeric): The second number.
- `c` to `i` (numeric, optional): Up to eight additional numbers.
- `operator` (str, optional): The operator to perform the operation. Default is `'+'`.

Returns:

- `result` (numeric or str): The result of the operation. If the operator is division (`/`) and the second number is zero, return the error message: `"Error: Division by zero"`.

Example:

```
1 print(perform_operation(5, 3, 2, 4, operator='+')) # Output:
    14
2 print(perform_operation(10, 2, 3, operator='*')) # Output: 60
3 print(perform_operation(5, 0, operator='/')) # Output: "Error
    : Division by zero"
```

Note: The specific implementation details and variable names may vary.

1.5.2 Task 2 (10 points)

Write a Python function `fibonacci(n)` that calculates the nth Fibonacci number using recursion.

Example:

```
1 print(fibonacci(10))
```

This should print the 10th Fibonacci number.

1.5.3 Task 3 (10 points)

Let's calculate the value of the iterated integral of a differentiable and continuous function $f(x, y, x) = 1$:

$$\int_a^b \int_c^d \int_e^g 1 \, dx \, dy \, dz = (b-a)(d-c)(g-e)$$

Write a function `calculate_iterated_integral(a, b, c, d, e, g)` that calculates the value of the iterated integral of a differentiable and continuous function $f(x, y, z) = 1$.

The function should take the limits of integration aaa, bbb, ccc, ddd, eee, and ggg as inputs and return the calculated value of the iterated integral.

Function Inputs:

- `a` (numeric): The lower limit of integration for the variable xxx.
- `b` (numeric): The upper limit of integration for the variable xxx.
- `c` (numeric): The lower limit of integration for the variable yyy.
- `d` (numeric): The upper limit of integration for the variable yyy.
- `e` (numeric): The lower limit of integration for the variable zzz.
- `g` (numeric): The upper limit of integration for the variable zzz.

Returns:

- `result` (numeric): The calculated value of the iterated integral, which is equal to $(b-a)(d-c)(g-e)$.

Example:

```
result = calculate_iterated_integral(1, 3, 2, 4, 0, 5)
print(result) # Output: 36
```

Note: The specific implementation details, function name, and variable names may vary.

1.5.4 Task 4 (10 points)

Write a Python function `is_palindrome(s)` that takes a string `s` as input and returns `True` if the string is a palindrome and `False` otherwise. A palindrome is a word, phrase, number, or other sequence of characters that reads the same forward and backward, ignoring spaces, punctuation, and capitalization.

Example:

```
is_palindrome("racecar") # Output: True
```

```
is_palindrome("Hello, World!") # Output: False
```

Write the function `is_palindrome` to solve the task and test it with different strings.

1.5.5 Task 5 (10 points)

Create a function that takes a day of the year (an integer from 1 to 365) as input and returns the corresponding month. You can assume a non-leap year. For simplicity, you can consider each month to have a fixed number of days. The function should return the month as a string. You can choose the month names as per your preference. **Example:**

```
print(get_month(75))  # Output: March
```

2 Tasks P

2.1 Project 1: Turtle drawing

2.1.1 Description

The task of this program is to create an interactive drawing application using the turtle module in Python. The program allows the user to control a turtle object on the screen and perform various actions.

2.1.2 Specification

Here is a breakdown of the tasks performed by the program:

[label=0.]Set up the Turtle:

1.
   - Create a turtle object.
   - Set the turtle's speed to 100.
   - Set the turtle's initial color to red.
   - Set the turtle's width to 1.
   - Set the turtle's shape to "turtle".
   - Put the turtle's pen down to start drawing.

2. Define Color Changing Functions:
   - Implement `turtle_color_red()` to change the turtle's color to red.
   - Implement `turtle_color_green()` to change the turtle's color to green.

3. Define Mouse Event Function:
   - Implement `fxn(x, y)` to handle mouse drag events.
   - Stop backtracking of the turtle.
   - Adjust the turtle's angle and direction towards the new coordinates (x, y).
   - Move the turtle to the new coordinates (x, y).
   - Enable the function to be called again for further dragging.

4. Define Keyboard Event Functions:
   - Implement `move_forward()` to move the turtle forward by 50 units.
   - Implement `move_backward()` to move the turtle backward by 50 units.
   - Implement `turn_left()` to rotate the turtle left by 45 degrees.
   - Implement `turn_right()` to rotate the turtle right by 45 degrees.
   - Implement `fill_screen()` to fill the entire screen with color.

5. Set up Event Listeners:
   - Get the turtle's screen object.
   - Enable listening for key and mouse events.

- Register event handlers for specific keys and mouse clicks.
- When events occur, the corresponding functions are called to perform the desired actions.

6. Enter the Main Event Loop:

- Start the turtle's event loop.
- The program continuously listens for events and responds accordingly.
- The program remains interactive until the window is closed.

