სამეცნიერო-ტექნიკური ამოცანების გადაწყვეტა Python დაპროგრამების ენაზე

ლელა გაჩეჩილაძე, ლაშა იაშვილი, ლიანა ნონიკაშვილი

საქართველოს ტექნიკური უნივერსიტეტი

l_gachechiladze@gtu.ge, l.iashvili@gtu.ge, nonikashvili55@mail.ru

რეზიუმე

განხილულია Python დაპროგრამების ენის ფართო შესაძლებლობები სამეცნიერო-ტექნიკური ამოცანების გადაწყვეტის მიზნით. შემუშავებულია ვექტორების სკალარული ნამრავლის გამოთვლის, წრფივ განტოლებათა სისტემის და პირველი რიგის დიფერენციალური განტოლების ამოხსნის და დაპროგრამების სხვადასხვა ენის web-ში, მონაცემთა მეცნიერებასა და თამაშებში გამოყენების ამსახველი პროგრამული კოდები.

დასმული ამოცანების პროგრამული რეალიზების მიზნით შემოთავაზებულია Python დაპროგრამების ენის ისეთი მნიშვნელოვანი ბიბლიოთეკები, როგორიცაა: numpy - მასივებზე ოპერაციის შესრულებისთვის, SciPy - დიფერენციალური განტოლებების რიცხვითი ამოხსნისა და ფუნქციების ოპტიმიზაციისთვის, Matplotlib - მონაცემების დამუშავების, გაფილტვრისა და ვიზუალიზაციისთვის. პროგრამული კოდების შედეგები წარმოდგენილია როგორც კონსოლზე, ასევე გრაფიკული ფორმით.

პროგრამები შესრულებულია Pycharm ინტეგრირებულ გარემოში, რომელიც გამოირჩევა მოქნილობითა და პროგრამისტისთვის კომფორტული შესაძლებლობებით.

ვიმედოვნებთ, რომ სტატია სასარგებლო იქნება როგორც დამწყებთათვის, ასევე გამოცდილი პროფესიონალებისთვის, რომლებიც დაინტერესებულნი არიან Python-ის გამოყენებით სამეცნიერო და ტექნიკურ კვლევებში.

საკვანძო სიტყვები: Python, შესრულების ინტეგრირებული გარემო Pycharm, ბიბლიოთეკები: SciPy, NumPy, Matplotlib.

1. შესავალი.

Python-ი სამეცნიერო კვლევებისა და ინჟინერიის სფეროში ერთ-ერთ ყველაზე პოპულარულ დაპროგრამების ენას წარმოადგენს თავისი სიმარტივის, მოქნილობისა და ბიბლიოთეკების მდიდარი ეკოსისტემის გამო.

წინამდებარე სტატიაში განვიხილავთ Python-ის გამოყენების შესაძლებლობებს სამეცნიერო და საინჟინრო მიმართულების ისეთი პრობლემების გადასაჭრელად, როგორიცაა: მონაცემთა ანალიზი და სამეცნიერო ამოცანების რიცხვითი გადაწყვეტა.

კვლევის მიზნის მისაღწევად ჩვენ მიერ გამოყენებულ იქნა Python-ზე დაპროგრამების არაერთი ბიბლიოთეკა. კერძოდ, მასივებთან და მატრიცებთან სამუშაოდ მივმართეთ NumPy ბიბლიოთეკას, რომელიც მაღალეფექტურ ინსტრუმენტებს გვთავაზობს რიცხვითი გამოთვლებისათვის. ამას გარდა, დიფერენციალური განტოლებების რიცხვითი ამოხსნისა და ფუნქციების ოპტიმიზაციისთვის გამოყენებულ იქნა SciPy ბიბლიოთეკა, ხოლო მონაცემთა ანალიზისა და შედეგების ვიზუალიზაციისთვის - Matplotlib ბიბლიოთეკის შესაძლებლობები.

2. მეთოდოლოგია.

