Journal of
Information
Systems
Education

Volume 32
Issue 4
Fall 2021

First Course Programming Languages within US Business College MIS Curricula

Tim C. Smith and Leslie Jones

Recommended Citation: Smith, T. C., & Jones, L. (2021). First Course Programming Languages within US Business College MIS Curricula. *Journal of Information Systems Education,* 32(4), 283-293.

Article Link: https://jise.org/Volume32/n4/JISE2021v32n4pp283-293.html

Initial Submission:	24 October 2020
Accepted:	17 March 2021
Published:	15 December 2021

ISSN: 2574-3872 (Online) 1055-3096 (Print)

First Course Programming Languages within US Business College MIS Curricula

Tim C. Smith
Department of Information Systems and Business Analytics
Iowa State University
Ames, IA 50011, USA
timsmith@iastate.edu

Leslie Jones
College of Social Sciences, Mathematics and Education
University of Tampa
Tampa, FL 33606, USA
lbjones@ut.edu

ABSTRACT

There are many factors to consider when selecting a first programming language for the management information systems (MIS) program. Determining the appropriate language for introductory MIS programming courses is challenging due to the lack of research guidance that addresses the specific context of programming in MIS curricula. This paper reports a summary of results from a survey that explores the languages used within introductory programming courses in US business colleges. Findings indicate that Python has emerged as the most popular language used in MIS curricula and that many of the languages currently in use have only recently been adopted. Moreover, there is both a transition from and transition to the Java language, possibly indicating a leading and trailing language transition wave. Emergent themes suggest that integration with other courses, industry demand, and faculty training and support are essential factors in the selection process.

Keywords: Curriculum design & development, Programming languages, Application development, Job skills, Introductory programming, Curriculum alignment

1. INTRODUCTION

While other business disciplines have arguably achieved relative stability and strong identity, the management information systems (MIS) discipline has long navigated the ambiguous, abstract, and continually changing nature of its situated practice (Agarwal & Lucas, 2005; Benbasat & Zmud, 2003; Orlikowski, 2001). This dynamic property challenges MIS curricula developers to increase agility and respond to current industry trends while simultaneously maintaining the fundamental concepts that bring continuity across time. One area that has proven particularly challenging in this regard is that of application development and computer programming.

Though programming is not included in 2010 IS curriculum guidance (Topi et al., 2010), Baugh (2016) found that 99% of MIS programs have at least one required introductory programming course in their curriculum. Bohler et al. (2020) put this percentage at 81% and found that 34.8% of the universities in their study require multiple programming courses in their IS curriculum. They note this as a sign of high value placed on student programming skills by IS educators (Bohler et al., 2020). Moreover, the industry demand for technical skills that include programming from graduates is increasing (Apigian & Gambill, 2009). Cummings and Janicki (2020) conclude from their 2020 study of employer demand that IS curricula should continue to include programming courses, with Software Developer at the top of their future job growth category list and Business/System Analyst in second place. Despite this prevalence of programming courses within MIS programs, much of the scarce literature on programming languages selection and teaching is from the perspective of computer science, software engineering, and related technical fields. Little has been published in areas specifically targeting the languages used in MIS programming courses and the challenges MIS curricula developers face when selecting and transitioning to new languages.

With a lack of guidance provided by the MIS literature, there is a propensity to use broad industry data on language popularity to help guide computer programming curriculum development. MIS programming curricula developers face a unique challenge: changing the curriculum to match industry demand while maintaining consistent program delivery. This challenge is further compounded by the need to differentiate from the more technically oriented computer science and engineering degree programs and identify the specific skills demanded from MIS graduates versus general industry demand.

This paucity of direction from the literature results in many curricula developers assembling course content using stripped-down introductory computer science courses that may not sufficiently support business concepts nor integrate well with following core courses such as systems analysis & design, analytics, and data management. MIS curricula developers' key challenges will be to continue to separate programming course development from computer science and engineering directions and further understand MIS graduates' unique roles within the industry. These efforts must include identifying the unique properties of MIS programming courses that support the development of competencies that best staddle the technical and the business aspects of the jobs MIS students will later perform.

This study provides guidance to help those responsible for language selection in MIS programs' introductory programming curriculum. Our research objectives are to identify the existing languages in use, any previous language used, and what, if any, future languages MIS faculty are considering. Further, we investigate satisfaction indicators with the current language and how long the current language has been in use. The latter set of questions thus provides insights on issues of the stability of the current language choice. We explore factors involved in both the willingness and perceived difficulties in migrating to a new language, which could indicate effects that impede curriculum advancement. Though these questions may not solve the grander challenge of identifying the unique programming competencies that MIS students require, this work does provide evidence of current languages in use, issues surrounding the selection and transition to new languages, and the stability of current language choices.