The main goal of this program is to provide an interactive drawing experience where the user can control the turtle's movement, change its color, and fill the screen with color. The program utilizes various event-driven functions to respond to user inputs and update the turtle's behavior on the screen.

2.2 Project 2: Number Guessing Game

2.2.1 Description

In this project, you will create a number guessing game. The program will generate a random number between a specified range, and the user will have to guess the number within a certain number of attempts. After each guess, the program will provide feedback to the user if the guess is too high or too low. The game will continue until the user guesses the correct number or runs out of attempts.

2.2.2 Specifications

- The program should generate a random number between a specified range.
- The user should be prompted to enter their guess.
- The program should provide feedback to the user if the guess is too high or too low.
- The program should keep track of the number of attempts.
- The game should continue until the user guesses the correct number or runs out of attempts.
- The program should display a message indicating whether the user won or lost the game.

2.3 Project 3: To-Do List

2.3.1 Description

In this project, you will create a simple to-do list application. The program will allow the user to add tasks, mark tasks as completed, and view the list of tasks. The tasks will be stored in memory while the program is running, and they will be lost once the program is closed.

2.3.2 Specifications

- The program should provide a menu with options to add a task, mark a task as completed, and view the list of tasks.
- The user should be able to enter the details of a task (e.g., task name, due date) when adding a task.
- The program should store the tasks in a list or data structure.
- The program should display the list of tasks with their details.
- The user should be able to mark a task as completed, which will update its status in the list.
- The program should handle invalid inputs and provide appropriate error messages.

2.4 Project 4: Simple Calculator

2.4.1 Description

In this project, you will create a simple calculator program. The program will prompt the user to enter two numbers and an operation (+, -, *, /), and it will perform the corresponding calculation and display the result.

2.4.2 Specifications

- The program should prompt the user to enter the first number.
- The program should prompt the user to enter the second number.
- The program should prompt the user to enter the operation (+, -, *, /).
- The program should perform the corresponding calculation based on the entered numbers and operation.
- The program should display the result of the calculation.
- The program should handle invalid inputs and provide appropriate error messages.

2.5 Project 5: Hangman Game

2.5.1 Description

In this project, you will create a Hangman game. The program will select a random word from a predefined list, and the user will have to guess the letters of the word one by one. The user will have a limited number of attempts, and the program will provide feedback on the correctness of each guess.

2.5.2 Specifications

- The program should select a random word from a predefined list of words.
- The program should display the initial state of the word with underscores for the unknown letters.
- The program should prompt the user to enter a letter guess.
- The program should check the correctness of the guess and update the state of the word accordingly.
- The program should display the updated state of the word with the correctly guessed letters.
- The program should keep track of the number of attempts and limit the guesses to a certain number.
- The program should display a message indicating whether the user won or lost the game.

3 Завдання S

3.1 Завдання S1

3.1.1 Завдання 1 (10 балів)

Напишіть функцію `construct_course_list()`, яка створює і повертає список предметів MIT у наступному форматі:

- Course 1 - Civil and Environmental Engineering
- Course 2 - Mechanical Engineering
- Course 3 - Materials Science and Engineering
- Course 4 - Architecture
- Course 5 - Chemistry
- Course 6 - Electrical Engineering and Computer Science
- Course 7 - Biology
- Course 8 - Physics
- Course 9 - Brain and Cognitive Sciences
- Course 10 - Chemical Engineering

Повертає:

- `course_list` (список): Список рядків, які представляють назви курсів MIT.

Приклад:

```
1 construct_course_list()
```

Це поверне наступний список:

```
[
"Course 1 - Civil and Environmental Engineering",
"Course 2 - Mechanical Engineering",
"Course 3 - Materials Science and Engineering",
"Course 4 - Architecture",
"Course 5 - Chemistry",
"Course 6 - Electrical Engineering and Computer Science",
"Course 7 - Biology",
"Course 8 - Physics",
"Course 9 - Brain and Cognitive Sciences",
"Course 10 - Chemical Engineering"
]
```

3.1.2 Завдання 2 (10 балів)

Напишіть функцію `get_course_name(number)`, яка приймає число як вхідні дані і повертає назву предмету MIT, що відповідає заданому номеру.

Вхідні дані:

- `number` (ціле число): Номер курсу, для якого потрібно отримати назву.

Повертає:

- `course_name` (рядок): Назва предмету MIT, що відповідає заданому номеру.