კვლევის ერთ-ერთი მთავარი მიზანი Python-ის გამოყენების ეფექტურობის შესწავლა გახლდათ სამეცნიერო და ტექნიკური პრობლემების გადასაჭრელად. შედეგებმა გვიჩვენა, რომ Python-ის ბიბლიოთეკების გამოყენება მაღალ წარმადობას და სიზუსტეს უზრუნველყოფს მათემატიკური ოპერაციებისა და რიცხვითი მეთოდების შესრულებისას. კერძოდ:

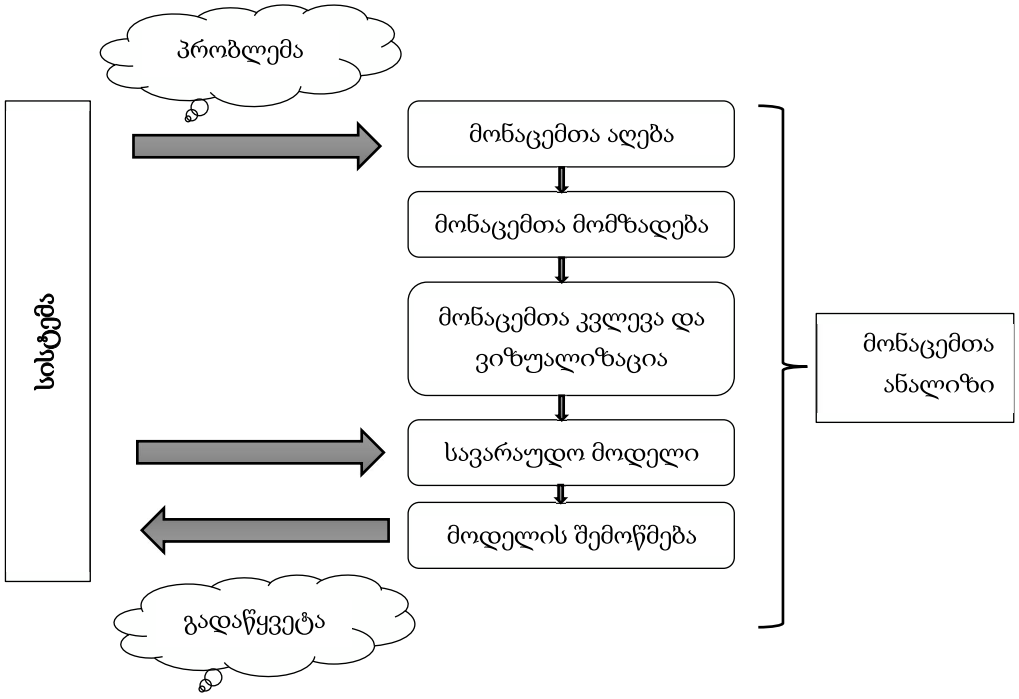
NumPy ბიბლიოთეკის გამოყენებამ მნიშვნელოვნად დააჩქარა ოპერაციები მასივებსა და მატრიცებზე, რაც აუცილებელია დიდი რაოდენობის მონაცემების დამუშავებისას და რთული ფუნქციების გამოთვლისას.

SciPy ბიბლიოთეკამ უზრუნველყო საიმედო ალგორითმები დიფერენციალური განტოლებების რიცხვითი ამოხსნისა და ფუნქციების ოპტიმიზაციისთვის, რაც Python-ს ძლიერ ინსტრუმენტად აქცევს რთული მოვლენების მოდელირებისა და სისტემის პარამეტრების ოპტიმიზაციისთვის.

Matplotlib ბიბლიოთეკის გამოყენებამ გაამარტივა მონაცემების დამუშავება, გაფილტვრა და ვიზუალიზაცია. მრავალფეროვანი გრაფიკებისა და დიაგრამების აგებამ შესაძლებელი გახადა, დაგვენახა მონაცემთა უერთიერთდამოკიდებულებები, რასაც უდიდესი მნიშვნელობა ენიჭება კანონზომიერების დადგენისა და გადაწყვეტილებების მიღების დროს.

3. მსჯელობა და შედეგები

პირველ სურათზე ნაჩვენებია, თუ როგორ წარმოებს მონაცემთა ანალიზი.