Our contribution is unique for its focus on the US business college curriculum and MIS business faculty perspectives. Our review of the existing literature reveals that very few studies distinguish between the competencies generated from an MIS course sequence and those in computer science and software engineering. To contribute to a more thorough discussion of this area, we address central questions of programming language use in MIS through a survey of a diverse sample of US business college faculty stakeholders in introductory programming language choice.

The paper proceeds as follows. Firstly, we evaluate the existing literature. Secondly, we introduce our survey, target sample, and method. Thirdly, we review results. Lastly, we discuss the findings, conclusions, and opportunities for future research.

2. PRIOR RESEARCH

We used a snowballing approach to investigate prior introductory programming research (Badampudi et al., 2015). We began this process by identifying an initial set of seed papers from *Web of Science* using combinations of search strings that included "Introductory Programming Language" and "Information Systems." We then conducted both a backward and forward search of relevant literature, investigating the references to and within each identified paper from this initial list. This literature review's objective was to identify sources along three key dimensions: 1) industry demand, 2) languages in use in introductory programming courses, and 3) challenges of language selection.

2.1 Industry Demand

Though there are examples in the literature that identify industry demand for specific programming-related competencies, the delineation between the competencies required from MIS students from other fields such as computer science and software engineering are not well defined nor understood. The small number of examples that identify MIS specific demand include a 2002 study (Koong et al., 2002) of 300 IS job postings that indicated 163 of these jobs fit into a programming language skill category. Also, a 2008 study (McMurtrey et al., 2008) of 159 IS professionals indicated 24% of their roles included development programming, while 14.6% indicated maintenance programming (though the total percentage of respondents' roles that involve programming could be higher than 24%, the overlap between the two measures was not clear and therefore this total figure could not be determined).

Though the above examples focus specifically on industry demand for MIS graduates, they do not sufficiently detail the specific languages in use within these areas of demand. For guidance in this area, we found several sources that rank the popularity of programming languages in use within the industry. The most influential rankings include The Institute of Electrical and Electronics Engineers (IEEE) journal *IEEE Spectrum* (Interactive: The Top Programming Languages, 2018), RedMonk (The RedMonk Programming Language Rankings, 2021), the PYPL Index (PYPL PopularitY of Programming Language Index, 2021), and TIOBE (TIOBE Index, 2021). A summary of results from these rankings is provided in Table 1. An aggregate ranking of languages common across all lists is found in Table 2.

Rank	RedMonk	TIOBE Index	PYPL	IEEE Spectrum
1	JavaScript	C	Python	Python
2	Python	Java	Java	Java
3	Java	Python	JavaScript	C
4	PHP	C++	C#	C++
5	C#	C#	C/C++	C#
6	C++	Visual Basic	PHP	JavaScript
7	CSS	JavaScript	R	Assembly
8	TypeScript	PHP	Obj-C	PHP
9	Ruby	SQL	Swift	HTML
10	C	Assembly	Matlab	Scala
11	Swift	R	Kotlin	Shell
12	R	Goovy	TypeScript	Ruby
13	Obj-C	Go	Go	Matlab
14	Shell	Ruby	VBA	R
15	Scala	Swift	Ruby	Perl
16	Go	MATLAB	Rust	Go
17	Shell	Delphi/ Pascal	Ada	SQL
18	Kotlin	Visual Basic	Visual Basic	Obj-C
19	Rust	Perl	Scala	Swift
20	Perl	Obj-C	Lua	Arduino

Table 1. Language Popularity Indices

Though many of these studies can be used in the MIS language selection process, they do not necessarily represent the specific language skills expected of MIS graduates. Moreover, for all but the top languages, there is a significant disparity in language popularity. This difference is indicative of the challenge of aggregating languages in use across differing regions, industries, and fields.

	RedMonk	TIOBE Index	PYPL	IEEE Spectrum	Score (lower is better)
Python	2	3	1	1	7
Java	3	2	2	2	9
C/C++**	6	4	5	3	18
JavaScript	1	8	3	7	19
C#	5	5	4	5	19
PHP	4	8	6	9	27
R	12	11	7	14	44
Ruby	9	14	15	12	50
Swift	11	15	9	19	54
Go	16	13	13	16	58
Objective-C	13	20	8	18	59

** Total score for C/C++ is calculated by selecting the higher ranking of the two from each list.

Table 2. Aggregate Scoring of Common Languages

2.2 Languages Used in Introductory Programming Courses

Several studies investigate current languages in use within introductory programming courses. Mason and Simon (2017) found that Python and Java were the most popular in Australasian universities. A more recent study of Greek universities indicated C as the most popular language, at more than twice the rate of the second most popular language Matlab (Avouris, 2018). Another 2018 study in Ireland (Becker, 2019) found multiple language usage, with Python and Java at 90% and 78%, respectively, and the number one reason for the choice of languages being industry demand. In the United Kingdom, Murphy et al. (2017) published survey results from a 2017 study that identified Java as the primary programming language in use in introductory programming courses. Another European study from 2016 found the C programming language dominated first-semester programming languages used by universities (Aleksić & Ivanović, 2016). In the United States, Richard Reid of Michigan State University started The Reid List, which documented introductory programming course languages in computer science and information systems majors at 400 universities across the country, beginning in the early 1990s. The most recent review was published in 2016 by Siegfried et al. (2016) and showed Java at the top of the list, with Python and C++ within a percentage point of each other, but at less than half the popularity of Java.

