THE PYTHON PROGRAMMING LANGUAGE AND ITS CAPABILITIES IN THE GIS ENVIRONMENT

Katarzyna Szczepankowska mgr inż.
Department of Real Estate Management and Regional Development
University of Warmia and Mazury in Olsztyn
Olsztyn, Poland
e-mail: katarzyna.szczepankowska@uwm.edu.pl

Krzysztof Pawliczuk mgr inż.
The Office of Student Systems Services
University of Warmia and Mazury in Olsztyn
Olsztyn, Poland
e-mail: krzysztof.pawliczuk@uwm.edu.pl

Sabina Źróbek
Department of Real Estate Management and Regional Development
University of Warmia and Mazury in Olsztyn
Olsztyn, Poland
e-mail: zrobek@uwm.edu.pl

Abstract
The first chapter of this paper reviews the most popular programming languages, including Java, PHP, Python, C# and C++, and describes their characteristic features. Chapter discusses the capabilities, technical features and advantages of Python which is quickly taking the lead as one of the most popular programming languages in the world. The authors emphasize Python's key strengths, including high efficiency, easy implementation of new functions, code readability, easy upgrade and its open source nature.

The third chapter presents Python scripting options in the GIS environment and its popularity with major technology developers, among them Google, YouTube, Nokia and NASA.

The last chapter discusses the rapid increase in Python’s popularity and discusses its most recent applications, including in ESRI software and computer games. The extensive capabilities of Python make it suitable for developing various types of interactive maps.

Key words: Programming language, Python, GIS, interactive map.

1. History of programming languages
The history of computing machines dates back to the 20th century, but the breakthrough event in digital electronics was the invention of the transistor, the key component of various electronic systems. The Electronic Numerical Integrator and Computer (ENIAC), the first computer termed "bicycle for the mind" by Steve Jobs, revolutionized computing science. Successive machine programs were described, laying the foundation for the development of the first programming languages. Computer programming is a complex process which involves designing, writing, testing and maintaining the software source code. Source codes are written in a programming language with the use of specific rules and principles (syntax and semantics).

This report does not offer a detailed description of the hundreds of thousands of programming languages that have been designed to date. It focuses on languages which, according to the authors’ subjective opinion, have made the greatest contribution to computer programming.

According to the PYPL Popularity of Programming Language index published on the pyDatalog website in March 2013, the top ten languages were:
The above list was compiled by analyzing how often language tutorials are searched on Google. This is a leading indicator which reflects current trends and users’ real needs.

Ruby occupies the last place in the discussed ranking list. It enjoys a stable position in comparison with last year’s results. Ruby is a dynamic, interpreted programming language which was designed in the mid 1990s. An interpreted language is stored in the form of a source code, and it is loaded, interpreted and executed by the interpreter (Still, 2006.)

Slot number 9 is occupied by Visual Basic, a Microsoft programming tool which moved down one place from March 2012. Visual Basic was derived from Basic (Beginner’s All-purpose Symbolic Instruction Code), a simple language designed for writing short programs. Visual Basic relies largely on Basic syntax, it contains several hundred instructions, functions and key words, but it is not a true object-oriented programming language (Herman et. al, 2008).

JavaScript defended last year’s position of the 7th most popular programming language.

The languages occupying positions 8, 6, 5 and 4 in the ranking are related to C, in programmers opinion the most important programming language of all time (ranked at No. 6, marking a significant drop of three positions from March 2012). Its structure combines the features of high-level and low-level programming languages (Prata, 2006). High-level languages rely on ready-made functions, whereas low-level languages deal directly with memory addresses and data. The source code in C is transferable, therefore, a developed program can be run on different machines and in different operating systems after it has been compiled in a new environment.

In the early 1980s, the C language was provided with an object-oriented extension which gave rise to Objective-C (position 8, up by one step from March 2012). In Objective-C, messages are passed to object instances (Fairbairn et. al, 2012). Objective-C allows single inheritance (a class can inherit characteristics and features from only one superclass). The main features differentiating C from Objective-C include symbol [] for calling methods and symbol @ for language-specific definitions.

The emergence of C++ (slot No. 5, down by one step from March 2012), which combines the simplicity of classical C programming and object-oriented programming, was a breakthrough moment. Object-oriented programs change the way we think about programming. Small and large models of real-life objects are developed in the program, their state and behavior are described, and the models embedded in the program exchange instructions (Grębosz, 2000).