სურ. 1. მონაცემთა ანალიზი

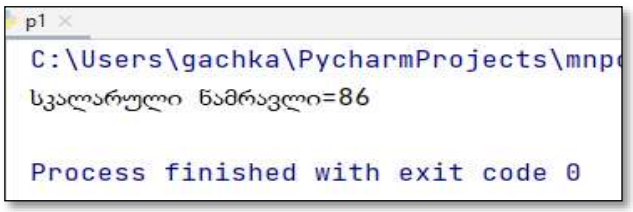
განვიხილოთ მასივებთან მუშაობის ძირითადი ნაწილი:

Python დაპროგრამების ენა NumPy ბიბლიოთეკის მეშვეობით მასივებთან და მატრიცებთან მუშაობის მძლავრ ინსტრუმენტებს გვთავაზობს. ეს უკანასკნელი მრავალგანზომილებიან მასივებზე მაღალეფექტურ ოპერაციებს უზრუნველყოფს და მას მათემატიკური ოპერაციების ფართო სპექტრის მხარდაჭერა აქვს.

ქვემოთ წარმოდგენილია ორი ვექტორის სკალარული ნამრავლის გამოთვლის პროგრამული კოდი:

```
import numpy as np
vec1=np.array([3,4,5]); vec2=np.array([6,7,8])
result=np.dot(vec1, vec2); print("სკალარული ნამრავლი=",result, sep='')
```

პროგრამის შესრულების შედეგი მე-2 სურათზეა ნაჩვენები.



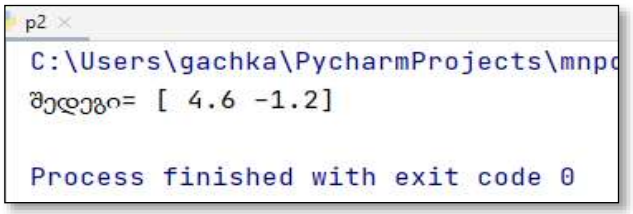
სურ. 2. ვექტორების სკალარული ნამრავლის გამოთვლის შედეგი

განვიხილოთ წრფივ განტოლებათა შემდეგი სისტემის ამოხსნის მაგალითი:

$$\begin{cases} 2x + y = 8 \\ x + 3y = 1 \end{cases} \quad (1)$$

დასმული ამოცანის გადაწყვეტის პროგრამულ კოდს ქვემოთ ნაჩვენები სახე აქვს, ხოლო მისი შესრულების შედეგები მე-3 სურათზეა ნაჩვენები.

```
import numpy as np
# განტოლებათა სისტემის კოეფიციენტები
coeff1 = np.array([[2, 1], [1,3]])
# მნიშვნელობები
values1 = np.array([8, 1])
```




```
result = np.linalg.solve(coeff1, values1); print("შედეგი=", result)
```

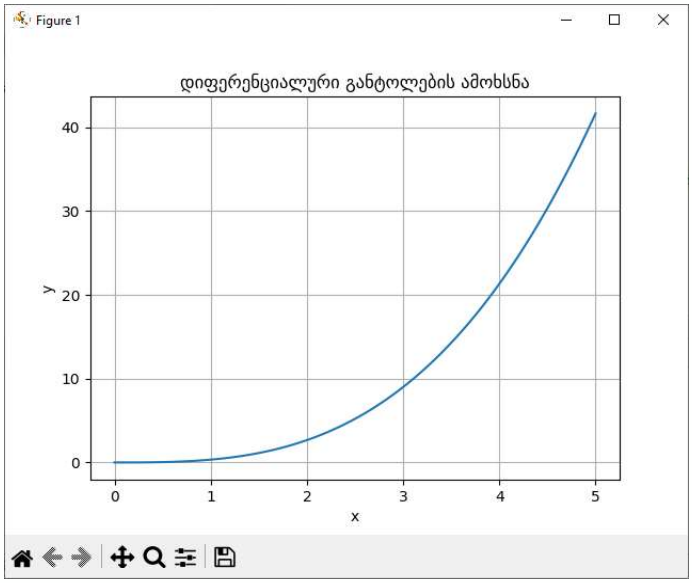
სურ. 3. წრფივ განტოლებათა სისტემის ამოხსნის შედეგი

აღსანიშნავია, რომ წრფივ განტოლებათა სისტემის ამოსახსნელად numpy ბიბლიოთეკიდან პითონი გვთავაზობს linalg.solve() ფუნქციის გამოყენებას.

ახლა წარმოვადგინოთ $\frac{dy}{dx} = x^2$ პირველი რიგის დიფერენციალური განტოლების ამოხსნის პროგრამული კოდი, რომლის შედეგი გრაფიკული სახით მე-4 ნახაზზეა ნაჩვენები.