Приклад:

```
1 get_course_name(1)
```

Це поверне наступний рядок:

```
"Course 1 - Civil and Environmental Engineering"
```

3.1.3 Завдання 3 (10 балів)

Створіть функцію, яка приймає день року (ціле число від 1 до 365) як вхідні дані і повертає відповідний місяць. Ви можете припустити, що рік не є високосним. З метою спрощення, ви можете вважати, що кожен місяць має фіксовану кількість днів, як це вказано у календарі (Січень: 31 день, Лютий: 28 днів (ви повинні врахувати це число у своєму рішенні), Березень: 31 день, Квітень: 30 днів, Травень: 31 день, Червень: 30 днів, Липень: 31 день, Серпень: 31 день, Вересень: 30 днів, Жовтень: 31 день, Листопад: 30 днів, Грудень: 31 день). Функція повинна повертати місяць у вигляді рядка. Ви можете вибрати назви місяців за своїм бажанням. **Приклад:**

```
1 print(get_month(75)) #            : March
```

3.1.4 Завдання 4 (10 балів)

Створіть функцію `perform_operation(a, b, c=0, d=0, e=0, f=0, g=0, h=0, i=0, operator='+')`, яка приймає два числа (`a` і `b`) та до восьми додаткових чисел (`c` до `i`) як аргументи. Функція також повинна приймати оператор (`+`, `-`, `*`, `/`) в якості іменованого аргументу. Функція повинна виконувати відповідну операцію над усіма заданими числами і повертати результат. Якщо оператор - це ділення (`/`), а друге число - нуль, функція повинна повертати повідомлення про помилку.

Вхідні дані функції:

- `a` (числовий): Перше число.
- `b` (числовий): Друге число.
- `c` до `i` (числовий, необов'язковий): До восьми додаткових чисел.
- `operator` (рядок, необов'язковий): Оператор для виконання операції. За замовчуванням - `'+'`.

Повертає:

- `result` (числовий або рядок): Результат операції. Якщо оператор - це ділення (/), а друге число - нуль, повертається повідомлення про помилку: `"Error: Division by zero"`.

Приклад:

```
print(perform_operation(5, 3, 2, 4, operator='+')) #                : 14
print(perform_operation(10, 2, 3, operator='*')) #    : 60
print(perform_operation(5, 0, operator='/')) #                : "Error: Division by zero"
```

Примітка: Конкретні деталі реалізації та назви змінних можуть варіюватися.

3.1.5 Завдання 5 (10 балів)

Розглянемо дріб $\frac{a}{b} =$, де a позначає чисельник, а b позначає знаменник. У цьому завданні попросіть користувача ввести значення чисельника та знаменника. Програма повинна обчислити результат ділення та (за можливості) залишок від операції ділення.

Створіть функцію `divide_fraction(numerator, denominator)`, яка приймає чисельник та знаменник як вхідні дані і повертає результат ділення та залишок (якщо є).

Усередині функції створіть змінну `i` і прирівняйте її до `numerator`, а також створіть іншу змінну `answer` і прирівняйте її до 0.

Використовуйте цикл `while`, щоб ітерувати, доки `i` більше 0. На кожній ітерації віднімайте від `i` значення `denominator` і збільшуйте `answer` на 1.

Якщо `i` стає рівним 0 після циклу, поверніть рядок у форматі: "Результат: `answer`".

Якщо `i` не дорівнює 0, поверніть рядок у форматі: "Результат: `answer-1` \n Залишок: `i+denominator`".

Не соромтеся обрати власні назви змінних для цього завдання.

Вхідні дані функції:

- `numerator` (ціле число): Чисельник дробу.
- `denominator` (ціле число): Знаменник дробу.

Повернення:

- `result` (рядок): Рядок, що містить результат ділення та залишок (якщо є) у форматі, зазначеному вище.

Приклад:

```
divide_fraction(7, 3)
```

Це поверне наступний рядок:

```
"Result: 2
 Reminder: 1"
```

Можливо, використання `\n` буде корисним.

Примітка: Конкретні деталі реалізації та назви змінних можуть варіюватися.

3.2 Завдання S2

3.2.1 Завдання 1 (10 балів)

Напишіть функцію `calculate_the_sum_of_n_numbers(n)`, яка обчислює суму n чисел, де n є природним скінченним числом. **Приклад:**

```
1 calculate_the_sum_of_n_numbers(100)
```

Це буде використовуватись для обчислення суми $1 + 2 + 3 + \cdots + 100 = \frac{100101}{2} = 50101 = 5050$.

3.2.2 Завдання 2 (10 балів)

Напишіть функцію `nums_to_n(n)`, яка виводить всі непарні числа від 1 до n, де n є природним скінченним числом, а якщо число ділиться на 5, функція повинна звернути увагу користувача на це. Функція нічого не повертає, вона лише виводить всі непарні числа від 1 до n. **Приклад:**

```
1 nums_to_n(10)
```

Цей код повинен вивести:

```
1;
3;
5 ділиться на 5;
7;
9;
```

Не забудьте додати крапку з комою в кінці кожного рядка!

