IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 August 2022, accepted 28 September 2022, date of publication 4 October 2022, date of current version 12 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3211973

RESEARCH ARTICLE

Developing Computing Competencies Without Restrictions

CRISTIAN VIDAL-SILVA[1], NICOLÁS A. BARRIGA[1], (Member, IEEE), FRANCO ORTEGA-CORDERO[2], JAVIERA GONZÁLEZ-LÓPEZ[2], CLAUDIA JIMÉNEZ-QUINTANA[2], CLAUDIA PEZOA-FUENTES[3], AND IVÁN VEAS-GONZÁLEZ[3]

[1]Facultad de Ingeniería, Escuela de Ingeniería en Desarrollo de Videojuegos y Realidad Virtual, Universidad de Talca, Talca 3460000, Chile
[2]Department of Ingeniería Civil Informática, Escuela de Ingeniería y Negocios, Universidad Viña del Mar, Viña del Mar 2520000, Chile
[3]Departamento de Administración, Facultad de Economía y Administración, Universidad Católica del Norte, Antofagasta 1270709, Chile

Corresponding author: Cristian Vidal-Silva (cvidal@utalca.cl)

ABSTRACT The information society represents a great revolution. Computing programming is a relevant competence nowadays for everybody, regardless of educational background. However, traditional programming languages consider syntax barriers that complicate their adoption and usefulness for beginners. Python is an exception for its open-source, cross-platform nature and syntax simplicity, which facilitate the development of algorithmic thinking and dissemination of programming solutions. Several Python extensions support modern functionalities such as web development, videogame, and machine learning, making it one of the most used programming languages. Google Colab or Colaboratory facilitates the online learning and development of Python solutions. This article presents positive academic experiences of Chilean students of majors from two Chilean universities, a traditional university in the north and a private university in the middle of Chile, using Google Colab to develop programming competencies remotely for the Covid pandemic. We highlight the promising results obtained for basic programming and operating system programming subjects, which motivate us to use Python and Google Colab widely, not only in university contexts. We expect to continue developing programming competencies using Google Colab and Python. The main limitation encountered in this experience is the internet connection requirements for online education. However, it does not represent an issue for education in developing and developed countries. Google Colab permits the development of highly demanded competencies worldwide at home, only with internet access and a web browse, an excellent motivation for learning for all students regardless of age and academic level.

INDEX TERMS Programming competencies, operating systems, python, Google colab, colaboratory, online education.

I. INTRODUCTION

Developing algorithmic and programming competencies is essential in the information society [1], [2]. Algorithms and programming courses often include algorithm design and analysis, data structures, software engineering, Web development, and intelligent systems. Multiple programming languages exist, and almost all of them are text-based such as C, C++, and Java [3]. Traditional text-based programming languages impose a set of syntax and semantic rules that are a significant barrier for beginners. The use of animated block-based programming languages such as Scratch permit notably reducing syntax mistakes in the teaching-learning process. Still, those languages do not transparently address some algorithmic issues such as networking, concurrent and parallel computing that limit the development of algorithmic competencies [4]. Python is a text-based programming language that helps addressing those issues. As Li et al. [5] argue, for simplicity, easy of learning, being open- source, its strongly portability and the existence of rich libraries, Python nowadays is a popular programming language to teach programming courses in educational institutios worldwide. Hence, the Python programming language represents a great revolution. Code 1 and 2 show the classic 'Hello World'

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Afzal.

example in C and Python programming languages, respectively. We can appreciate in this example that the Python solution considers only one line concerned with its primary task. Code 3 and 4 illustrate a programming example to ask for the name, age and year of birthday, and to show the data.

As Futsche [6] remarked, algorithmic thinking can start early and its orientation should be on the thinking ability. Traditional programming languages such as C, C++, PHP and Java usually apply similar syntax and semantic rules and they seem adequate for developing computing science competencie [7]. Nonetheless, those rules usually make students uncomfortable and distract them from understanding the purpose and usefulness of algorithms and programming [8].

First programming courses present algorithmic problems to reason and solve them sequentially. Therefore, parallel algorithmic reasoning seems non-natural, even though human beings typically apply it in daily life [9]. For example, when working team distributes tasks to carry out, or when we parallelize traditional actions, such as walking and talking to someone at the same time. Those are examples of parallel tasks. Traditional multitask programming subjects review the execution of multiple tasks by processes and threads. Code 5 and 6 illustrate code in which a main process creates a child process in C and Python respectively. Likewise, Code 7 and 8 illustrate code in which a main thread creates a child thread in C and Python respectively.

Previous code examples in Python confirm its easy-to-learn syntax that makes it a simple programming language to work with and emphasizes its readability. Nonetheless, python is a software tool and we need to install different software packages to work and successfully use it. Google Colab or Collaboratory allows developing and running Python solutions without installing and configuring Python, with free access to multiprocessing servers for the execution of parallel solutions, and with Google's ease of sharing content, as well as facilitating interaction and integration with other Google solutions [10]. We Almost all presented Python code are in Google Colab.

A. PROBLEM STATEMENT, GOAL AND CONTRIBUTIONS

Vidal et al. [11] highlight that traditional programming languages demand a high level of abstraction regarding syntax rules and algorithmic thinking abilities. In addition to problem-solving skills, the science of programming also takes into account abstract concepts that are closely related to programming mechanics [12]. We can describe usual human beings activities as algorithm [13], and algorithms increasingly rule our social life [14]. Hence, domaining and applying programming languages rules seem to be relevant issues for successfully developing programming competencies. Although they are a set of sequential steps without conditions or cycles, previous code examples validate that the Python programming language is an exception because it simplifies syntax barriers.

Currently, developing programming competencies is relevant for education [15], [16] even more in early age [17], [18]. Successful experiences using Python for developing programming competencies already exist in the computer science area such as [19], [20], [21], and [22]. Furthermore, Python has also successfully applied in non-computer science domains with the same goal; that is, to develop programming competencies [21], [23], [24]. Given the current usefulness and experiences of using Google Colab for building Python applications interactively and developing programming skills in educational scenarios [25], [26], [27], [28], the main goal of this article is to answer the following research questions.

RQ1 [Impact of Google Colab and Python in university students] How can Python and Colab affect the development of programming competencies in students of computer science and non-computer science majors? Although the case studies are with Chilean students, obtained results and applied experiments could be useful for other countries.

RQ2 [Students' Perception] How did students of both case studies perceive the benefits of using Python for developing programming competencies?

RQ3 [Motivation to Continue Using Python and Colab] How motivated are students to continue developing more advanced programming competencies and applying them using Colab and Python in the future?

This article describes two academic experiences using the Colab and Python to develop programming competencies with students of a first programming course of a computer science major from the Viña del Mar University (VMU) in the middle of Chile, and students of a first programming course of a non-computer science major from the Catholic of the North University (CNU) in the north of Chile. Specifically, Computer Engineering at VMU and Information and Management Control Engineering at CNU. Material of both programming courses, homework, and students score are available in the GitHub repository.[1] The results obtained demonstrate that the development of algorithmic and programming competencies by Python and Google Colab is highly feasible, with the simplicity of using online tools without requiring software installation. Hence, the followings are the main contributions of this paper:

- First, to confirm the applicability of Python and Google Colab for developing algorithmic-thinking and programming competencies in students regardless their main study focus, that is, not only for computer-science students.
- Second, to demonstrate the usefulness of Python and Google Colab in developing algorithmic-thinking and programming competencies in remote education.
- Third, to project the applicability of Python and Google Colab for developing algorithmic-thinking and programming competencies everywhere without restriction access.

