Turk. J. Math. Comput. Sci.
11(1)((2019)) 24–28
© MatDer
http://dergipark.gov.tr/tjmcs
http://tjmcs.matder.org.tr

MATDER

Teaching an Introductory Programming Course to Non-Computer Science Majors Using SageMath

Razvan A. Mezei

Computer Science Department, Saint Martin's University, Lacey, WA 98503, USA.

Received: 06-02-2019 • Accepted: 08-03-2019

Abstract. In this paper we propose an introduction to programming course using SageMath for non-Computer Science majors. A course outline on developing and designing the course is briefly presented. Given the large number of packages available in SageMath, such a course could easily be tweaked to match the need of a diverse student population, whether it is dominated by students majoring in Mathematics, Data Science, Computer Science, Information Technology, or a mix of these. We also include some course evaluation results from the first iteration of the course.

2010 AMS Classification: 97P50, 97P40.

Keywords: CS 0, CS 0.5, non-major courses, SageMath, Python, introductory programming language, first-year computing, mathematics in computing.

1. Introduction

In 2018, for the second consecutive year, Python was considered the most popular programming language [2]. Being perceived as one of the easiest programming languages to learn by beginners, it is no wonder that it has become widely adopted in introduction to programming courses all across the U.S. [5]. It was also very well received in the areas of Machine Learning and Data Science [8]. Some even consider Python "an integral part of data science" [12] while others say it is "the top language to learn if you are looking to skill up in areas around machine learning" [3]. In the field of Mathematics, a powerful Python-based tool that is getting more and more popular is called SageMath." It builds on top of many existing open-source packages: NumPy, SciPy, matplotlib, Sympy, Maxima, GAP, FLINT, R" [11], it is a free open source tool, and it is already adopted in courses such as Calculus, Linear Algebra, Abstract Algebra, Combinatorics, Data Structures, Linear Programming, and several other ones. A simple google search on the terms 'SageMath' and 'Syllabus' will provide you with ample of such courses (currently more than 10,000 results are being returned). Because of this wide range of courses that adopted SageMath, the tool has a special appeal to Mathematics students who have been exposed to it. In addition, the powerful R can also be used from within SageMath [10], hence students majoring in Data Science could also benefit from SageMath. And, for a focus on user interface applications, one could use SageMath to create interacts and animations. What we propose in this paper is an introduction to programming course using SageMath. Since SageMath is Python-based, instructors with could easily adjust a Python curriculum to a SageMath one. Moreover, given the large number of packages available in SageMath, such a course

Email address: rmezei@stmartin.edu (R. A. Mezei)

could be tweaked to match the need of a diverse student population, whether it is dominated by students majoring in Mathematics, Data Science, Computer Science, Information Technology, or a mix of these. The reason why we first created such a course was the desire of several junior and senior level undergraduate Mathematics and Mathematics Educations students to learn programming using SageMath, a tool they had already been exposed to in some upper-level Mathematics courses. Given the fact that Python is considered one of the easiest programming languages to learn by beginners, and that SageMath is a Python-based tool, we believe that an Introduction to Programming using SageMath course could be introduced at the freshmen level, and then students could use the knowledge acquired in here for upper level courses (such as Calculus, Numerical Analysis, etc). Based however on the experience we had when we first created this course, junior and senior students who already got some exposure to SageMath (from other upper-level non-Computer Science courses) could come into this course motivated either to ”learn about computer programming” in general or specifically ”learn more about SageMath”. As such, we believe that non-Computer Science students of all levels, especially Freshman and Sophomore, could benefit substantially from an introduction to programming using SageMath course.

2. Existing Curriculum

As noted in the previous section there are many institutions that already teach introduction to programming using Python. As such there are many resources available to support introductory Python curricula, some better suited than others, depending on your intended audience. See for example: [4], [7], and [13]. Similarly, there are several SageMath introductory courses with a focus on how to use SageMath as a computational tool, rather than focusing on the programming side of it. These mostly focus on applications to specific areas such as Calculus, Abstract Algebra, Combinatorics, etc. Some great references are: [1] and [14].