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
def function(y, x):
  return x**2
# საწყისი პირობა
condition = 0
# x-ის მნიშვნელობები ინტეგრირებისთვის
x_values = np.linspace(0, 5, 100); result = odeint(function, condition, x_values)
plt.plot(x_values, result); plt.xlabel("x"); plt.ylabel("y")
```

plt.title("დიფერენციალური განტოლების ამოხსნა"); plt.grid(True); plt.show()



სურ. 4. პირველი რიგის დიფერენციალური განტოლების ამოხსნის შედეგი

დიფერენციალურ განტოლებათა რიცხვითი ამოხსნისთვის ჩვენ მიერ გამოყენებული იქნა scipy ბიბლიოთეკა.

ამჯერად, ერთ ფანჯარაში წარმოვადგინოთ შემდეგ ფუნქციათა გრაფიკები:

$y = 4x + 3;\ \ y1 = 3x^2 + 5;\ y2 = 5 * x^3\ ;\ \ y3 = x^4$, სადაც x იცვლება ინტერვალში: [1;9]. (2)

დასმული ამოცანის პროგრამულ კოდს ქვემოთ ნაჩვენები სახე აქვს, ხოლო პროგრამის შესრულების შედეგები მე-5 სურათზეა ნაჩვენები.

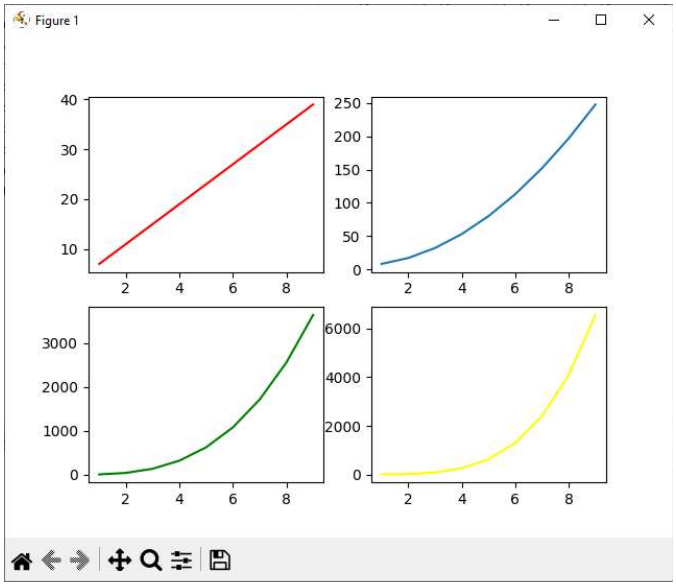
```
import matplotlib.pyplot as plt
import numpy as np
x =np.arange(1,10)
y =4*x+3
y1 =3* x**2+5
y2 =5* x**3
y3= x**4
plt.subplot(2,2,1); plt.plot(x,y,'red')
plt.subplot(2,2,2)
plt.plot(x,y1)
plt.subplot(2,2,3); plt.plot(x,y2, 'green')
plt.subplot(2,2,4)
plt.plot(x,y3,'yellow')
plt.show()
```

აღსანიშნავია ის ფაქტი, რომ ზემოთ წარმოდგენილ პროგრამულ კოდებში განთავსებული numpy და scipy ბიბლიოთეკების ინსტალირების მიზნით Pycharm შესრულების ინტეგრირებული გარემოს ტერმინალზე გამოყენებულ იქნა შემდეგი ბრძანებები:

pip install pyinstaller numpy

pip install pyinstaller scipy

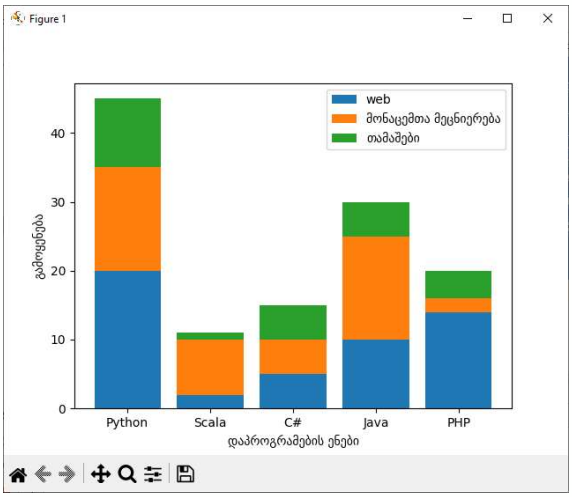
რომელთა ინსტალირების გარეშე უბრალოდ ვერ იმუშავებდა ნაჩვენები პროგრამული კოდები.



სურ. 5. ფუნქციების გრაფიკები

ახლა განვიხილოთ, თუ როგორ ხდება სვეტოვანი დიაგრამის აგება, რომლიც გვიჩვენებს დაპროგრამების ისეთი ენების, როგორიცაა: Python, Scala, Java, C# და PHP გამოყენებას web-ში, მონაცემთა მეცნიერებასა და თამაშებში. ქვემოთ წარმოდგენილი პროგრამული კოდის შესრულების შედეგი მე-6 სურათზეა ნაჩვენები.