[1] https://github.com/cvidalmsu/AcademicReport_CNU_VMU

CODE TABLE 1. 'Hello World' C and Python code.

```c
#include<stdio.h>

int main(){
   printf("Hello World\n");
}
```

Code 1: 'Hello World' C code

```python
print("Hello World\n");
```

Code 2: 'Hello World' Python code

CODE TABLE 2. 'Input/Output Java and Python code examples.

```java
import java.util.*;

public class program{
  public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.println("Name? ");
    String name = sc.nextLine();

    System.out.println("Age? ");
    String age = sc.nextLine();

    System.out.println("Year of birth? ");
    String year = sc.nextLine();

    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
    System.out.println("Year of bityh: " + year
     );
   }
}
```

Code 3: Input/Output Java code example

```python
name = input("Enter your name: ")
age = input("Enter your age: ")
year = input("Enter your year of birth: ")

print("Name: " + name);
print("Age: " + age);
print("Year of birth:" + year);
```

Code 4: Input/Output Python code example

CODE TABLE 3. 'Hello World' with Processes in C and Python.

```c
#include<unistd.h>
#include<stdio.h>

int main(){
   int childProcess = fork();

   if (childProcess == 0)
      printf("I'm the Parent
              Process\n");
   else
      printf("I'm the Child
              Process\n");
}
```

Code 5: 'Hello World' with Processes in C

```python
from multiprocessing import Process

def toSayHello():
   print("I'm the Child Process\n")

childProcess = Process(target = toSayHello)
childProcess.start()

print("I'm the Parent Process\n")
```

Code 6: 'Hello World' with Processes in Python

CODE TABLE 4. 'Hello World' with Threads in C and Python.

```c
#include<pthread.h>
#include<stdio.h>

void *toSayHello(void *args){
    printf("I'm the Child Thread\n");
}

int main(){
   pthread_t childThread;

   pthread_create(&childThread. NULL,
                  toSayHello, Null);
    pthread_join(childThread. NULL);

    printf("I'm the Parent
              Thread\n");
}
```

Code 7: 'Hello World' with Threads in C

```python
from threading import Thread

def toSayHello():
   print("I'm the Child Thread\n")

childThread = Thead(target = toSayHello)
childThread.start()

print("I'm the Parent Thread\n")
```

Code 8: 'Hello World' with Threads in Python

The rest of this paper is organized as follows. Section II describes and exemplifies the use of Python and Google Colab. Section III details the competencies and experiments in the programming courses of UVM and UCN students during 2020 using traditional programming languages and 2021 using Python and Google Colab. Section IV gives

detail and discusses the academic results obtained by using Python and Google Colab as a teaching-learning methodology. Section V presents similar studies regarding the benefits of using Python and Google Colab to develop programming competencies. Section VI details a few practical issues. The paper concludes by summarizing the benefits of our academic experience and motivation for continuing applying Python and Google Colab in the current and future years.

II. PYTHON, MULTIPROGRAMMING AND GOOGLE COLAB

A. PYTHON

The work of Lutz [29] highlights that software quality, developer productivity, program portability, support libraries, component integration and enjoyment are factors to answer the question Why do people use Python? According to their teaching experience, the authors of this work remark that Python's relevant characteristics for novice users are its simple and readable syntax and highly coherent programming model. As Abdulkareem et al. [30] remark, Python is a very user-friendly and the most straightforward programming language compared to Java, C++ and Javascript. The work of [31] signals that Python is more efficient than Java and R for solving machine learning problems. Code 2 and 4 exemplify the Python syntax simplicity.

As Javed et al. [32] report, Python is a highly demanded programming language. It is simpler to learn and more understandable for mature programmers, such as students with a programming background. The design of Python permits giving complete solutions in the world of programming [20] even more for the availability of open resources and communities for learning. As the work of Srinath [33] describes, some of the main features of Python are its simplicity, portability, open-source nature, extensibility and embebebility, high-level interpreted language, standard libraries, object-oriented, and variability of programming solutions (system, GUI, network and internet, database, games).

Although Python would permit the development of programming competencies in novice students, it is software that requires an installation process that is not always a simple task [34]. Usually, novice users face significant challenges while installing and using Python tools and packages because of some missing requirement or version incompatibility [35]. Those issues are common to all programming languages. Considering the positive factors for using Python, Google offers Colab permits solving this issue since we can create and test Python solutions everywhere without restrictions.

B. MULTIPROGRAMMING

Processes and threads are computing tasks in modern computing systems that share resources such as available processors, and memory [36], [37]. The main difference between processes and threads is that a process represents a program in execution, and a thread is not more than a segment of a process; that is, a process contains at least one thread. Figure 1 illustrate a multitask system running n processes; each one contains multiple threads (three threads in this case).

Parallel systems are systems that can run multiple tasks at the same time [38], and, thanks to the current multi-core technology, they are standard systems nowadays. Hence, current computing systems require the development of parallel computing skills from the earliest education levels, a worthy goal for the academy in computer science and information to go hand in hand with advances in technology.

If the primary goal of a course is understanding concepts and applying them for solutions development, Python is an adequate programming language. Currently, Python seems a preferred option for scientific computing [39]. Nonetheless, Python does not permit running several threads at once, and multiprocessing is the solution. Multithreading and multiprocessing represent similar concepts (both look for multitasking). The simplicity of Python permits applying both without significant syntax differences as Code 6 and 8 exemplify.

One of the primary goals of multitask programming is to develop efficient computing solutions. As Kumar et al. [40] signal, coordinating the access of shared resources is one of the main issues in multitasking computing, a problem that Operating Systems courses aim to teach.

C. GOOGLE COLAB

Google Colaboratory, commonly referred to as Colab, is a cloud platform for sharing machine learning research and education that is built on Jupyter Notebooks. It provides free access to a powerful GPU as well as a runtime that is fully equipped for deep learning [41].

Colab enables the creation of notebooks based on Jupyter, a browser-based, open-source application that combines interpreted languages, libraries, and visualization features [42], [43]. Each notebook is made up of a number of cells, each containing markdown or script code, and the output is embedded within the content. Text, tables, charts, and images are typical outputs. As Randles et al. argue [44], using this technology makes it easier to share and duplicate scientific studies.

Colab enables working with notebooks based on Jupyter like a Google Docs object: users can share and collaborate on the same notebook. Colab provides Python 2 and 3 runtimes preconfigured with essential machine learning and artificial intelligence libraries such as TensorFlow, Matplotlib, and Keras. The Virtual Machine (VM) deactivates after some time, and all user data and settings are lost, although one can preserve and transfer files from the virtual machine's hard drive to the user's Google Drive account. This Google service provides a GPU-accelerated runtime, also fully configured. Google Collaboratory infrastructure is on the Google Cloud Platform. Figure 2 exemplifies actions for sharing the 'Hello World' example.