3.2.3 Завдання 3 (10 балів)

Напишіть функцію з назвою `analyse_the_number(x, a_less, b_greater)`, яка буде:

- Повідомляти користувачу, чи число є непарним чи парним;
- Повідомляти користувачу, чи число більше або дорівнює `"b_greater"`;
- Повідомляти користувачу, чи число менше або дорівнює `"a_less"`;
- Повідомляти користувачу, який факторіал має число x: якщо $x = 6$, воно виведе $65432 * 1$;
- Виводити число $(x^{b\_greater})^{a\_less}$;
- Якщо число належить множині $[-2, 2]$ (число x може бути 2, -1, 0, 1, 2), функція буде виводити число наступним чином: якщо число x дорівнює 2, програма виведе: "two".

3.2.4 Завдання 4 (10 балів)

Розв'яжіть наступну систему лінійних рівнянь:

$$2x + 3y = 7$$
$$4x - 2y = 10$$

Напишіть функцію `solve_system(a1, b1, c1, a2, b2, c2)`, яка приймає коефіцієнти двох лінійних рівнянь у вигляді $a_1x + b_1y = c_1$ та $a_2x + b_2y = c_2$ як вхідні дані і розв'язує систему рівнянь. Функція повинна повертати розв'язок у вигляді кортежу (x, y), який представляє значення x та y, які задовольняють обом рівнянням. Якщо система рівнянь не має розв'язку або має нескінченно багато розв'язків, поверніть "No unique solution".

Приклад:

```
1 solve_system(2, 3, 7, 4, -2, 10) #            : (2.0, 1.0)
```

```
1 solve_system(1, -2, 3, 2, -4, 6) #            : "No unique
    solution"
```

```
1 solve_system(1, 2, 3, 2, 4, 6) #            : "No unique
    solution"
```

```
1 solve_system(0, 0, 0, 0, 0, 0) #            : "No unique
    solution"
```

Примітка: Покажіть всі необхідні кроки та пояснення, щоб обґрунтувати ваші розв'язки. Ви можете використовувати будь-які відповідні методи або техніки, які вивчалися у школі.

3.2.5 Завдання 5 (10 балів)

Напишіть функцію Python `is`$_palindrome(s)$, `s`$iiii$`True`$, i, i$`False`$.i - , , iiii, iii, i, iiii.$

Приклад:

```
1 is_palindrome("racecar") #           : True
```

```
1 is_palindrome("Hello, World!") #           : False
```

Напишіть функцію is_palindrome, щоб вирішити завдання та протестуйте її з різними рядками.

3.3 Завдання S3

3.3.1 Завдання 1 (10 балів)

Напишіть функцію calculate_factorial(n), яка обчислює і повертає факторіал натурального числа n. Ця функція повинна викликати виключення, якщо n є від'ємним числом або не цілим числом. Приклад:

```
1 calculate_factorial(5) #            : 120
```

Це поверне $5 \times 4 \times 3 \times 2 \times 1 = 120$.

3.3.2 Завдання 2 (10 балів)

Напишіть функцію print_fibonacci(n), яка виводить перші n чисел послідовності Фібоначчі. Функція повинна вивести кожне число на новому рядку. Функція не повертає нічого, вона лише виводить числа Фібоначчі. Приклад:

```
1 print_fibonacci(7)
```

Цей код повинен вивести:

```
0
1
1
2
3
5
8
```

3.3.3 Завдання 3 (10 балів)

Напишіть функцію is_prime(n), яка повертає True, якщо число є простим, і False у протилежному випадку. Просте число - це натуральне число, більше 1, яке не має додатних дільників, крім 1 і самого себе. Приклад:

```
1 is_prime(7)
```

Це поверне True, оскільки 7 є простим числом.

3.3.4 Завдання 4 (10 балів)

Напишіть функцію find_palindromes(words), яка приймає список слів і повертає новий список, що містить тільки паліндроми з оригінального списку. Паліндром - це слово, яке читається однаково зліва направо і справа наліво. Функція повинна ігнорувати регістр букв, тобто "Mom"і "mOm"вважаються паліндромами.

Приклад:

```
1 words = ["level", "deed", "hello", "Madam", "world"]
2 find_palindromes(words)
```

Це поверне ["level "deed "Madam"], оскільки ці слова є паліндромами.

3.3.5 Завдання 5 (10 балів)

Напишіть функцію calculate_mean(numbers), яка приймає список чисел і повертає середнє значення (середнє арифметичне). Функція повинна повертати середнє значення у форматі числа з плаваючою комою.

Приклад:

```
1 numbers = [1, 2, 3, 4, 5]
2 calculate_mean(numbers) #             : 3.0
```

Це поверне середнє значення чисел у списку, яке дорівнює 3.0.