Table 1

<table>
<thead>
<tr>
<th>Position</th>
<th>Position</th>
<th>Delta in</th>
<th>Programming</th>
<th>Share in</th>
<th>Trends</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mar 2013</td>
<td>Mar 2012</td>
<td>position</td>
<td>language</td>
<td>Mar 2013</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>Java</td>
<td>27.1 %</td>
<td>-1.0 %</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
<td>PHP</td>
<td>14.4 %</td>
<td>-2.3 %</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>↑↑↑↑</td>
<td>Python</td>
<td>9.6 %</td>
<td>+0.8 %</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>↑</td>
<td>C#</td>
<td>9.4 %</td>
<td>-1.5 %</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>↓</td>
<td>C++</td>
<td>9.3 %</td>
<td>+0.9 %</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>↓↓↓↓</td>
<td>C</td>
<td>8.2 %</td>
<td>-1.6 %</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
<td>Javascript</td>
<td>7.3 %</td>
<td>-0.3 %</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>↑</td>
<td>Objective-C</td>
<td>6.1 %</td>
<td>-1.4 %</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>↓</td>
<td>Visual Basic</td>
<td>5.6 %</td>
<td>-0.8 %</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td></td>
<td>Ruby</td>
<td>2.7 %</td>
<td>+0.1 %</td>
</tr>
</tbody>
</table>

© 2012 Pierre Carbonnelle

**Source:** https://sites.google.com/site/pydatalog/pypl/PyPL-Popularity-of-Programming-Language

The above list was compiled by analyzing how often language tutorials are searched on Google. This is a leading indicator which reflects current trends and users’ real needs.

Ruby occupies the last place in the discussed ranking list. It enjoys a stable position in comparison with last year’s results. Ruby is a dynamic, interpreted programming language which was designed in the mid 1990s. An interpreted language is stored in the form of a source code, and it is loaded, interpreted and executed by the interpreter (Still, 2006.)

Slot number 9 is occupied by Visual Basic, a Microsoft programming tool which moved down one place from March 2012. Visual Basic was derived from Basic (Beginner’s All-purpose Symbolic Instruction Code), a simple language designed for writing short programs. Visual Basic relies largely on Basic syntax, it contains several hundred instructions, functions and key words, but it is not a true object-oriented programming language (Herman et. al, 2008).

JavaScript defended last year’s position of the 7th most popular programming language.

The languages occupying positions 8, 6, 5 and 4 in the ranking are related to C, in programmers opinion the most important programming language of all time (ranked at No. 6, marking a significant drop of three positions from March 2012). Its structure combines the features of high-level and low-level programming languages (Prata, 2006). High-level languages rely on ready-made functions, whereas low-level languages deal directly with memory addresses and data. The source code in C is transferable, therefore, a developed program can be run on different machines and in different operating systems after it has been compiled in a new environment.

In the early 1980s, the C language was provided with an object-oriented extension which gave rise to Objective-C (position 8, up by one step from March 2012). In Objective-C, messages are passed to object instances (Fairbairn et. al, 2012). Objective-C allows single inheritance (a class can inherit characteristics and features from only one superclass). The main features differentiating C from Objective-C include symbol [] for calling methods and symbol @ for language-specific definitions.

The emergence of C++ (slot No. 5, down by one step from March 2012), which combines the simplicity of classical C programming and object-oriented programming, was a breakthrough moment. Object-oriented programs change the way we think about programming. Small and large models of real-life objects are developed in the program, their state and behavior are described, and the models embedded in the program exchange instructions (Grębosz, 2000).
Microsoft **C#** (position 4, up by one position from March 2012) also belongs to the C and C++ family of programming languages. C# is an object-oriented language with memory recovery and exception handling options (Lis, 2012). It is closely related to the .NET framework for application execution, and new implementations are released to ensure compatibility with other platforms.

**Python** ranked 3rd in the discussed ranking, moving up three positions from March 2012. Python was conceived in the early 1990s by Dutch programmer Guido van Rossum. It was designed as an easy, intuitive and powerful open-source programming language. Python is a language "whose main goal is to overcome the complexity of the programming process and preserve the developed programs" (Eckel, 2006).

In 1994, **PHP** (Hypertext Preprocessor) (stable position at No. 2) was developed by Rasmus Lerdorf based on a small program written in Perl. Owing to fast scripting options, its ability to run on any platform and work with many databases, PHP has become one of the most popular scripting languages which is particularly useful for developing websites (Atkinson, 2003).

**Java**, the ranking leader with a stable position, has also evolved from the C language. Java is an object-oriented language for designing source codes compiled to bytecode that can run on any Java virtual machine (Eckel, 2006). Compilation is a process during which a code is automatically translated into an equivalent code in another programming language. According to the PYPL ranking, Python has witnessed the highest rise in popularity in the past year. A growing number of users have opted for Python due to its unquestioned strengths and continued dynamic development.