III. METHODOLOGY AND RESEARCH DESIGN

This section reviews the main competencies and learning objectives in the Programming course for the Information and

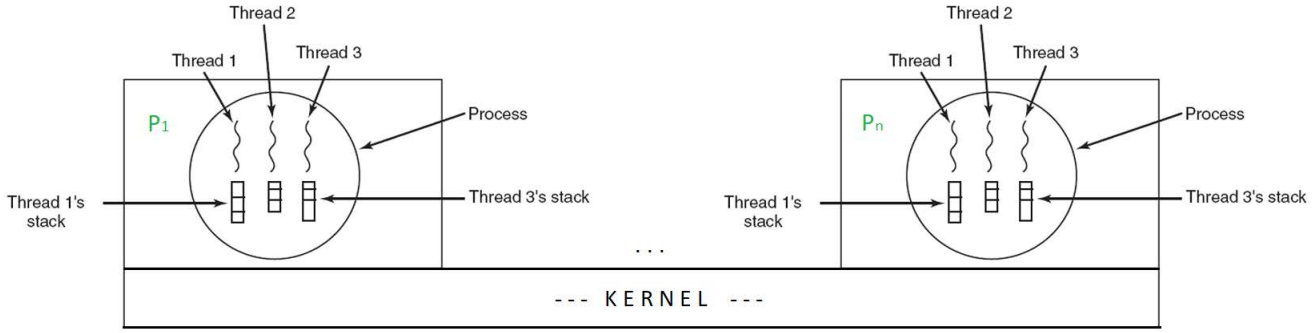


FIGURE 1. Proccess and threads in a multitask computing system.

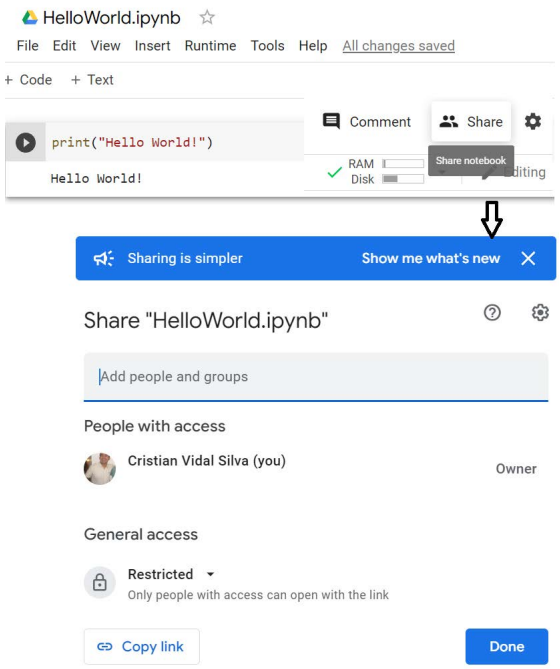


FIGURE 2. Sharing 'Hello World' notebook in Colab.

Management Control Engineering major at the CNU and the the Operating Systems course for the Computer Engineering major from the VMU.

Considering the competencies and learning objectives of Programming at CNU and Operating Systems at VMU, the research questions and the main goal of this article, both courses review an introduction to Python and Colab in different times concerning their learning objectives and schedule. These courses are in the first semester of each academic year, that is, sixteen weeks from March to July. This article report our experience in both courses during 2020 and 2021. In 2021 we applied Python.

A. PROGRAMMING COMPETENCIES AND LEARNING OBJECTIVES

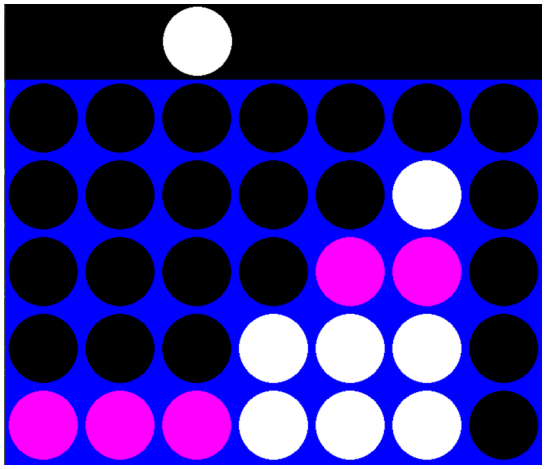
The Programming course at the CNU defines the following competencies and learning objectives:

1) To develop technical skills in Information Technology applied to business.
   - 1.1 Understand the fundamentals of computer programming.
   - 1.2 Develop analytical thinking to deal with complex situations in the business environment.
2) To develop a systemic vision to solve computational problems.
   - 2.1 Develop the ability to participate in developing software projects according to each need in the company.
   - 2.2 Understand how to use advanced objects in programming.
3) Capacity for teamwork.
   - 3.1 Management of obligations related to group projects.
   - 3.2 Evaluating oneself and the job done in collaboration.

The Programming course contents at CNU contain two units. The first unit reviews variables and data types, input/output instructions, conditional and iterations structures, and the division of problems and modularization of solutions by functions. The second one reviews the main concepts of data structure and object-oriented programming.

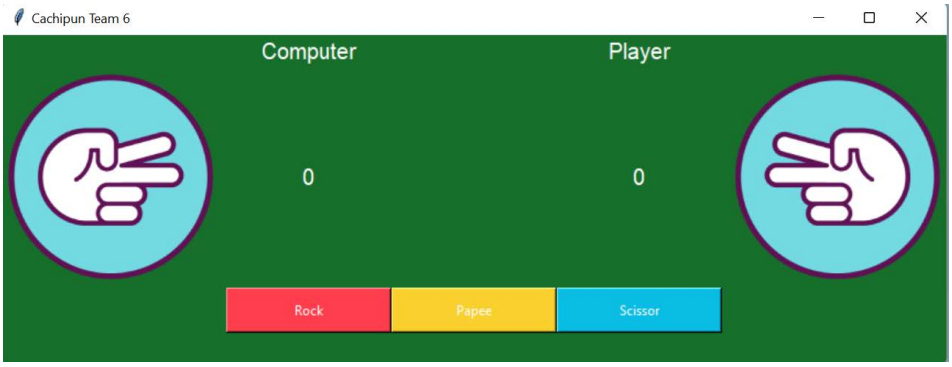
The course was for 20 students in 2020 and 31 in 2021. We applied a practical approach by asking for homework every two weeks in both years. Regarding the programming language in those years, in 2020, we applied the PSeInt tool[2] for three weeks to start working on programming using pseudocode in Spanish. After, we started using Java with the Eclipse IDE.[3] Although students were familiar with elementary programming concepts such as variables, using functions for input and output, and some math operations, students experienced issues working with Java and Eclipse while developing the workshops. In 2021, we applied Python in the whole course because we decided that Python's syntax simplicity does not require starting with a pseudocode tool. We can appreciate that Python is applicable in any course considering programming competencies. Hence, writing pseudocode in paper is not common nowadays [45]. The final project of the 2021 course considers using the PyGame package for developing games [46]. Figure 3 shows a few

[2]http://pseint.sourceforge.net/
[3]https://www.eclipse.org/downloads/



(a) 'Connecta4' game in Python with PyGame

(b) 'Hanged' game in Python with PyGame



(c) 'Cachipún' game in Python with PyGame

FIGURE 3. Resulting game examples with Python and PyGame in the Programming course in 2021.

game results produced by teamwork in the Programming course at CNU in 2021.