These studies indicate relatively high use of C/C++ and Java, but this result might be influenced by a large sampling of computer-science-oriented curricula. The C programming language has a long history in computer science and is the foundation of many system-level application development projects. C's lack of automated garbage collection requires students to fully understand the inner workings of system memory allocation and deallocation, and in general, get "closer" to the computer hardware. For computer science and engineering majors, the need to understand such concepts may be essential. Still, the rise in Java use within these disciplines indicates that early exposure to such low-level constructs may not be necessary. Java's garbage collection and portability would seem to outweigh the need to provide the detailed hardware-level exposure that C/C++ provides. For purposes of management information systems, we argue that much of this knowledge is secondary to developing competencies useful in business-related tasks such as data analysis, process automation, simulation, and user interface design. Though we could elaborate further contrasts between MIS and computer science, we establish that the motivations for language selection within MIS should differ from those found in computer science and engineering fields. We find little indication that the literature addresses such delineation.

2.3 Challenges of Language Selection

Much discussion has also occurred surrounding the challenges in teaching introductory programming (Cheah, 2020; Koulouri et al., 2014; Mehmood et al., 2020; Prasad & Li, 2004; Zhang et al., 2013). Within this literature, we find criticism of language selection processes that give precedence to industry adoption over ease of learning (Parker et al., 2006). Interestingly, Ben Arfa Rabai et al. (2015) found that a given language's intrinsic properties play only a minor role in language choice and contextual factors have a much more significant influence on language selection. We find evidence of the effect within academia, where languages developed with the express purpose of supporting education (e.g., Scratch) are not well adopted. Within industry, a related effect can be seen when the accumulation of large code repositories impedes the transition to new, potentially better languages.

2.4 Summary of Literature

From these identified studies across dimensions of industry demand, languages in use within introductory programming courses, and challenges of language selection, we do not find results specifically associated with US business colleges, nor with MIS competencies demanded of MIS graduates. At issue is the need for a more rigorous investigation into MIS-specific language selection issues.

Our literature review results confirm our concern that there is insufficient guidance to assist MIS programming curriculum development. Moreover, though traditional computer science and engineering disciplines include many studies on language selection and education, our concern is that the findings from such studies may be driven by contextual factors that do not align with MIS educational goals (Clear et al., 2019). With the growing demand by industry for MIS students to obtain programming skills, we believe that the selection of language used in MIS programming courses is an important but under-investigated research area.

3. METHODOLOGY

To address our primary research questions, we surveyed US business college MIS programs.

3.1 Population and Sample

We defined our initial population as public and private US business colleges that offer a four-year degree in MIS. The selection of this population is motivated by the need to inform decisions on curricula that involve introductory computer programming in the MIS curriculum at our respective universities.

To identify our population sample, we used a list of US business colleges extracted from a publicly maintained list (List of Business Schools in the United States, n.d.). This list consisted of 427 business colleges. With the support of two graduate assistants, we reviewed the website of each college to identify our sample using the following criteria:

- Institution is a 4-year degree-granting public or private university.
- Institution offers either an MIS or business analytics major.
- An appropriate contact can be identified (either a department chair or a recognized instructor of introductory programming).

These efforts yielded a sample of 135 colleges.

3.2 Survey Instrument

The MIS literature revealed no previously established set of questions used to measure the adoption and use of languages within MIS curricula. In response, we developed a survey instrument based on the research questions we sought to pursue. The resulting survey consisted of direct questions about languages used in introductory programming, other questions related to identifying previous languages used, overall satisfaction with current language used, languages under consideration, and the impediments to moving forward with any new language.

We selected 15 contacts randomly from our initial list of 135. We solicited feedback on the survey instrument, which we incorporated into the final version.

We conducted our survey from fall semester 2018 through 2019. Invitations were initially sent to either the chair of each department or directly to an identified instructor of introductory programming. Each recipient was asked to forward the invitation if they were not familiar with the programming course they offer. Forty-seven of the recipients actively participated in the survey (35%).

3.3 Sample Characteristics

The responses show a skew towards medium to larger sized universities (Figure 1), with no apparent bias regarding location (Figure 2).