### 2. Why Python? Python’s advantages over other programming languages

Python is an object-oriented scripting language (Lee, 2011). The operating system in which a Python program is written and/or run does not affect the program’s operation. The Python interpreter recognizes the platform and behaves accordingly in a given environment, whether it is Windows, Linux or Mac. The key advantages of Python are short code and easy to read format. A program written in Python has the same functions as a program written in Java, PHP or C++, but it is characterized by much shorter code and greater code readability. Those differences can be illustrated on the example of a program written in each of the above languages to output the word "test":

<table>
<thead>
<tr>
<th>Python</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print(&quot;test&quot;)</code></td>
<td><code>public class Hello{ public static void main(String[] args){ System.out.print(&quot;test&quot;); } }</code></td>
</tr>
<tr>
<td><code>#include &lt;iostream&gt;</code> <code>int main() { std::cout&lt;&lt;&quot;test&quot;&lt;&lt;std::endl; return 0; }</code></td>
<td><code>&lt;?php echo &quot;test&quot;; ?&gt;</code></td>
</tr>
</tbody>
</table>

**Source:** *own elaboration*

### Python and other programming languages - advantages

Unlike other languages, Python defines code blocks by whitespace indentation. "Python functions have no explicit beginning or end, and no curly braces to mark where the
function code starts and stops. The only delimiter is a colon (:) and the indentation of the code itself. (...) Indenting starts a block and unindenting ends it. There are no explicit braces, brackets, or keywords. This means that whitespace is significant, and must be consistent." (http://www.diveintopython.net/getting_to_know_python/indenting_code.html). The below example shows that indentation and an absence of separating brackets which are found in Java or C++ contributes to code legibility.

<table>
<thead>
<tr>
<th>Code legibility in Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>def factorial(n):</td>
</tr>
<tr>
<td>print ('n =', n)</td>
</tr>
<tr>
<td>if n &gt; 1:</td>
</tr>
<tr>
<td>return n * factorial(n - 1)</td>
</tr>
<tr>
<td>else:</td>
</tr>
<tr>
<td>print ('end')</td>
</tr>
<tr>
<td>return 1</td>
</tr>
</tbody>
</table>

Table 3  

Source: own elaboration  

Indentation has many advantages, and it makes code simpler and easier to read. Some programmers make the common error of mixing tabs and spaces. A program can be written using two or more spaces, but it has to be consistent from the beginning to the end, and only one or the other (tabs or spaces) have to be used for indentation. According to Python’s Style Guide, spaces and tabs do not mix, and the preferred way of indenting a new line is 4 spaces (http://google-styleguide.googlecode.com, Python Style Guide).

Another advantage of Python is that its runtime objects have a type. Data are objects, and variables are references to objects. Variable names are untyped, and the type of the object does not have to be declared or defined before hand. Python is a dynamically typed language, where a reference to one object can be linked to another object which, in itself, can be an object of a different data type. Strongly typed languages, such as C++ and Java, support the execution of only those operations that have been defined for the given type of data. The above limitation also exists in Python, but the discussed language is not strongly typed because executable operations can change, for example when a reference to an object is recreated and associated with a differently typed object. To illustrate:

<table>
<thead>
<tr>
<th>Keeping track of data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>road=866</td>
</tr>
<tr>
<td>print(road, type(road)) # output data: 866 &lt;class 'int'&gt;</td>
</tr>
<tr>
<td>road=&quot;North&quot;</td>
</tr>
<tr>
<td>print(road, type(road)) # output data: North &lt;class 'str'&gt;</td>
</tr>
</tbody>
</table>

Table 4  


Python automatically identifies and keeps track of data type, therefore, data does not have to be explicitly described. In Java, C++ and other statically-typed languages, the data type of the function return value and each function argument have to be specified. In Python, datatypen ever has to be explicitly specified. Python keeps track of the data type internally based on the value assigned to the variable [http://www.diveintopython.net]. In Python, everything is an object [data, classes, types, functions, modules, lists, dictionaries, tuples, files, strings and integers].

The list of objects represented in Python includes tuples, lists and dictionaries which are highly useful in operations performed on various types of objects. A tuple is an immutable list of independently available references. ">>>" is a primary prompt in the Python interpreter.