B. OPERATING SYSTEMS COMPETENCIES AND LEARNING OBJECTIVES

The Operating Systems course at the VMU defines the following competencies and learning objectives:

1) To conceptualizes a modern computer system's fundamental hardware and software components.
   - 1.1 Definition of computer systems architecture for companies.
   - 1.2 Definition of a server-style computing system's basic hardware and software components.
2) To propose concurrent prototype solutions for both hardware and software in an organization.
   - 2.1 Description of processes, use of command line user interface and file system of a Linux-based operating system.
   - 2.2 User system domain and input/output of a Linux-based operating system.

Regarding the course content, the Operating System course started using Python from week eight, that is, in the middle of the course, with the advantage that students had already developed programming competencies. In programming courses of the computer engineering major at VMU, students start learning C and Java. Hence, learning Python is simple, although developing multiprocessing and multiprogramming competencies requires additional abstractions. This course reviews processes and threads synchronization, and resource sharing.

In 2020, the course was for 13 students, whereas it was for ten students in 2021. The course plan considers three main projects with advances in each of them. In 2020, we applied C in Linux operating system to develop a practical understanding of multiprogramming concepts, although issues existed regarding the adequate synchronization of tasks and resource sharing. In 2021, we wanted to change the programming approach to appreciate eventual changes. Because students were familiar with elementary and more advanced programming concepts such as variables, data structures, functions and object-oriented programming, we started exemplifying the 'Hello World' with processes and threads in C. In the next class, we reviewed programming elements of Python, and students were surprised by its simplicity. We reviewed the 'Hello World' in the same class with Processes and Threads in Python. After that, we continued applying Python. Again,

TABLE 5. Groups and solved problems in the Operating System course in 2021.

Score	Evaluation
Group 1	Readers and Writers
Group 2	Sleeping Barber
Group 3	Dinning Philosophers
Group 4	Producer and Consumer

TABLE 6. Academic results in the Programming (Prog) and Operating Systems (OS) courses during 2020 and 2021.

Prog 2020		OS 2020	
Average	4,345	Average	5,292
Median	4,300	Median	5,400
Mode	4,300	Mode	5,700
σ	1,054	σ	0,751
20 students		13 students	
Prog 2021		**OS 2021**	
Average	5,489	Average	6,185
Median	5,350	Median	6,245
Mode	5,040	Mode	6,370
σ	0.739	σ	0.199
31 students		12 students	
Prog score differences		**OS score differences**	
Average %	26.33	Average %	16.87

we appreciate that Python is suitable for all computer science courses. Table 5 describes the classic multiprogramming problem solved in Python by each teamwork of the Operating Systems course at VMU in 2021.

We can appreciate differences in the main focus of both courses and engineering majors: Operating System at VMU follows a computer science approach; and Programming at CNU follows an information science approach.

IV. RESULTS

A. ACADEMIC RESULTS

We compared students' academic results of the Programing course at UCN and Operating Systems at UVM in 2020 and 2021. Table 6 summarizes the statistics of all students' final grades in both courses. As previously indicated, throughout 2020, the Programming course applies Java for developing programming competencies, whereas the Operating System course uses C. In 2020, we introduced Python in both courses. As Tupac-Yupanqui et al. [47] mentioned, Python represents an evolution from other text-based programming languages such as C, C++, and Java mainly for its syntax rules. The obtained results confirm that Python permits the successful development of programming competencies currently without restrictions. The academic institution grades in Chile are on a scale of 1,0 to 7,0 points, where 1,0 is the minimum score, 7,0 is the maximum score, and 4,0 is the minimum for approval.

Because qualification scores are independent (although working teams exist), and both courses exemplify random samples, we can apply a student's t-test [48] to analyse the academic performance for both classes in 2020 and 2021. Our null hypothesis is the same academic performance for each analysis, with a significance level $\alpha = 0.05$. We evaluated t-student test for equal and unequal variances.

- Student's t-test for Programming at CNU in [2020-2021].
  - Assuming equal variances
    - t Stat: 4.593094611
    - t Critical one-tail: 1.676550893
    - t Critical: 2.009575237
  - Assuming unequal variances
    - t Stat: 4.253743838
    - t Critical one-tail: 1.695518783
    - t Critical: 2.039513446
- Student's t-test for Operating Shystems at VMU in [2020-2021].
  - Assuming equal variances
    - t Stat: 4.511535907
    - t Critical one-tail: 1.708140761
    - t Critical: 2.059538553
  - Assuming unequal variances
    - t Stat: 4.36986425
    - t Critical one-tail: 1.761310136
    - t Critical two-tails: 2.144786688
- Student's t-test for Programming at CNU and Operating Systems at VMU 2021.
  - Assuming equal variances
    - t Stat: 7.582146364
    - t Critical one-tail: 1.693888748
    - t Critical two-tails: 2.036933343
  - Assuming unequal variances
    - t Stat: 5.248325089
    - t Critical one-tail: 1.683851013
    - t Critical two-tails: 2.02107539

With the obtained results, we reject the null hypothesis: academic performance is not the same. Table 6 shows significant academic performance improvements in 2021 regarding 2020: a 26.33% academic grade of improvement for Programming at CNU and a 16.87% of improvement for Operating Systems at VMU. Furthermore, we can appreciate a more significant impact of applying Python in the academic performance improvement for Programming students; that is, for students who did not get previous programming courses.

Professors were the same in 2020 and 2021, and the academic evaluation tools were similar: laboratory homework and programming projects aiming to develop the same competencies. Both courses finish the academic semester with a final project where students give an oral presentation to validate the teaching-learning results. Thanks to Python's benefits, final projects were of a higher quality. For example, at CNU, students successfully developed games (although without using Colab).

B. SATISFACTION RESULTS

To measure students' satisfaction [49] in 2021, in both courses, we conducted surveys about using Python (Table 7). The survey contains 4 two block of questions. The first two consider three items, each one for measuring the level of frustration and satisfaction by a Likert scale from 1 to 10 for extremely [frustrating | dissatisfying], and very [stimulated | satisfying], respectively. The 3rd block contains three options to indicate the reached familiarity the software used in the course, that is, Python. The final block permits communicating, in a few words, additional ideas regarding the experience in the course. This survey permits answering the research questions of this paper.

The following lines describe the survey and the positive results obtained for the Programming at CNU and Operating Systems at VMU. Those results motivate us to continue applying Python for developing computing and information science competences that involve programming tasks. We classified the survey scores in ranges in the first two questions block for their summary and illustration purposes(8).

- **Survey results**.
  - Frustration level.
    - (i) 35.48% found stimulating, and 45.16% found very stimulating the studied material.
    - (ii) 22.58% found stimulating, and 67.74% found very stimulating their working group.
    - (iii) 25.81% found stimulating, and 64.52% found very stimulating the applied technology, that is, the Python programming language and Colab environment.
  - Satisfaction level.
    - (i) 30.23% were satisfied, and 62.80% were very satisfied regarding the studied material.
    - (ii) 20.93% were satisfied, and 69.77% were very satisfied regarding the working group.
    - (iii) 20.93% were satisfied, and 69.77% were Very satisfied regarding the applied technology, that is, the Python programming language and Colab environment.
  - Python and Colab familiarity.
    - (i) 95% of the students indicated that they were familiar with Python and Colab for future application, and 5% indicated that they were not familiar with Python but they were that for other programming languages.
  - Communication words.
    - (i) Common words were simplicity, easyness, and fun programming.