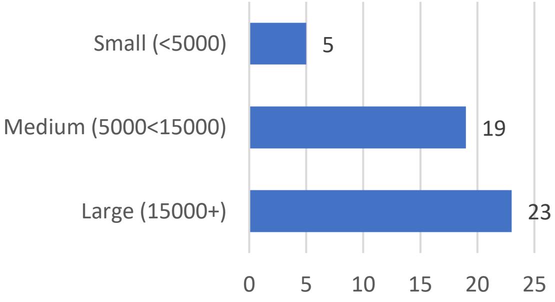

Figure 1. Distribution of Responses by University Size (undergraduate enrollment)

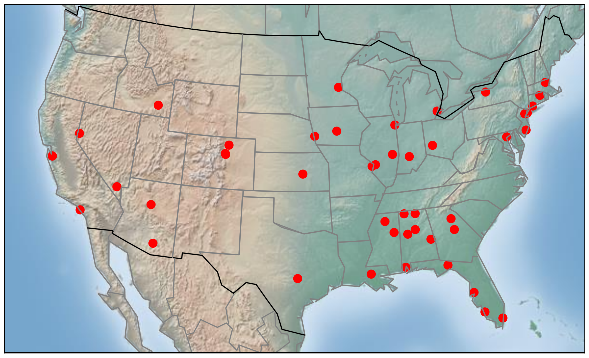

Figure 2. Geographic Distribution of Responses Received

Of the responses received, nine (19%) identified as, at least partially, deciding on the languages used, 23 (49%) identified as being both faculty and administration, and six (13%) identified as currently teaching introductory programming. Lastly, 44 responses (94%) indicated that introductory programming was required for the MIS degree.

4. RESULTS

4.1 Language Selection

To understand the current MIS programming curriculum, we measured several factors related to language selection. These factors included asking what language is used, the previous language used, and how long the current language has been in use.

Our findings indicated that the most popular languages in use (Figure 3) are Python (n=17), followed by Java (n=10), Visual Basic (n=6), and C# (n=5). Additional languages on this list include JavaScript (n=3), PHP (n=2), C++ (n=1), and Swift (n=1).

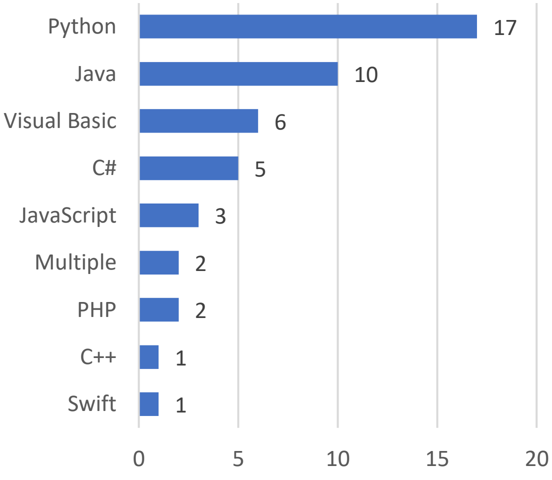

Figure 3. Primary Language Used (count)

Measurement of how long the current language has been in use (Table 3) indicates that departments have only recently adopted the top languages, with Python dominating recent transitions.

Primary Language	Years in Use	Count
C#	0-1 years	1
	1-2 years	1
	3-4 years	1
	5-6 years	1
	5-9 years	1
C++	3-4 years	1
Java	0-1 years	5
	3-4 years	2
	5-6 years	1
	5-9 years	1
	No Response	1
JavaScript	5-6 years	2
	No Response	1
Multiple	0-1 years	1
	1-2 years	1
PHP	1-2 years	1
	5-6 years	1
Python	1-2 years	14
	3-4 years	1
	5-6 years	1
	5-9 years	1
Swift	1-2 years	1
Visual Basic	0-1 years	4
	1-2 years	1
	5-6 years	1

Table 3. Recency of Adoption

The measurement of previous languages used by respondents shows that Visual Basic and Java were the most referenced (Figure 4).

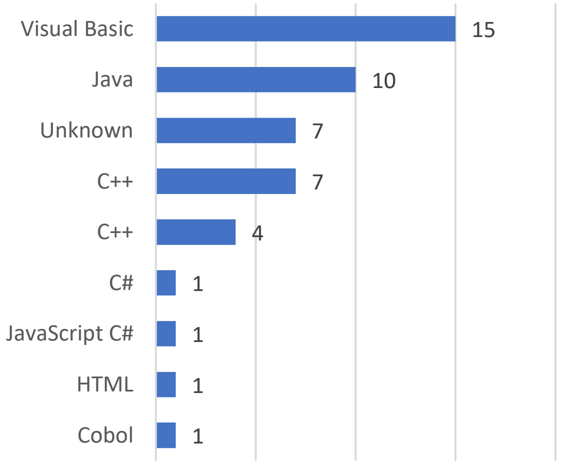

Figure 4. Prior Languages Used

A comparison of the survey responses to previous and current languages (Table 4) indicated that the most frequent language transitions were Visual Basic and Java to Python.