What we propose in here is an Introduction to Programming curriculum, just like the ones that already exist and use Python, C++, Java, C#, etc., but we would instead be using SageMath. We offered such a course in 2016 and, based on the research we conducted, we believe this is the only such course created so far.

3. Why SageMath?

As noted above, the initial reason for choosing to create such a course was the desire of several undergraduate Mathematics and Mathematics Education students to learn about programming using SageMath, because they have already seen it as a great computational tool is some Mathematics courses. As we created the course, it quickly became apparent that SageMath could not only be used for Math students, but also for all non-Computer Science majors, provided the applications are carefully chosen to raise students’ interest. As such we chose to teach the typical introduction to programming curriculum but used SageMath as a tool.

Since SageMath is a Python-based tool, the students would essentially get to learn Python, but they will also have access to many packages available in SageMath (some included by default). This way students would get to learn programming concepts and apply them using graphing, interacts, animations, and even symbolic computation. This could allow an instructor to easily shift the focus of the course projects and applications, so they would match the need of a diverse student population while maintaining the same overall structure expected from introductory programming courses.

It may be worth mentioning that SageMath can be used as a standalone application on a computer, web-based, or even cloud-based. This makes it available to all students regardless of the operating system used or the computing power available.

4. Course Objectives

The course objectives were set to be similar to what one would expect from an introductory programming course in Python, Java, or C#. They included (see also [9] and [6]):

- provide students with enough knowledge about computer organization (software and hardware) and data representation,
- help students become familiar with programming in an IDE (either host-based, web-based, or cloud-based),
- teach students learn how to design and implement basic programming constructs that include statements, control structures, methods, and lists,
- train students good programming style and help them produce programs with some abstraction,

- help students understand of each of the following concepts: abstraction, debugging and program correctness, functions and objects, recursion, efficiency, reusability,
- introduce students to some GUI elements, including interacts and animations,
- help students learn to use SageMath to solve problems of interest to them.

5. Course Outline

Below is a brief outline for an introduction to programming using SageMath course that was taught to a class where majority of students were Math majors:

- Introduction to Computer systems, Hardware and Software, Data Representation.
  - This module provides a brief introduction to computer systems, data representations (of integers, character sets, colors, images), programming languages, IDEs.
  - This part introduces the SageMath environment and instructions on how to access SageMath (host-based, web-based, or cloud-based).
- Comments, Variables, Operators, Introduction to Lists:
  - In here the following are introduced: comments, variables, symbolic variables, operators, introduction to lists.
- Input and Output:
  - Here we introduce raw input, exception handling, input/output using files.
  - Then we introduce SageMath interacts, such as input boxes, sliders, selectors, check boxes, and range sliders, that allow introduction to GUI programming.
  - We finish this module with a discussion on output formatting, and creation of animations.
- Introduction to SageMath functions:
  - Here we introduce symbolic constants.
  - Then we introduce library functions, including graphs and plots.
- Boolean Expressions, Relational Operators, If Statements:
  - Here we introduce boolean expressions.
  - Then we introduce if statements, and flowchart representations.
  - This module explores if, if - else, if - elif - else.
- Loops and Strings:
  - Here we introduce for and while loops.
  - Then we introduce string operations.
- Functions:
  - Here we introduce functions and symbolic functions.
  - Then we introduce string operations.
  - Then recursive functions are defined, several examples are provided, and fractals are introduced.
  - Animated plots are also introduced in here.
- Algorithms:
  - We finish this course with an introduction to searching and sorting algorithms.

Using interacts, the applications can be given a nice graphical user interface, that is intuitive, and easy to grasp.

6. Conclusions

Before we conclude this work, we would like to share some selected course evaluation metrics anonymously submitted by students at the end of the semester. Unfortunately, at the end of the first iteration of this course, the author of this paper took a two-year break from Academia, to pick up an Industry job, and hence he was unable to follow up with a second iteration of such a course in order to validate those numbers. As such we hope that the metrics below would serve as an insight regarding the potential of success of the proposed course and that other faculty members could follow up with a more longitudinal study.