3.4 Завдання S4

3.4.1 Завдання 1 (10 балів)

Напишіть функцію calculate_poly_function_val(a, b, c, d, e, k, g), яка обчислює значення функції $f(a, b, c, d, e, k, g) = 10a^3 + 11b^{10} + 12c^3 + 3d^2 + 6e^{18} + 67k^{12} + 22g^4 + \frac{127}{168}a^4 + 4e + \frac{6}{7}g^2 + \sqrt[3]{\frac{3}{2}a^2} - \frac{2}{5}d + 10e^2 - \pi k + \frac{(c+d+k+g)a^2}{b}$ в заданій точці (a, b, c, d, e, k, g).

Вхідні дані функції:

- a (float): Значення змінної a.
- b (float): Значення змінної b.
- c (float): Значення змінної c.
- d (float): Значення змінної d.
- e (float): Значення змінної e.
- k (float): Значення змінної k.
- g (float): Значення змінної g.

Повернення функції:

- result (float): Обчислене значення функції $f(a, b, c, d, e, k, g)$ в заданій точці.

Приклад:

```
1 calculate_poly_function_val(1.5, 2.3, -0.7, 4.2, 0.8, -1.1,
    3.6)
```

Це поверне обчислене значення функції в заданій точці.

Примітка: Переконайтеся, що ви правильно виконуєте необхідні математичні операції згідно з вказаною функцією.

3.4.2 Завдання 2 (10 балів)

Напишіть функцію calculate_factorial(n), яка обчислює і повертає факторіал числа n. Функція повинна викликати виключення, якщо n є від'ємним числом або не цілим числом. Приклад:

```
1 calculate_factorial(5)
```

Ця функція поверне $5 \times 4 \times 3 \times 2 \times 1 = 120$.

3.4.3 Завдання 3 (10 балів)

Напишіть функцію calculate_combination(n, r), яка обчислює і повертає комбінацію n елементів, вибраних r разів, де n і r є натуральними числами і $r \leq n$. Детальніша інформація про символ Ньютона бінома для підрахунку кількості способів вибрати r різних елементів з n наведена нижче. Приклад:

```
1 calculate_combination(5, 2)
```

Ця функція поверне 10, оскільки є 10 способів вибрати 2 елементи з 5.

Комбінація, позначена як $\binom{n}{r}$, представляє кількість способів вибрати r елементів із набору з n різних елементів, без урахування їх порядку. Це можна обчислити за формулою:

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

3.4.4 Завдання 4 (10 балів)

Напишіть функцію analyze_apple_weights(apple_weights), яка приймає список ваг яблук в якості вхідних даних і виконує такий аналіз:

Обчислити середню вагу яблук. Визначити максимальну вагу серед яблук. Визначити мінімальну вагу серед яблук.

Функція повинна повертати словник, що містить результати аналізу.

Вхідні дані функції:

- apple_weights (list): Список чисел з плаваючою комою, що представляє ваги яблук.

Повернення функції:

- analysis_results (dictionary): Словник, що містить наступні результати аналізу:

  "average_weight": Середня вага яблук. "max_weight": Максимальна вага серед яблук. "min_weight": Мінімальна вага серед яблук.

Приклад:

```
1 apple_weights = [0.2, 0.5, 0.3, 0.6, 0.4]
2 analyze_apple_weights(apple_weights)
```

Це поверне наступний словник:

```
{
"average_weight": 0.4,
"max_weight": 0.6,
"min_weight": 0.2
}
```

Примітка: Це спрощене завдання зосереджується на обчисленні середньої, максимальної та мінімальної ваг яблук у наборі даних. Воно не включає додаткові вимоги до аналізу, згадані в початковому завданні.

3.4.5 Завдання 5 (10 балів)

Розгляньте список курсів MIT у наступному форматі:

- Course 1 - Civil and Environmental Engineering
- Course 2 - Mechanical Engineering
- Course 3 - Materials Science and Engineering

- Course 4 - Architecture
- Course 5 - Chemistry
- Course 6 - Electrical Engineering and Computer Science
- Course 7 - Biology
- Course 8 - Physics
- Course 9 - Brain and Cognitive Sciences
- Course 10 - Chemical Engineering

Напишіть функцію get_course_name(number), яка приймає число в якості вхідних даних і повертає назву курсу MIT, що відповідає заданому числу.

Вхідні дані функції:

- number (int): Номер курсу, для якого потрібно отримати назву.

Повернення функції:

- course_name (str): Назва курсу MIT, що відповідає заданому числу.