From		To	n
Visual Basic	→	Python	6
C++	→	Java	5
Java	→	Python	4
Visual Basic	→	C#	4
C++	→	Python	4
Visual Basic	→	Java	3
Java	→	Visual Basic	3
C++	→	Java Script	2
Visual Basic	→	PHP	2
JavaScript	→	Python	1
Java	→	C++	1
Java	→	C#	1
C#	→	Python	1
Cobol	→	Visual Basic	1
C++	→	Visual Basic	1
HTML	→	Python	1
Visual Basic	→	JavaScript	1

Table 4. Language Transitions

4.2 Stability of Current Selection

To understand the stability of choice in the current language used, we asked respondents if there were plans to change the existing language. Of those that responded, five (10.6%) indicated that plans to change were in place (Figure 5).

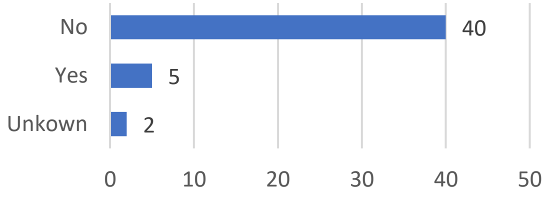

Figure 5. Plans to Change Current Language

Of the five respondents that indicated plans to transition from the current language, three were transitioning from Java, and two were transitioning from C/C++ (Table 5).

Language Currently Used	n
Java	3
C++	1
C#	1

Table 5. Languages Under Transition From

4.3 Satisfaction with Current Language

To measure satisfaction with the current language, we asked respondents the questions outlined in Table 6.

A summary of responses (Figure 6) indicates agreement across many of these questions, with the least satisfaction observed in questions LS1, LS2, and LS4.

Due to the rather small sample size, statistically significant correlations between satisfaction factors and other indicators, such as language, duration of use, etc., are not identified.

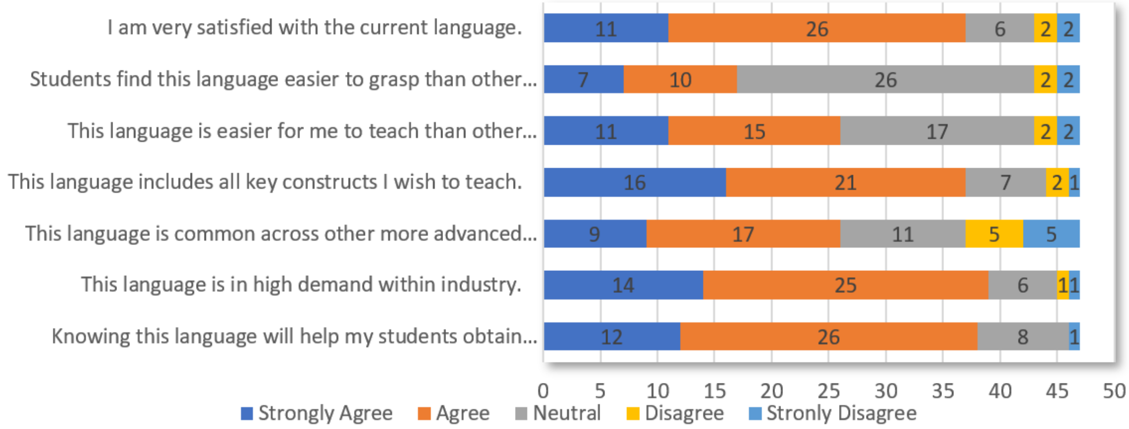

Figure 6. Satisfaction Indicator Results

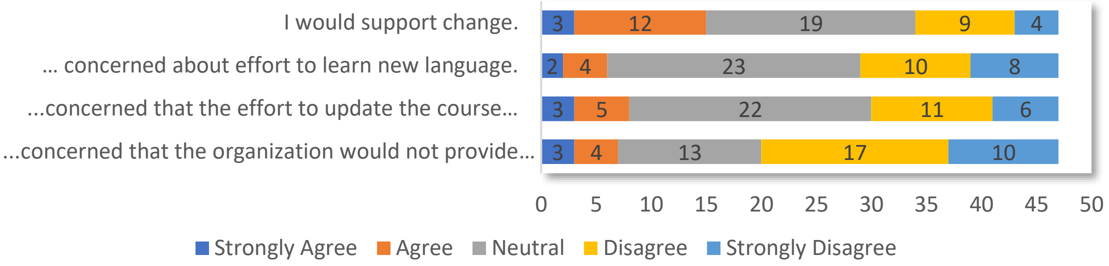

Figure 7. Transition Impediment Indicator Results

Full Question Text	Code
I am very satisfied with the current language.	LS0
Students find this language easier to grasp than other languages.	LS1
This language is easier for me to teach than other languages.	LS2
This language includes all key constructs I wish to teach.	LS3
This language is common across other more advanced courses at our institution.	LS4
This language is in high demand within industry.	LS5
Knowing this language will help my students obtain employment.	LS6

Table 6. Language Satisfaction Indicators

4.4 Inhibiting Factors to Changing Language

To understand potential inhibiting factors for language change, we asked respondents questions relating to concerns with switching languages (Table 7).