From these results (see Tables 1-4) one can see that it appears that the instructor managed to stimulate students to think critically but failed to increase the interest of all students in the class. Furthermore, it appears that one student didn't feel that the he/she has advanced his course knowledge and that one student (probably the same) did not feel that

Table 1. Stimulated me to think critically

N	Avg	Std. Dev.	Str. Agree	Agree	Disagree	Str. Disagree
7	3.7	0.5	71%	29%	0%	0%

Table 2. Interest Increased

N	Avg	Std. Dev.	Str. Agree	Agree	Disagree	Str. Disagree
7	3.6	0.8	71%	14%	14%	0%

Table 3. Knowledge of course subject advanced

N	Avg	Std. Dev.	Str. Agree	Agree	Disagree	Str. Disagree
7	3.4	1.1	71%	14%	0%	14%

Table 4. Content relevant to me

N	Avg	Std. Dev.	Str. Agree	Agree	Disagree	Str. Disagree
7	3.4	1.1	43%	43%	14%	0%

Table 5. Student Gender

N	Female	Male
7	43%	57%

the content was relevant to him/her. As such, the instructor should try to more closely match the course applications with the students' interests and/or major.

A notable success (as seen in Table 5) is the fact that although the course was an elective, it somehow managed to attract a good proportion of female students. One potential reason for this is the fact that SageMath was used in some upper-level Math courses and as such Math Education students seemed particularly interested in getting to learn programming by means of a tool they have already been exposed to: SageMath. Overall, we feel that SageMath can successfully be used as an introductory programming course to non-Computer Science majors. Students would not only get to learn a Python-like syntax, but also could greatly benefit from the Math libraries that come with SageMath and hence be exposed to a wide range of applications. Depending on each student's interests, the course could focus more on the Math applications (from advanced numerical and symbolic computations to simple graphing), data science applications (by making use of R from within SageMath code), GUI applications (through interacts and animations) or a mix of these. This fluidity could help the instructor better reach out to an increasingly diverse student population.

For small liberal arts schools, where class enrollments may change significantly from one semester to another, some departments could choose to combine the course proposed in here with a CS 0 or CS 0.5 course that includes Computer Science majors.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this article.

References

[1] Bard, G.V., *Sage for undergraduates (online version). http://www.gregorybard.com/sage.html*, retrieved Dec. 2018. 2

[2] Cass, S., *The 2018 top programming languages. https://spectrum.ieee.org* , 2017. 1

[3] Davis, J., *5 top languages for machine learning, data science. https://www.informationweek.com*, retrieved Dec. 2018. 1

[4] Gaddis, T., Starting Out with Python (4th Edition), 2017. 2

[5] Guo, P., *Python is now the most popular introductory teaching language at top U.S. universities. https://cacm.acm.org* , retrieved Dec. 2018. 1

[6] Jochen, M., *Introduction to computer programming course syllabus - fall 2012. http://quantum.esu.edu/ mjochen/Teaching/CPSC130/12f/*, 2012. 4

[7] Lutz, M., Learning Python, 5th Edition, O'Reilly Media, 2013. 2

[8] Puget, J.F., *The most popular language for machine learning is... https://www.ibm.com*, retrieved Dec. 2018. 1

[9] Rufinus, J. and Kortsarts, Y., *Teaching an introductory programming course for non-majors using Python.*, In Proc ISECON, v22 (ColumbusOH), 2005. 4

[10] Stein, W.A. et al., *Sage quickstart for statistics. http://doc.sagemath.org*, retrieved Dec. 2018. 1

[11] Stein, W.A. et al., *SageMath - open-source mathematical software system. http://www.sagemath.org*, retrieved Dec. 2018. 1

[12] Voskoglou, C., *What is the best programming language for machine learning? https://towardsdatascience.com*, retrieved Dec. 2018. 1

[13] Zelle, J., Python Programming: An Introduction to Computer Science, 3rd Ed., Franklin, Beedle & Associates, 2016. 2

[14] Zimmermann, P. et al., *Computational mathematics with SageMath. http://www.sagemath.org*, retrieved Dec. 2018. 2