Figures 4 and 5 depict the mentioned results. In Figure 4, the green area is for Very [stimulating | satisfied] class, light green area is for [Stimulated | Satisfied] class, white area is for neutral class, orange area is for [Frustrating | Disatifying] class, and the red area represents Extremely [frustraing | disastifying] class for the obtained responses in the survey. In summary, a high satisfaction exists in students for the use of Python and Colab for developing computing compentences either in Programming and Operating Systems courses. Thus, due to the project's success, professors at the Programming course at CNU and Operating Systems at VMU will continue using Python and Colab.

Figure 4 shows that students of Programming at the CNU and Operating Systems at VMU consider Python and Colab stimulating tools for developing computing competencies: over 95% indicated that Python and Colab were stimulating and very stimulating tools. Moreover, students also indicated that using Python and Colab represented a high satisfaction level: over 93% mentioned that using those tools was satisfying and very satisfying. Thus, RQ1 and RQ2 are fully satisfied, considering the results obtained. Likewise, RQ3 is also fully satisfied because Figure 5a indicates that students are highly familiar with Python and Colab for future use (95%). Regarding the word cloud of Figure 5b, we can appreciate that While, Python, SimpleCode, Thread, Programming, Return, AI, and MachineLearning are the most mentioned words. We expected words directly connected to the Programming and Operating Systems courses, like the first six. The attraction of students to Python and its usefulness with new computing approaches would be the source of the last two words we think. Table 9 summarizes the words occurence.

V. DISCUSSION

As Tupac et al. [47] highlight, many countries provide programming courses and workshops to help children learn algorithms and programming abilities early on. Making the teaching-learning process engaging and applicable is one of the fundamental problems of acquiring programming abilities. Nonetheless, successful development of algorithmic thinking and programming competencies with traditional programming languages can be ineffective [6].

Applying Python and Colab demonstrates its adequacy for developing computing competencies without restrictions. However, Colab, since it is a web tool, seems inadequate to work with external devices such as input and output hardware devices [50]. In those situations, different Integrated Development Environments (IDEs) exist for free access that students can use for those purposes. Colab is a great tool for familiarizing yourself with elementary computing and programming tasks. Moreover, for its simple syntax rules, open-source nature, applicability in different contexts (artificial intelligence, robotics, and games, among others), currency worldwide, computing advances and high integration with other solutions and platforms, undoubtedly, Python is a great programming language option nowadays.

Even though multiple online platforms for writing and testing programs in different programming languages exist (e.g. reply[4] and W3 School,[5] as we already highlighted, Google Colab, in addition to permitting writing and testing Python code, preserves the work functions of Google Drive for sharing and teamwork. Furthermore, Colab enables the

[4]https://https://replit.com/languages/python3
[5]https://www.w3schools.com/python/python_compiler.asp

TABLE 7. Academic survey for students in 2021.

Name or Id _____
I. On a scale of 1 (extremely frustrating) to 10 (not frustrating at all) rate your frustration level with elements of the course. Please write the rating in the space provided. **Extremely frustrating** **Very stimulating**
1 2 3 4 5 6 7 8 9 10
__ studied material __ working group __ technology
II. On a scale of 1 (extremely dissatisfying) to 10 (very satisfying) rate your satisfaction with the course. Please write the rating in the space provided. **Extremely dissatisfying** **Very satisfying**
1 2 3 4 5 6 7 8 9 10
__ studied material __ working group __ technology
III. What is your familiarity with the software in this course and motivation for future applications?
_____ I am very familiar with the software used for this lab.
_____ I am not familiar with the software but have successfully used similar software.
_____ I am not familiar with the software.
IV. In a few words, Is there anything else you would like to communicate about the course?

TABLE 8. Classification for Stimulation and Satisfaction ranges.

Range	Classification
[8, 10]	Very [stimulating | satisfying]
[6, 7]	[Stimulating | Satisfying]
[5]	Neutral
[3, 4]	[Frustrating | Dissatisfying]
[0, 2]	Extremely [frustrating | dissatisfying]

TABLE 9. Words Ocurrence in Studied Survey.

Word	Occurrence
While	7
Python	6
Return	6
MachineLearning	5
SimpleCode	5
Else	4
For	4
AI	4
Programming	4
Processes	3
Threads	3
Thread	3
If	2
Language	2
Cloud	2
GoogleColab	2
Int	2
Print	2
Code	2
Linux	2
Software	2
Interpreter	2
GUI	2
Concurrency	2
Semaphores	2
Drive	1
Colab	1
Paradigm	1
Double	1
Multiparadigm	1
Try	1
Google	1
Free	1
Interface	1
Windows	1
Hardware	1
Core	1
Process	1
Kerner	1
User	1
Root	1
Architecture	1
MultiThread	1
Deadlock	1
ParallelProgramming	1

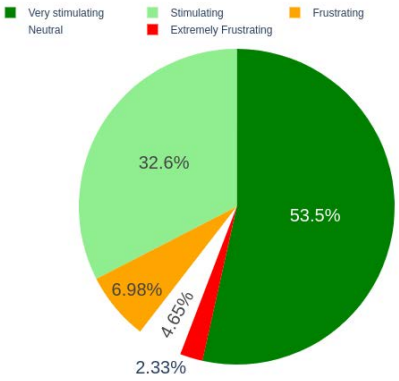
writing and execution of high-performance distributed and parallel computing.

Developing computing competencies in people using Python is a great goal. Several developed countries nowadays use and demand expert people with this technology. For example, Liu et al. [51] describe the necessity of training professionals in the field of artificial intelligence and report positive academic experiences in China of developing related competencies applying Python. Similarly, the work of Magnono et al. [52] present Python with machine learning applications in the United States of America. In contrast, Cart al. [53] describe the development of a machine learning tool using Python for financial forecasting and trading using a statistical strategy in Europe.

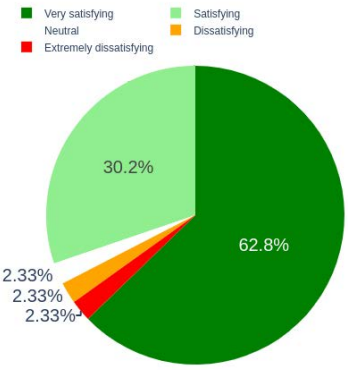
VI. THREATS TO VALIDITY

This work presents relevant results regarding using Python and Colab to develop high-education students' computing competencies effectively. Nonetheless, it is necessary to discuss the following practical issue:

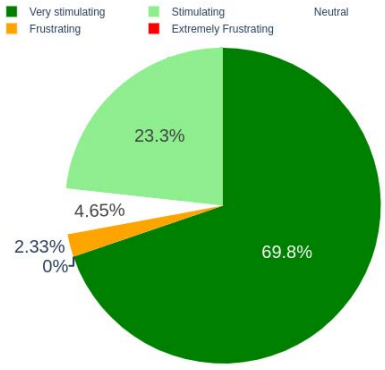
- Internet requirement: students need internet for using Colab and Python. If they did not want to use Colab, they needed the internet to install the software. It was an issue for students of CNU and VMU because there were cases with low-speed internet access. That was a real issue because classes were online during the pandemic, a problem that does not apply to traditional courses.