Приклад:

```
1 get_course_name(1)
```

Це поверне наступний рядок:

"Course 1 - Civil and Environmental Engineering"

3.5 Завдання S5

3.5.1 Завдання 1 (10 балів)

Створіть функцію perform_operation(a, b, c=0, d=0, e=0, f=0, g=0, h=0, i=0, operator='+'), яка приймає два числа (a і b) і до восьми додаткових чисел (c до i) як аргументи. Функція повинна також приймати оператор (+, -, *, /) як ключовий аргумент. Функція повинна виконувати відповідну операцію над усіма наданими числами і повертати результат. Якщо оператор є діленням (/), а друге число дорівнює нулю, функція повинна повернути повідомлення про помилку.

Вхідні дані функції:

- a (числовий тип даних): Перше число.
- b (числовий тип даних): Друге число.
- c до i (числовий тип даних, необов'язкові): До восьми додаткових чисел.
- operator (рядок, необов'язковий): Оператор, що визначає операцію. За замовчуванням - '+'.

Повернення функції:

- result (числовий тип даних або рядок): Результат операції. Якщо оператор є діленням (/) і друге число дорівнює нулю, поверніть повідомлення про помилку: "Error: Division by zero".

Приклад:

```
1 print(perform_operation(5, 3, 2, 4, operator='+')) #
                : 14
2 print(perform_operation(10, 2, 3, operator='*')) #
                : 60
3 print(perform_operation(5, 0, operator='/')) # Output: "Error
   : Division by zero"
```

Примітка: Конкретні деталі реалізації та назви змінних можуть відрізнятися.

3.5.2 Завдання 2 (10 балів)

Напишіть функцію fibonacci(n), яка обчислює n-те число Фібоначчі за допомогою рекурсії. Приклад:

```
1 print(fibonacci(10))
```

Це повинно вивести 10-те число Фібоначчі.

3.5.3 Завдання 3 (10 балів)

Обчислимо значення ітерованого інтегралу диференційовної та неперервної функції $f(x, y, z) = 1$:

$$\int_a^b \int_c^d \int_e^g 1 \, dx \, dy \, dz = (b-a)(d-c)(g-e)$$

Напишіть функцію calculate_iterated_integral(a, b, c, d, e, g), яка обчислює значення ітерованого інтегралу диференційовної та неперервної функції $f(x, y, z) = 1$. Функція повинна приймати межі інтегрування a, b, c, d, e і g як вхідні дані і повертати обчислене значення ітерованого інтегралу.

Вхідні дані функції:

- a (числовий тип даних): Нижня межа інтегрування змінної x.
- b (числовий тип даних): Верхня межа інтегрування змінної x.
- c (числовий тип даних): Нижня межа інтегрування змінної y.
- d (числовий тип даних): Верхня межа інтегрування змінної y.
- e (числовий тип даних): Нижня межа інтегрування змінної z.
- g (числовий тип даних): Верхня межа інтегрування змінної z.

Повернення функції:

- result (числовий тип даних): Обчислене значення ітерованого інтегралу, яке дорівнює $(b - a)(d - c)(g - e)$.

Приклад:

```
result = calculate_iterated_integral(1, 3, 2, 4, 0, 5)
print(result) #              : 36
```

Примітка: Конкретні деталі реалізації, назва функції та змінних можуть відрізнятися

3.5.4 Завдання 4 (10 балів)

Напишіть функцію is_palindrome(s), яка приймає рядок s як вхідні дані і повертає True, якщо рядок є паліндромом, і False - в іншому випадку. Паліндром - це слово, фраза, число або інша послідовність символів, яка читається однаково в обох напрямках, ігноруючи пропуски, пунктуацію та регістр символів.

Приклад:

```
is_palindrome("racecar") #              : True
```

```
is_palindrome("Hello, World!") #              : False
```

Напишіть функцію is_palindrome для вирішення завдання та перевірте її роботу з різними рядками.

3.5.5 Завдання 5 (10 балів)

Створіть функцію, яка приймає номер дня року (ціле число від 1 до 365) як вхідні дані і повертає відповідний місяць. Ви можете вважати, що це не високосний рік. З метою спрощення, ви можете вважати, що кожен місяць має фіксовану кількість днів. Функція повинна повертати назву місяця у вигляді рядка. Приклад:

```
print(get_month(75)) #              : March
```

4 Завдання P

4.1 Проект 1: Малювання за допомогою Turtle

4.1.1 Опис

Метою цієї програми є створення інтерактивного додатку для малювання за допомогою модуля turtle у Python. Програма дозволяє користувачу керувати об'єктом черепахи на екрані та виконувати різні дії.

4.1.2 Специфікація

Ось розбиття задач, які виконує програма:

[label=0.]Налаштування черепахи:

1. • Створити об'єкт черепахи.
   • Встановити швидкість черепахи на 100.
   • Встановити початковий колір черепахи на червоний.
   • Встановити ширину черепахи на 1.
   • Встановити форму черепахи на "turtle".
   • Опустити перо черепахи для початку малювання.