Full Question Text	Code
I would support change.	CL0
I would be concerned about the time it would take me to learn a new programming language.	CL1
I would be concerned that the effort to update the course material would be excessive.	CL2
I would be concerned that the organization would not provide sufficient support for such a change to be a success.	CL3

Table 7. Change Support

A summary of responses to these questions (Figure 7) indicates a slight concern with creating new material and a lack of sufficient organizational support.

4.5 Popularity/Health of Introductory Programming

Forty-four of the respondents (94%) indicated that the introduction to programming course was a required course. The remaining four (6%) respondents indicated that the course was an elective within the MIS program.

Our survey also investigated the relative enrollment in MIS programming courses. In general, programming course enrollment was similar to other college classes (Figure 8), and no indication of decreasing demand for such courses was observed (Figure 9).

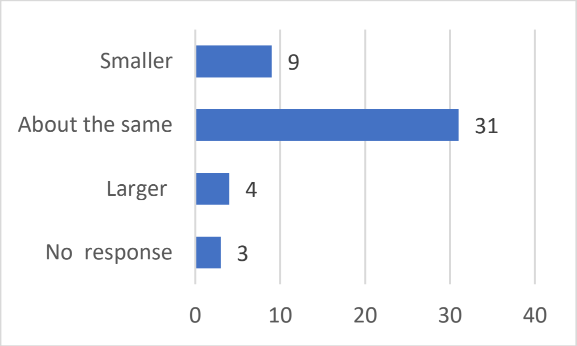

Figure 8. Relative Class Size

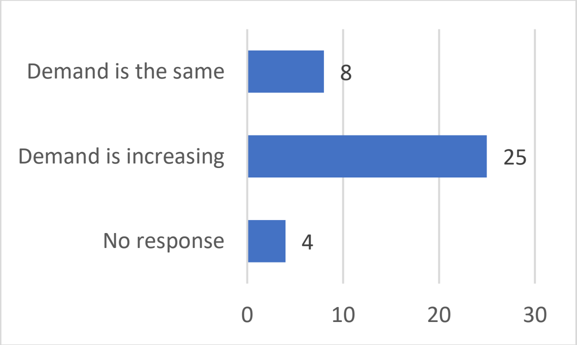

Figure 9. Student Demand for Programming

4.6 Open Ended Responses

The survey included four questions where respondents could provide open text responses (Table 8).

To identify themes in open-ended responses, we used an inductive coding technique (Saldana, 2015). From this coding effort, ten themes emerged (Table 9).

We provide a detailed summary of responses and associated coding in Tables 10 through 14.

Summary of Open Responses	n
Offered to all respondents	
• What other benefits do you see in using this current language?	11
• What other concerns would you have about such a change?	9
Offered to those indicating plans to change in place	
• Please provide your thoughts as to why the current language is being abandoned.	5
• Please add your thoughts on what key benefits will be achieved by switching to this new language?	3
• Please provide your thoughts on the key challenges your department will face in transitioning to this new language.	3

Table 8. Open Response Questions used in Survey

Theme	n
Integration with other courses	10
Industry Demand	8
Training & Support	6
Platform Support	4
Features Constructs	3
Available Resource	3
No Benefit	2
Ease of Teaching	2
Faculty Driven	1
Language Does not Matter	1

Table 9. Open Response Themes

Q1: "What other benefits do you see in using this current language?"	Ease of Teaching	Features/Constructs	Available Resources	Integration with Other Courses	Industry Demand	Platform Support	Language Doesn't Matter
"simpler than Python with respect to object-oriented features and easier to understand o-o references for beginners"	X	X					
"Excellent textbook (David Shneider) that provides many opportunities to practice"			X				
"VBA is used extensively in combination of Excel, which can be directly applied in the industry setting."				X	X		
"This is one of the toughest decisions we have had to make [switch to JavaScript]: relevance of coding in IS curriculum. We have gone full circle: from three required programming classes, to one, now back to two and probably another. Our advisory board is recommending more coding."					X		
"As it is .NET framework Visual Basic allows the student to go out and work in C#.NET as well as have an understanding of web-based, event driven coding using windows-based controls."						X	
"This survey is missing the point: We INTENTIONALLY do NOT teach a single language. That does the aspiring programmer a severe disservice. We teach PROGRAMMING FUNDAMENTALS and intentionally use multiple languages to reinforce the fundamental concepts regardless of the language syntax. Languages come and go so frequently that the learning that is needed to be achieved is NOT language-centric. By leading all of these survey questions in the direction of coming up with 'specific languages' you are creating a self-fulfilling prophecy that the language matters when, in fact, it does not"		X					X
"C# comes with a well integrated IDE (Visual Studio), version control integration (GitHub) and cloud deployment setup (Azure)"			X			X	
"We like having the option of extending the use of this language into data analytics courses."		X		X			
"Working with the Apple/AACSB initiative -- helps students see the benefits of moving to a 'mobile first' strategy"						X	
"heavily used in data analytics"				X			
"Open source so they learn an open-source platform. Taught VB.net before which was easier but the learning curve for the IDE was much higher"						X	
Total	1	3	2	3	2	4	1