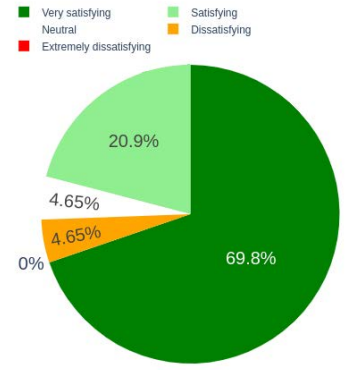
(a) Stimulation level about the studied material



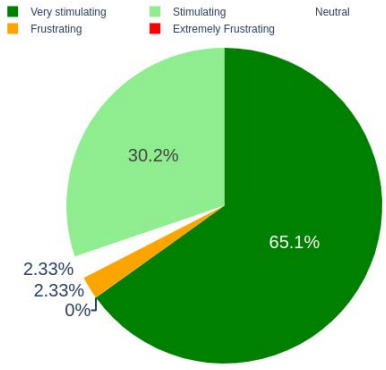
(b) Satisfaction level about the studied material



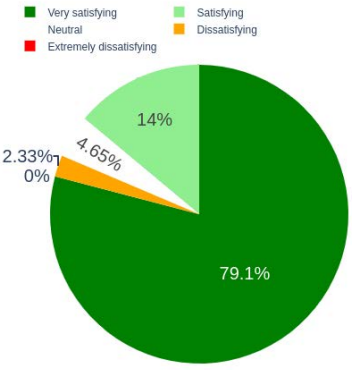
(c) Stimulation level about the working group



(d) Satisfaction level about the working group



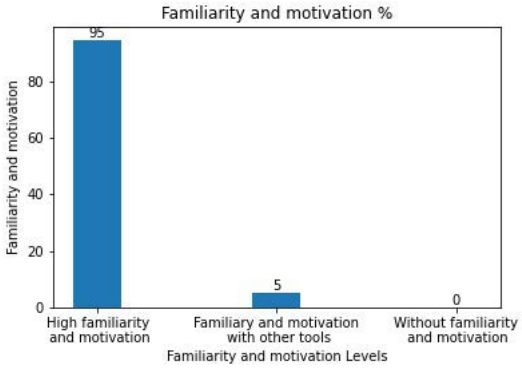
(e) Stimulation level about the applied technology



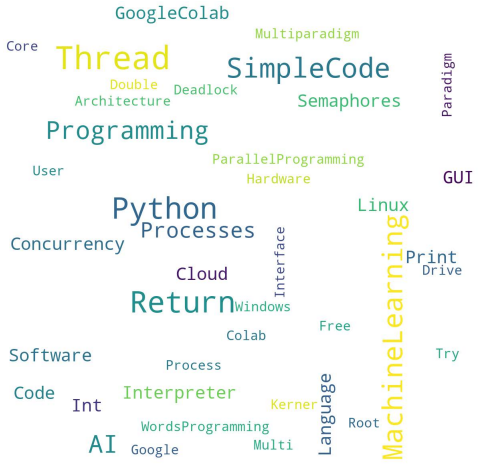
(f) Satisfaction level about the applied technology

FIGURE 4. Survey results regarding Frustration and Satisfaction levels in Programming and Operating System courses in the Development of Computing Competences.

- Programming rules and abstraction: Python is the best programming language for its syntax and simplicity. Van Rossum, the creator of Python, in 2001 [54], already stipulated a set of commitments for writing good-code solutions in Python. Nonetheless, one can think that learning Python limits students from not being familiar



(a) Survey results regarding the familiarity and motivation levels for continue using Python and Colab



(b) Survey results regarding the WordCloud

FIGURE 5. Survey results regarding the familiarity and motivation levels for continue using Python and Colab and WordCloud for Additional Comments.

with the more complex syntax of other and still popular programming languages like C, C++ and Java. That can be an issue for beginners in programming because, as we exemplify, high syntax rules difference exist between Python and other programming languages. We recommend starting with Python to develop algorithmic thinking competencies and learning different programming languages to apply that knowledge. Using Python was a pleasure for students who started with programming languages like CNU students and those who already had a programming background as Operating Systems at VMU students. Regarding abstraction and programming semantics, Python presents more benefits than issues: we can motivate to learn more computing approaches, paradigms and tools.

VII. CONCLUSION

According to the presented work and the results obtained, we conclude that:

- Python represents a highly suitable programming language for teaching initial programming courses thanks to its accessibility, simplicity, and current usability worldwide.
- Google Colab represents a highly suitable tool for using Python without restrictions in the information society. Colab offers a set of pre-installed libraries, and we can also install additional ones. In education scenarios, students can use Colab and interactively execute solutions without memory restrictions, apply advanced computing units like CPUs and TPUs, and share their working solutions to stimulate teamwork.
- Python and Colab is the perfect team to facilitate the development of computing competencies at different education levels. Students can evolve in the development of computing solutions from basic to advance. The currency of Python and Colab permits the high dissemination of solution examples with both tools.
- This study confirms the usability and positive results by applying Python and Google Colab in developing computing competencies for introductory or advanced computing courses. That represents a significant achievement for online education.

In summary, from an academic viewpoint, this article satisfactorily answered each proposed research question highlighting the usefulness of Python and Google Colab in the collaborative professor-students teaching-learning process for developing computing competencies. The experiments in the Programming at CNU and Operating Systems at VMU demonstrated the feasibility of applying Python and Colab for teamwork in online education. In future work, we expect to continue using those tools for developing computing competencies in other courses and topics such as databases, math computing, artificial intelligence, data structures and object-oriented programming.

ACKNOWLEDGMENT

The authors are currently living in a difficult time, plagued by a pandemic. Regardless of that, we continue working as a team and motivating the use of Python and Colab for developing computing competences.

REFERENCES

[1] F. Dağ, "Prepare pre-service teachers to teach computer programming skills at K-12 level: Experiences in a course," *J. Comput. Educ.*, vol. 6, no. 2, pp. 277–313, 2019.

[2] W. Susanti, "An overviuw of the teaching and learning process basic programming in algorithm and programming courses," *Turkish J. Comput. Math. Educ.*, vol. 12, no. 2, pp. 2934–2944, Apr. 2021.

[3] M. S. Naveed, "Comparison of C++ and Java in implementing introductory programming algorithms," *Quaid-E-Awam Univ. Res. J. Eng. Sci. Technol.*, vol. 19, no. 1, pp. 95–103, Jun. 2021.

[4] L. Moors, A. Luxton-Reilly, and P. Denny, "Transitioning from block-based to text-based programming languages," in *Proc. Int. Conf. Learn. Teaching Comput. Eng. (LaTICE)*, Apr. 2018, pp. 57–64.

[5] M. Li, Y. Guo, H. Zhao, K. Wang, and M. Lu, "Reform of the multi-platform blended teaching model of Python programming based on BOPPPS," in *Proc. 16th Int. Conf. Comput. Sci. Educ. (ICCSE)*, Aug. 2021, pp. 398–401.

[6] G. Futschek, "Algorithmic thinking: The key for understanding computer science," in *Proc. Int. Conf. Informat. Secondary Schools-Evol. Perspect.* Berlin, Germany: Springer, 2006, pp. 159–168.