```
import matplotlib.pyplot as pyplot
labels=('Python','Scala','C#', 'Java', 'PHP')
index=(1,2,3,4,5); web=[20, 2, 5, 10, 14]; data_science=[15, 8, 5, 15, 2]; games=[10, 1, 5, 5, 4]
pyplot.bar(index,web,tick_label=labels,label='web')
pyplot.bar(index,data_science,tick_label=labels,label='მონაცემთა მეცნიერება', bottom=web)
web_and_games = [web[i] + data_science[i]
for i in range(0, len(web))]
pyplot.bar(index, games, tick_label=labels, label='თამაშები', bottom=web_and_games)
pyplot.xlabel("დაპროგრამების ენები");
pyplot.ylabel("გამოყენება");pyplot.legend();pyplot.show()
```



სურ. 6. დაპროგრამების ენების გამოყენება სხვადასხვა მიმართულებით

4. დასკვნა

ამრიგად, ჩატარებული კვლევის საფუძველზე, თამამად შეიძლება ითქვას, რომ Python-ი არის დაპროგრამების ძლიერი და უნივერსალური ენა სამეცნიერო-ტექნიკური პრობლემების გადასაჭრელად. სიმარტივე, მოქნილობა და მდიდარი სტანდარტული ბიბლიოთეკა მას იდეალურ პლატფორმად აქცევს მონაცემთა ანალიზისა და სხვადასხვა სახის განტოლებების რიცხვითი ამოხსნისთვის. ამიტომ, Python-ი კვლავ პოპულარულ არჩევანს წარმოადგენს მეცნიერებსა და ინჟინრებს შორის, რომლებიც ცდილობენ ეფექტურად გადაჭრან რთული სამეცნიერო და ტექნიკური პრობლემები.

გამოყენებული ლიტერატურა:

1. W. McKinney Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. Sebastopol, CA: O'Reilly Media. 2018.
2. J. VanderPlas. Python Data Science Handbook: Essential Tools for Working with Data. Sebastopol, CA: O'Reilly Media. 2016.
3. M. Lutz. Learning Python: Powerful Object-Oriented Programming. Sebastopol, CA: O'Reilly Media, 2013.
4. M. Waskom. Seaborn: statistical data visualization. Journal of Open Source Software, 5(86), 3529. 2020.
5. J. D. Hunter. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9(3), 90-95. 2007.
6. L. Gachechiladze. Python Programming Language. Publishing House "Technical University". 2018

Solving scientific- technical problems in the Python programming language

Lela Gachechiladze, Lasha Iashvili, Liana Nonikashvili

Georgian Technical University

l_gachechiladze@gtu.ge, l.iashvili@gtu.ge, nonikashvili55@mail.ru

Abstract

The article discusses the wide possibilities of the Python programming language for solving scientific and technical problems. Program codes have been developed for calculating the scalar product of vectors, solving a system of linear equations, solving a first-order differential equation and using them in various programming languages for the web, data science and games.

For the programming implementation of the tasks are used, such important libraries of the Python programming language as: numpy - for performing operations on arrays, SciPy - for numerically solving differential equations and optimizing functions, Matplotlib - for processing, filtering, and visualizing data. The results of the program codes are presented both in console and graphical form.

The programs are written in the Pycharm integrated environment, which is flexible and comfortable for the programmers.

We hope that the article will be useful for both beginners and experienced specialists interested to using Python in scientific and technical research.

Keywords: Python, Pycharm - integrated development environment, libraries: SciPy, NumPy, Matplotlib.