Table 10. Benefits of Current Language

Q2a: "What other concerns would you have about such a change?"	Training & Support	Ease of Teaching	Industry Demand	Integration with Other Courses	Faculty Driven	No Benefit
"Cost for training faculty is a concern for us. Ours is an established 30-year-old program where incentives to relearn are minimal."	X					
"The new language might not be a good teaching language, and so the students would struggle with it."		X				
"We are adding a second course (like programming part 2) for Python - keeping JavaScript as the intro course to teach programming concepts. This way the Python course can get more advanced quickly."				X		
"It would have to be driven by market forces in the industry. We are not creating coders per se, but they need to be knowledgeable in current coding/programming environments at the least and many choose to become more technical rather than less as they gain experience in the IS/IT world."			X			
"I do not see a benefit to changing. If there were, we would have changed already"						X

"It is VERY difficult to get the school to adopt/change material."	X					
"has some effect on downstream courses"				X		
"It really does not matter what language they learn as long as they learn all the constructs rather than a drag and drop scenario."						X
"The 'institution' would not change the language that our faculty chose for our major."					X	
Total	2	1	1	2	1	2

Table 11. Concerns about Changing Language

Q2b: "Please provide your thoughts as to why the current language is being abandoned."	Training & Support	Ease of Teaching	Industry Demand	Integration with Other Courses
"VBA and C# are both useful, but Python might be used most among all languages."			X	
"Python is easier to learn, is becoming increasingly important in organizations, and is the language in MS Cyber program."	X		X	X
"Articulation with subsequent courses."				X
"We are adding both cybersecurity and data analytics as new majors (in addition to CIS). Python is more applicable to these new majors. We haven't made the decision yet whether we will add a new course on Python for those two majors, or change the existing class and make all 3 majors take the Python course (doing away with Java in the Introductory class)."				X
"Other languages are in higher demand in the marketplace; our students would be better served by learning a different language."			X	
Total	1	0	3	3

Table 12. Why Current Language Is Being Abandoned

Q2c: "Please add your thoughts on what key benefits will be achieved by switching to this new language?"	Industry Demand	Integration with Other Courses
"As mentioned above, Python is a general-purpose language and can see applications outside of the business discipline."	X	X
"Primarily because python is more applicable for cybersecurity and data analytics majors."		X
"Students would have more marketable skills; student placement rates would be enhanced."	X	
Total	2	2

Table 13. Benefits of Moving to New Language

Q2d: "Please provide your thoughts on the key challenges your department will face in transitioning to this new language."	Training & Support	Available Resources
"New course preparation, identification of suitable textbook for business students."	X	X
"Faculty need to find time to learn python and then change the course. We are making drastic changes to several classes with the addition of two new majors beginning this fall. So, learning python and making the changes isn't really difficult, it's just difficult to find the time to do this."	X	
"The instructors will have to change the curriculum and possibly have to familiarize themselves with the new language."	X	
Total	3	1

Table 14. Key Challenges in Transitioning to New Language

5. DISCUSSION

This paper examines the programming languages in use within US MIS degrees. This paper's primary objectives were to identify the most common languages in use, the most recent previous languages used before the current language, and what, if any, future languages are being considered. The secondary objectives were to explore related factors such as satisfaction with the current language, concerns, and motivations for transitioning to new languages.

The results indicate that Python is now the dominant language in use, followed by Java and Visual Basic. Our analysis suggests a recent transition to Python has occurred, with many observations indicating recent shifts from both Java and VB to Python.

Though the small sample does not warrant statistically significant results, there is an indication that the language's overall satisfaction is independent of the language in use. Moreover, five of the 47 responses received (10.6%) indicated active plans to change from the current language. This information, coupled with the recency of use of many of the existing languages, provides evidence that resistance factors to changing languages do not seem to be resulting in any significant lock-in effects.