Table 5

Creating and calling a list, making a reference to a specific element on the list

```python
>>> tuple = 32, 12, 'cookie'          # we can use also >>tuple = (32, 12, 'cookie')
>>> tuple
(23, 21, 'cookie')
>>> tuple[2]
'cookie'
```

Source: own elaboration

The name and index number of the referenced element have to be specified to access a given tuple component [tuple[2]]. In Python, tuple indices are zero-based. A list is very similar to a tuple, and the main difference between the two is that a list is contained in square brackets (list=[1, ‘banana’, 44]) and that it can be modified. Dictionaries present the relationships between a value and a key, and they are enclosed in curly braces.

Table 6

Creating a dictionary and others operations (code description)

```python
>>> dict = {'1': 'variable', '2': 'fireplace'} # create dictionary
>>> dict    # dictionary reference
{'1': 'variable', '2': 'fireplace'}
>>> dict['1'] # reference to value of key 1
'variable'
>>> dict['1'] = 'other variable' # change value of key 1
>>> dict
{'1': 'other variable', '2': 'fireplace'}
>>> dict['3'] = 'third variable' # add pair "key":"value"
>>> dict
{'1': 'other variable', '2': 'fireplace', '3': 'third variable'}
>>> del dict[1] # delete pair "key":"value" for key 1
('2': 'fireplace', '3': 'third variable')
>>> dict.clear() # delete entire list
{}
```

Source: own elaboration

The above example shows a simple list with two "key":"value" pairs separated by commas and enclosed in a curly bracket. The comments to the above example indicate that dictionaries can be freely modified, added, deleted, and values can be changed.

Files also illustrate the simplicity of Python operations. In Java, a buffer has to be declared, a stream has to be initiated and enclosed in a large number of brackets. In Python, only three lines of code are required to open a file, write data and close a file:
Table 7

Saving a character string in a text file

```python
>>> f.open = open("/home/file.txt", "w")
>>> f.write = ("text stored inside file")
>>> f.close()
```

Source: own elaboration

Python is a modular language. A new code does not have to be written to develop an XML parser (DOM, SAX) or perform scientific calculations. Modules written by other programmers can be imported and customized, which considerably shortens programming time. Python can handle multiple threads to synchronize inputs with outputs, and it is capable of running multiple operations. Those features are highly useful for writing extensive programs. In multi-thread programs, several tasks can be executed at the same time. Python also handles exceptions and it can create documentation during the programming process. The latter is a highly useful feature in programs that have to be rebuilt or expanded to include new functions. Programs written in Python have a very simple structure. They comprise modules (self-designed or imported from the Python standard library), modules contain instructions, instructions are made of expressions, and expressions build and process objects.

This paper provides a general overview of the Python programming language, and it presents only a limited number of Python features and functions due to space constraints.

Advantages of Python over other programming languages

According to an Active State survey, dynamic programming languages play an increasingly important role for corporate clients. The below diagram presents the rise in demand for Python programmers over time. Python seems to have revolutionized the market: whereas the demand for PHP programmers remains constant, the demand for Python programmers has increased at a steep rate, suggesting that Python is likely to pose significant competition for Java, .NET or PHP in the future.

Source: http://www.indeed.com/jobtrends?q=Java,+PHP,+Perl,+Net,+Python&relative=1

One of the greatest advantages of Python is that it is an open-source programming language. Open source solutions are generally recognized by businesses. Programmers are increasingly likely to reach for dynamic programming languages which are characterized by:

- high speed,
- ease of use,
- low cost,
- flexible code,
- availability of support from the programming community.

The above features contribute to the popularity of dynamic programming languages in the IT sector.
2. **Python for GIS and not only**

Geographic information systems (GIS) are more difficult to define than could be imagined. One of the best definitions of the concept was proposed by J. Gaździcki and it states that GIS is a system for acquiring, gathering, verifying, analyzing, transferring and releasing spatial data; in a broad context, it incorporates methods, hardware, software, spatial databases, organizations, resources and people who have a special interest in geographic information systems (Gaździcki, 2003).

Developers of GIS software search for innovative and efficient solutions that would contribute to greater automation of the programming process. Python is one of the most popular solutions deployed by GIS developers. **Environmental Systems Research Institute (ESRI)** is the world’s leading developer of geo-spatial software, including ArcGIS. ESRI products are functional tools for gathering, managing, presenting and analyzing spatial data. They support decision-making at various management and response levels.

For several years now, ESRI has been promoting Python as the best choice for writing scripts to automate programming functionalities. ArcPy is a Python site package for performing GIS functions. According to Paul A. Zandbergen, the author of *Python Scripting for ArcGIS*, a step-by-step guide to the creation of Python scripts to enhance ArcGIS capabilities, "Python scripting is a fundamental tool for GIS professionals to extend the functionality of ArcGIS and automate workflows. You can use it to modify the existing capabilities of ArcGIS and create new ones".