[7] G. Mugweni and R. Mugweni, "Device programming for IoT: In defense of Python as the beginner's language of choice for IoT programming," in *ICT and Data Sciences*. Boca Raton, FL, USA: CRC Press, 2022, pp. 213–218.

[8] R. Rybarczyk and L. Acheson, "Integrating a career preparedness module into CS2 curricula through the teaching C++ and Java side-by-side," in *Proc. 49th ACM Tech. Symp. Comput. Sci. Educ.*, New York, NY, USA, Feb. 2018, pp. 592–597, doi: 10.1145/3159450.3159552.

[9] N. Marson, "Your parallel mind," in *Leading by Coaching*. Cham, Switzerland: Springer, 2019, pp. 3–28.

[10] M. J. Nelson and A. K. Hoover, "Notes on using Google colaboratory in AI education," in *Proc. ACM Conf. Innov. Technol. Comput. Sci. Educ.*, New York, NY, USA, Jun. 2020, pp. 533–534, doi: 10.1145/3341525.3393997.

[11] C. L. Vidal, C. Cabezas, J. H. Parra, and L. P. López, "Experiencias prácticas con el uso del lenguaje de programación scratch para desarrollar el pensamiento algorítmico de estudiantes en Chile," *Formación Universitaria*, vol. 8, no. 4, pp. 23–32, 2015.

[12] Y. Qian and J. Lehman, "Students' misconceptions and other difficulties in introductory programming: A literature review," *ACM Trans. Comput. Educ.*, vol. 18, no. 1, pp. 1–24, Dec. 2017.

[13] G. Génova and I. Q. Navarro, "Are human beings humean robots?" *J. Experim. Theor. Artif. Intell.*, vol. 30, no. 1, pp. 177–186, Jan. 2018.

[14] T. Matzner, "The human is dead—Long live the algorithm! Human-algorithmic ensembles and liberal subjectivity," *Theory, Culture Soc.*, vol. 36, no. 2, pp. 123–144, Mar. 2019.

[15] L. Mannila, L.-Å. Nordén, and A. Pears, "Digital competence, teacher self-efficacy and training needs," in *Proc. ACM Conf. Int. Comput. Educ. Res.*, New York, NY, USA, Aug. 2018, pp. 78–85, doi: 10.1145/3230977.3230993.

[16] C. Vidal-Silva, J. Serrano-Malebran, and F. Pereira, "Scratch and Arduino for effectively developing programming and computing-electronic competences in primary school children," in *Proc. 38th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Nov. 2019, pp. 1–7.

[17] J. Nouri, L. Zhang, L. Mannila, and E. Norén, "Development of computational thinking, digital competence and 21st century skills when learning programming in K-9," *Educ. Inquiry*, vol. 11, no. 1, pp. 1–17, Jan. 2020.

[18] M. L. Humphries, B. V. Williams, and T. May, "Early childhood teachers' perspectives on social-emotional competence and learning in urban classrooms," *J. Appl. School Psychol.*, vol. 34, no. 2, pp. 157–179, Apr. 2018.

[19] C. Cabo, "Student progress in learning computer programming: Insights from association analysis," in *Proc. IEEE Frontiers Educ. Conf. (FIE)*, Oct. 2019, pp. 1–8.

[20] A. Kumar and S. P. Panda, "A survey: How Python pitches in IT-world," in *Proc. Int. Conf. Mach. Learn., Big Data, Cloud Parallel Comput. (COMITCon)*, Feb. 2019, pp. 248–251.

[21] T. Srimadhaven, C. J. Av, N. Harshith, and M. Priyaadharshin, "Learning analytics: Virtual reality for programming course in higher education," *Proc. Comput. Sci.*, vol. 172, pp. 433–437, Jan. 2020.

[22] M. B. Garcia and T. F. Revano, "Assessing the role of Python programming gamified course on students' knowledge, skills performance, attitude, and self-efficacy," in *Proc. IEEE 13th Int. Conf. Humanoid, Nanotechnol., Inf. Technol., Commun. Control, Environ., Manage. (HNICEM)*, Nov. 2021, pp. 1–5.

[23] M. Tupac-Yupanqui, C. L. Vidal-Silva, A. Sánchez-Ortiz, and F. Pereira, "Experiencias y beneficios del uso de Arduino en un curso de programación de primer año," *Formación Universitaria*, vol. 14, no. 6, pp. 87–96, Dec. 2021.

[24] Y. Lee, "Python-based software education model for non-computer majors," *J. Korea Converg. Soc.*, vol. 9, no. 3, pp. 73–78, 2018.

[25] W. Vallejo, C. Díaz-Uribe, and C. Fajardo, "Google Colab and virtual simulations: Practical E-learning tools to support the teaching of thermodynamics and to introduce coding to students," *ACS Omega*, vol. 7, no. 8, pp. 7421–7429, Mar. 2022.

[26] M. Kuroki, "Using Python and Google Colab to teach undergraduate microeconomic theory," *Int. Rev. Econ. Educ.*, vol. 38, Nov. 2021, Art. no. 100225.

[27] L. Zuvanov et al., "The experience of teaching introductory programming skills to bioscientists in Brazil," *PLOS Comput. Biol.*, vol. 17, no. 11, Nov. 2021, Art. no. e1009534.

[28] K. Tock, "Google CoLaboratory as a platform for Python coding with students," *RTSRE Proc.*, vol. 2, no. 1, pp. 1–13, 2019.

[29] M. Lutz, *Learning Python: Powerful Object-Oriented Programming*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2013.

[30] S. A. Abdulkareem and A. J. Abboud, "Evaluating Python, C++, Javascript and Java programming languages based on software complexity calculator (Halstead metrics)," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 1076, no. 1, Feb. 2021, Art. no. 012046.

[31] B. Johnson and A. S. Chandran, "Comparison between Python, Java and R progrmming language in machine learning," *Int. Res. J. Modernization Eng. Technol. Sci.*, vol. 3, no. 6, pp. 1–6, 2021.

[32] A. Javed, M. Zaman, M. M. Uddin, and T. Nusrat, "An analysis on Python programming language demand and its recent trend in Bangladesh," in *Proc. 8th Int. Conf. Comput. Pattern Recognit.*, Oct. 2019, pp. 458–465.

[33] K. Srinath, "Python—The fastest growing programming language," *Int. Res. J. Eng. Technol.*, vol. 4, no. 12, pp. 354–357, 2017.

[34] M. Ali, S. Hussain, M. Ashraf, and M. K. Paracha, "Addressing software related issues on legacy systems—A review," *Int. J. Sci. Technol. Res.*, vol. 9, no. 3, pp. 3738–3742, 2020.

[35] A. K. Maji, L. Gorenstein, and G. Lentner, "Demystifying Python package installation with *conda-env-mod*," in *Proc. IEEE/ACM Int. Workshop HPC User Support Tools (HUST) Workshop Program. Perform. Visualizat. Tools (ProTools)*, Nov. 2020, pp. 27–37.

[36] A. Tanenbaum, *Modern Operating Systems*. London, U.K.: Pearson, 2009.

[37] A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Concepts Essentials*. Hoboken, NJ, USA: Wiley, 2013.

[38] A. S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*, 2nd ed. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2016.