2. Визначення функцій зміни кольору:

   • Реалізувати $\texttt{turtle}_{c}olor_{r}ed(), ii.i$

3• Визначення функції події миші:

   • Реалізувати fxn(x, y), щоб обробити події перетягування миші.
   • Зупинити відстеження шляху черепахи.
   • Змінити кут та напрям черепахи до нових координат (x, y).
   • Перемістити черепаху до нових координат (x, y).
   • Увімкнути можливість виклику функції знову для подальшого перетягування.

4. Визначення функцій подій клавіатури:

   • Реалізувати $\texttt{move}_{f}orward(), i50.i$

5• Налаштування прослуховувачів подій:

   • Отримати об'єкт екрану черепахи.
   • Увімкнути прослуховування подій клавіш та кліків миші.
   • Зареєструвати обробники подій для конкретних клавіш та кліків миші.
   • При виникненні подій викликаються відповідні функції для виконання необхідних дій.

6. Увійти до головного циклу подій:

   • Запустити цикл подій черепахи.

- Програма постійно прослуховує події та реагує на них.
- Програма залишається інтерактивною до закриття вікна.

Головною метою цієї програми є надання інтерактивного досвіду малювання, де користувач може керувати рухом черепахи, змінювати її колір та заповнювати екран кольором. Програма використовує різні функції, що реагують на події, для відповіді на введення користувача та оновлення поведінки черепахи на екрані.

4.2 Проект 2: Гра вгадування чисел

4.2.1 Опис

У цьому проекті ви створите гру вгадування чисел. Програма буде генерувати випадкове число у визначеному діапазоні, а користувачу потрібно буде вгадати число протягом певної кількості спроб. Після кожної спроби програма надасть зворотний зв'язок користувачеві, якщо вгадка занадто висока або занадто низька. Гра продовжуватиметься до тих пір, поки користувач не вгадає правильне число або не закінчиться кількість спроб.

4.2.2 Специфікація

- Програма повинна генерувати випадкове число у визначеному діапазоні.
- Користувача потрібно запитати введення вгадки.
- Програма повинна надавати зворотний зв'язок користувачу, якщо вгадка занадто висока або занадто низька.
- Програма повинна відстежувати кількість спроб.
- Гра має продовжуватися до тих пір, поки користувач не вгадає правильне число або не закінчиться кількість спроб.
- Програма повинна виводити повідомлення, що інформує користувача, чи виграв він чи програв гру.

4.3 Проект 3: Список справ

4.3.1 Опис

У цьому проекті ви створите простий додаток для списку справ. Програма дозволить користувачу додавати завдання, відзначати завдання як виконані та переглядати список завдань. Завдання будуть зберігатися в пам'яті протягом роботи програми, але втрачатимуться після закриття програми.

4.3.2 Специфікація

- Програма повинна надавати меню з опціями для додавання завдання, відзначення завдання як виконаного та перегляду списку завдань.
- Користувач повинен мати змогу ввести деталі завдання (наприклад, назву завдання, дату завершення) при додаванні завдання.
- Програма повинна зберігати завдання в список або структуру даних.
- Програма повинна відображати список завдань з їх деталями.
- Користувач повинен мати змогу відзначити завдання як виконане, що оновить його стан у списку.
- Програма повинна обробляти некоректні введення та надавати відповідні повідомлення про помилки.

4.4 Проект 4: Простий калькулятор

4.4.1 Опис

У цьому проекті ви створите просту програму-калькулятор. Програма буде просити користувача ввести два числа та операцію (+, -, *, /), а потім виконувати відповідний розрахунок та виводити результат.

4.4.2 Специфікація

- Програма повинна просити користувача ввести перше число.
- Програма повинна просити користувача ввести друге число.
- Програма повинна просити користувача ввести операцію (+, -, *, /).
- Програма повинна виконати відповідний розрахунок на основі введених чисел та операції.
- Програма повинна вивести результат розрахунку.
- Програма повинна обробляти некоректні введення та надавати відповідні повідомлення про помилки.

4.5 Проект 5: Гра Вісіллиця

4.5.1 Опис

У цьому проекті ви створите гру "Вісіллиця". Програма буде вибирати випадкове слово з попередньо заданого списку, і користувачеві доведеться вгадувати літери слова по одній. У користувача буде обмежена кількість спроб, і програма буде надавати зворотний зв'язок про правильність кожної відповіді.

4.5.2 Специфікації

- Програма повинна вибрати випадкове слово з попередньо заданого списку слів.
- Програма повинна відображати початковий стан слова з підкресленими літерами, які ще не вгадані.
- Програма повинна просити користувача ввести літеру для вгадування.
- Програма повинна перевіряти правильність вгадування і оновлювати стан слова відповідно.
- Програма повинна відображати оновлений стан слова з вірно вгаданими літерами.
- Програма повинна відстежувати кількість спроб і обмежувати їх до певної кількості.
- Програма повинна відображати повідомлення, що показує, чи переміг користувач у грі чи програв.