Quantum GIS [QGIS] is yet another system for processing geographic information. Unlike ArcPy, QGIS constitutes open source software. The program is available under the GPL license, and it can be used freely for both academic and commercial purposes. QGIS allows users to manage geographic data, create own data, apply GPS data, create maps and perform spatial analyses (http://en.wikipedia.org/wiki/Quantum_GIS).

Similarly to ArcGIS, QGIS features additional plugins for integration with other open source software, including PostgreSQL, GRAAS and MapServer. Plugins written in Python extend the capabilities of QGIS, including plugins to geocode using the Google Geocoding API and to interface with PostgreSQL and MySQL databases (http://en.wikipedia.org/wiki/Quantum_GIS).

Programs that process geographic information are also used to develop computer games. In 2005, Take-Two Interactive released *Sid Meier’s Civilization IV*, a strategic turn-based game where players lead civilizations, build cities, gather resources, make technological and cultural discoveries (such as the wheel, alphabet, cartography, electricity), trade resources with other users, levy taxes and conquer other states. The game is played on the map of the world, and players can choose between maps of the real and imaginary world.

*Source: http://agdom.wordpress.com/tag/civilization-4/*
Game attributes and rules are stored in XML files. Significant parts of the interface, map generators and scripted events are written entirely in Python. A software development kit was released after the game's launch. The kit is a plugin which allows players to modify nearly all games rules and settings. The plugin can also be used to modify maps or create them from scratch by adding landmasses, mountains, rivers, resources, cities, buildings, borders, etc. Civilization IV uses the boost.python library (which binds C++ and Python) to access various parts of the game, including the style and content of interface screens (http://en.wikipedia.org/wiki/Civilization_IV).


Python capabilities extend far beyond computer games and plugins for GIS software. Python features various modules for processing geospatial data. Django is a framework written in Python for creating functional web applications. Its main advantages include well-developed documentation, stability, scalability and the ease of writing applications. The latest version of the framework, Django 1.5, features the GeoDjango module with various functions for building GIS web applications.

The Geopy module is used for geocoding addresses. It relies on third-party geocoding engines to assist programmers in the process of finding geographic coordinates of addresses, cities and countries. Geopy can convert an address to geographic coordinates and vice versa. Nokia uses Python to develop software for its mobile telephones, such as PySpy. NASA relies on Python to build cloud computing applications where Python modules and applications are combined with other programming languages and technologies. Space shuttle engineers used Python to streamline mission design.

3. Conclusions

Research results show that the popularity of Python still lags behind other major languages such as Java and PHP. Despite the above, Python is increasingly often the language of choice for performing scientific and statistical calculations and developing business applications. In addition to major technology developers such as Google, YouTube, Nokia and NASA, smaller businesses, institutions and universities are showing a growing interest in Python.
Python's key strengths are high efficiency, easy implementation of new functions, code readability, easy upgrade and its open source nature. Python enjoys growing popularity on the IT market, and it is increasingly often deployed in GIS applications. The discussed programming language will undoubtedly contribute to the optimization of geographic information systems. Python scripts were successfully used to create and modify maps in *Sid Meier's Civilization IV*, which suggests that the discussed programming language carries a vast potential for the interactive map market. In view of its broad range of capabilities, Python could deployed in line with the provisions the Regulation of the Council of Ministers of 3 October 2011 on the types of thematic and special-purpose maps, including digital hydrographic and environmental maps and maps of average real estate prices. Python’s versatility could contribute to the fulfillment of a vision formulated by Galileo: “Perhaps with time, we will be able to see things that we cannot imagine today”.

**Reference literature**

9. Lis M. 2012,*C# Praktyczny kurs*, Helion, Gliwice.

**Internet sites:**


http://training.esri.com/gateway/index.cfm?fa=catalog.webCourseDetail&CourseId=1868


https://www.e-education.psu.edu/geog485/node/19 (accessed 77 February 2013)


http://gispython.org/ (accessed 13 March 2013)


http://freshsources.com/HTML/10.10/ALLISON/ALLISON.HTM

https://sites.google.com/site/pydatalog/pypl/PyPL-Popularity-of-Programming-Language (accessed 12 April 2013)
http://webhosting.pl/Python.i.PHP.bez.kompleksow.wchodza.do.swiata.aplikacji.biznesowych (accessed 13 April 2013)