[39] Z. A. Aziz, D. N. Abdulqader, A. B. Sallow, and H. K. Omer, "Python parallel processing and multiprocessing: A review," *Academic J. Nawroz Univ.*, vol. 10, no. 3, pp. 345–354, Aug. 2021.

[40] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: A survey," *ACM Comput. Surv.*, vol. 51, no. 6, pp. 1–35, Nov. 2019.

[41] T. Carneiro, R. V. M. D. Nóbrega, T. Nepomuceno, G.-B. Bian, V. H. C. De Albuquerque, and P. P. R. Filho, "Performance analysis of Google colaboratory as a tool for accelerating deep learning applications," *IEEE Access*, vol. 6, pp. 61677–61685, 2018.

[42] F. Perez and B. E. Granger, "IPython: A system for interactive scientific computing," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 21–29, May 2007.

[43] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. E. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. B. Hamrick, J. Grout, S. Corlay, and P. Ivanov, "Jupyter notebooks—A publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. Jupyter Development Team, IOS Press, 2016, pp. 87–90, doi: 10.3233/978-1-61499-649-1-87.

[44] B. M. Randles, I. V. Pasquetto, M. S. Golshan, and C. L. Borgman, "Using the Jupyter notebook as a tool for open science: An empirical study," in *Proc. ACM/IEEE Joint Conf. Digit. Libraries (JCDL)*, Jun. 2017, pp. 1–2.

[45] H. Fudaba, Y. Oda, K. Akabe, G. Neubig, H. Hata, S. Sakti, T. Toda, and S. Nakamura, "Pseudogen: A tool to automatically generate pseudo-code from source code," in *Proc. 30th IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Nov. 2015, pp. 824–829.

[46] Z. Wang, "Building experiments with Pygame," in *Eye-Tracking With Python and Pylink*. Cham, Switzerland: Springer, 2021, pp. 65–84.

[47] M. Tupac-Yupanqui, C. Vidal-Silva, L. Pavesi-Farriol, A. S. Ortiz, J. Cardenas-Cobo, and F. Pereira, "Exploiting Arduino features to develop programming competencies," *IEEE Access*, vol. 10, pp. 20602–20615, 2022.

[48] J. C. F. De Winter, "Using the student's T-test with extremely small sample sizes," *Practical Assessment Res. Eval.*, vol. 18, no. 12, p. 10, Jun. 2013.

[49] S. Kelley, C. Alger, and D. Deutschman, "'extreme programming' in a bioinformatics class," *Bioscene, J. College Biol. Teaching*, vol. 35, no. 1, pp. 58–65, 2009.

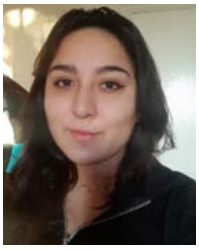
[50] V. Sharma. (2021). *Visualizations for Tkinter and PyGame in Colab*. Accessed: Jul. 15, 2022. [Online]. Available: https://vishnudsharma.medium.com/visualizing-tkinter-and-pygame-in-colab-272c5a245f8c

[51] Y. Liu and J. Huang, "Practice and exploration of artificial intelligence education in universities of political science and law with Python," in *Proc. 3rd Int. Seminar Educ., Manage. Social Sci. (ISEMSS)*. Amsterdam, The Netherlands: Atlantis Press, 2019, pp. 549–553.
[52] T. Mangono, P. Smittenaar, Y. Caplan, V. S. Huang, S. Sutermaster, H. Kemp, and S. K. Sgaier, "Information-seeking patterns during the COVID-19 pandemic across the United States: Longitudinal analysis of Google trends data," *J. Med. Internet Res.*, vol. 23, no. 5, May 2021, Art. no. e22933.
[53] S. Carta, S. Consoli, A. S. Podda, D. R. Recupero, and M. M. Stanciu, "An explainable artificial intelligence tool for statistical arbitrage," *Softw. Impacts*, vol. 14, Nov. 2022, Art. no. 100354.
[54] G. Van Rossum, B. Warsaw, and N. Coghlan, "PEP 8—Style guide for Python code," Python Enhancement Proposal PEP, Tech. Rep., Jul. 2001. [Online]. Available: https://peps.python.org/pep-0008/

CRISTIAN VIDAL-SILVA received the B.S. degree in information engineering from the Faculty of Engineering, Catholic University of Maule, Talca, Chile, in 2003, the M.S. degree in computer science from the University of Concepcion, Concepción, Chile, in 2007, the M.S. degree (Fulbright Scholar) in computer science from Michigan State University, East Lansing, MI, USA, and the Ph.D. degree in software engineering from the University of Seville, Seville, Spain. He is currently a full-time Professor with the Faculty of Engineering, Videogame Development and Virtual Reality Engineering (VDVRE) School, University of Talca, Chile. He is the principal author of more than 30 articles in research areas, such as feature-oriented and aspect-oriented software engineering, machine learning, digital circuits, and programming. He also worked as a reviewer of WoS and Scopus journals.

NICOLÁS A. BARRIGA (Member, IEEE) received the Ph.D. degree from the University of Alberta, Canada, for his work on state and action abstraction mechanisms for RTS games. He is currently a Professor with the School of Videogames Development and Virtual Reality Engineering, Universidad de Talca, Chile. His current research interests include procedural content generation for videogames and the broad area of video game AI.

FRANCO ORTEGA-CORDERO is currently pursuing the degree in civil computing engineering with Viña del Mar University, Viña del Mar, Chile. During his time as a student, he participated in different projects, such as a virtual store for entrepreneurs in the city of La Calera digital library for the Commercial Secondary School "Carol Zamora Vicencio," Quilpué. He was one of the best students in the operating systems course at VMU, in 2021.



JAVIERA GONZÁLEZ-LÓPEZ is currently pursuing the degree in civil computing engineering with the University of Viña del Mar, Viña del Mar, Chile. She was among the best in the operating systems course at VMU, in 2021.

CLAUDIA JIMÉNEZ-QUINTANA received the degree in civil computer engineering and the master's degree in computer science from Concepción University, Chile. She is currently pursuing the Ph.D. degree in engineering sciences with the Catholic University of Chile. She is also a full-time Professor and the Director of civil computing engineering at Viña del Mar University, Viña del Mar, Chile.

CLAUDIA PEZOA-FUENTES was born in Chile. She received the degree in administration from the University of Antofagasta, Chile, in 2003, and the master's degree (European) in market and enterprise and the Ph.D. degree in business administration and management from Universitat Rovira I Virgili, Spain, in 2008 and 2010, respectively. From 2006 to 2010, she was a Research Assistant with the Department of Business Management, Universitat Rovira I Virgili. Since 2015, she has been an Assistant Professor at the Department of Administration, Faculty of Administration and Economics, Católica del Norte University, Chile. Also, she is the author of several articles, one of them published in the international journal *Resources Policy*. Her research interests include management, engagement, innovation, territorial sector agglomerations, and proximity.

IVÁN VEAS-GONZÁLEZ received the Bachelor of Science degree in business administration, the master's degree in administration, and the master's degree in digital marketing and electronic commerce. He is currently pursuing the Ph.D. degree in marketing with the University of Valencia, Spain. He is currently an Assistant Professor and a Researcher in marketing and management with the Department of Administration, Faculty of Administration and Economics, Universidad Católica del Norte, Chile. His research interests include management, innovación, service, value, digital marketing, and electronic commerce.