4.6 Additional references

4.7 A Deep Dive into the Quadratic Equation

In the discipline of algebra, a preeminent form of a polynomial equation is the second-order polynomial equation, more commonly known as the quadratic equation. A polynomial function of a single variable x is typically expressed in the following general form:

$$f(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} + a_n x^n \tag{1}$$

where the $f : \mathbb{R} \to \mathbb{R}$ or generally $f : \mathbb{C} \to \mathbb{C}$, $a_i,\ i \in \mathbb{N}, i \in \{0, 1, ..., n\}$ are constants and they are known as the coefficients of the polynomial. The power n is a nonnegative integer and is called the degree of the polynomial. The coefficient a_n of the highest power is called the leading coefficient.

The standard form of second order equation is typically expressed as $ax^2 + bx + c = 0$, where a, b, and c are constants that are defined such that $a \neq 0$. The coefficients a, b, and c are often referred to as the quadratic, linear, and constant terms, respectively.

The problem is: find the values of the f domain, such that f(x) = 0, f defined in (1). An essential concept in the examination of these quadratic equations is the quadratic formula, which is universally used to calculate the roots of the equation. The quadratic formula is stated as:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

An interesting feature of the quadratic formula is the term under the square root, $b^2 - 4ac$. This term is known as the discriminant. It plays a pivotal role in determining the nature of the solutions to the quadratic equation.

- If the discriminant is positive, the equation possesses two distinct real roots.
- If the discriminant equals zero, the equation has one real root, often referred to as a repeated or double root.
- If the discriminant is negative, the equation gives rise to two complex roots.

Consider, for instance, the quadratic equation $x^2 - 5x + 6 = 0$. The coefficients of this equation are $a = 1$, $b = -5$, and $c = 6$. By calculating the discriminant, we find $(-5)^2 - 4 * 1 * 6 = 25 - 24 = 1$. As this is a positive value, we infer that the equation has two distinct real roots. By substituting the coefficients into the quadratic formula, we find the solutions to be:

$$x = \frac{-(-5) \pm \sqrt{(-5)^2 - 4 * 1 * 6}}{2 * 1} = \frac{5 \pm 1}{2}$$

Therefore, the roots of the equation are $x = \frac{5+1}{2} = 3$ and $x = \frac{5-1}{2} = 2$. The study of quadratic equations and their solutions is a fundamental part of algebra, and it underlies many of the more advanced topics in mathematics.

4.8 Solving Second Order Equations

A second order equation, also known as a quadratic equation, is an equation of the form $ax^2 + bx + c = 0$, where a, b, and c are constants, and $a \neq 0$.

The solutions to a quadratic equation are given by the quadratic formula (2):

The term under the square root, b^2-4ac, is known as the discriminant. It determines the nature of the roots of the quadratic equation.

- If the discriminant is positive, the equation has two distinct real roots.
- If the discriminant is zero, the equation has exactly one real root (or a repeated real root).
- If the discriminant is negative, the equation has two complex roots.

For example, let's solve the equation $x^2-5x+6=0$. Here $a=1$, $b=-5$, and $c=6$. The discriminant is $(-5)^2-4*1*6=25-24=1$, which is positive. Thus the equation has two distinct real roots. Applying the quadratic formula gives:

$$x = \frac{-(-5) \pm \sqrt{(-5)^2 - 4 * 1 * 6}}{2 * 1} = \frac{5 \pm 1}{2}$$

So the solutions are $x = \frac{5+1}{2} = 3$ and $x = \frac{5-1}{2} = 2$.

Recommended Literature

Література

[1] Matthes, E. (2019). *Python Crash Course: A Hands-On, Project-Based Introduction to Programming*. No Starch Press.

[2] Sweigart, A. (2015). *Automate the Boring Stuff with Python: Practical Programming for Total Beginners*. No Starch Press.

[3] Ramalho, L. (2015). *Fluent Python: Clear, Concise, and Effective Programming*. O'Reilly Media.

[4] Lutz, M. (2013). *Learning Python*. O'Reilly Media.

[5] VanderPlas, J. (2016). *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media.

[6] Slatkin, B. (2015). *Effective Python: 59 Specific Ways to Write Better Python*. Addison-Wesley Professional.

[7] Downey, A. B. (2012). *Think Python: How to Think Like a Computer Scientist*. Green Tea Press.

[8] Python Software Foundation. *Python Documentation*. Retrieved from https://docs.python.org

[9] Python Software Foundation. *Python Tutorial*. Retrieved from https://docs.python.org/3/tutorial/index.html

[10] Real Python. *Python Tutorials and Articles*. Retrieved from https://realpython.com