The analysis of open-ended response questions revealed ten themes, with the most prevalent of these related to contextual factors such as integration with other courses (n=11), industry demand (n=9), and support and resources for faculty to transition to any new language (n=6). Of the ten themes identified, only two – ease of teaching (n=1) and availability of constructs (n=3) – are directly related to the language's properties/characteristics. It is interesting to note that each of the themes is essentially unrelated to the language's features or properties and is instead focused on contextual factors. This observation matches that of Ben Arfa Rabai et al. (2015). They concluded that contextual factors play a more prominent role in the decision to adopt a language than language features or properties.

One open-ended response indicated that offering multiple languages was the most beneficial approach for students. A further investigation of quantitative results revealed that two (4.3%) responses indicated multiple languages being taught simultaneously in the introductory course. This result suggests a possible divide in selecting breadth over depth of coverage of programming constructs.

These results indicate that though some MIS programs may lag in transitioning to more popular languages, 30 respondents (63.8%) showed current language in use for one to two years. These are encouraging results that indicate MIS educators are responding to current trends in industry. This is also evidence of curriculum alignment to the most popular languages indicated in the current language popularity rankings (Table 2). However, it remains unclear if program-specific needs have been considered in these decisions or if there has been sufficient alignment of these choices within the overall MIS department curriculum.

6. CONCLUSIONS

Due to the lack of research into this area of MIS, these findings provide important insights for instructors and those involved in setting direction in curricula designed in MIS. Though business colleges would seem to be responding well to industry language preference changes, the question remains if academia should be leading or following industry – and with languages such as Cobol still being taught, a related question of what niches should be pursued. The common language popularity indices (Table 1) present an average across many different sub-areas. The popularity of language used in industry can vary greatly depending on the specific sub-field area. As competition for MIS students increases, the choice of language used could become a point of differentiation.

Moreover, this study's results indicate that administrators need to address factors that may inhibit curricula advancement. These include faculty support for change, training availability, and adequate support for periodic reviews of programming curricula. Another vital consideration is the use of the language in subsequent courses in the curriculum. Changes in introductory language must carry the burden of changes to follow-up courses. Such decisions must be driven by a much broader discussion of impacted courses.

It is important to consider the relatively small sample of this study when generalizing across the entire population. Moreover, because no previous instruments were identified, this study piloted a new survey instrument, and it is recommended that feedback from this experience be incorporated into a new survey instrument. We suggest that future studies refine the survey instrument and expand the studied population to include non-US colleges.

The results of this study indicate several interesting areas for continued exploration of programming language curriculum in MIS. Outstanding questions include the identification of specific MIS related programming competencies, industry demand for these competencies from MIS graduates, and the continued challenge of differentiation between MIS programming and more technically oriented degrees such as computer science and software engineering.

7. REFERENCES

Agarwal, R., & Lucas, H. C. (2005). The Information Systems Identity Crisis: Focusing on High-Visibility and High-Impact Research. *MIS Quarterly*, 29(3), 381-398.

Aleksić, V., & Ivanović, M. (2016). Introductory Programming Subject in European Higher Education. *Informatics in Education*, 15(2), 163-182.

Apigian, C. H., & Gambill, S. E. (2009). Are We Teaching the IS 2009 Model Curriculum ? *Journal of Information Systems Education*, 21(4), 411-421.

Avouris, N. (2018, November). Introduction to Computing: A Survey of Courses in Greek Higher Education Institutions. *Proceedings of the 22nd Pan-Hellenic Conference on Informatics* (pp. 64-69).

Badampudi, D., Wohlin, C., & Petersen, K. (2015, April). Experiences from Using Snowballing and Database Searches in Systematic Literature Studies. *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering* (pp. 1-10).

Baugh, J. M. (2016). Beginning Programming – Why Teach It and How To Teach It ? *Issues in Information Systems*, 17(3), 138-145.

Becker, B. A. (2019, July). A Survey of Introductory Programming Courses in Ireland. *Proceedings of the 2019 ACM Conference on Innovation and Technology in*

Computer Science Education (pp. 58-64).

Ben Arfa Rabai, L., Cohen, B., & Mili, A. (2015). Programming Language Use in US Academia and Industry. *Informatics in Education*, 14(2), 143-160.

Benbasat, I., & Zmud, R. W. (2003). The Identity Crisis within the IS Discipline: Defining and Communicating the Discipline's Core Properties. *MIS Quarterly*, 27(2), 183-194.

Bohler, J. A., Larson, B., Peachey, T. A., & Shehane, R. F. (2020). Evaluation of Information Systems Curricula. *Journal of Information Systems Education*, 31(3), 232-243.

Cheah, C. S. (2020). Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming: A Literature Review. *Contemporary Educational Technology*, 12(2), 1-14.

Clear, A., Parrish, A. S., Impagliazzo, J., & Zhang, M. (2019). Computing Curricula 2020: Introduction and Community Engagement. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 653-654).

Cummings, J., & Janicki, T. N. (2020). What Skills Do Students Need? A Multi-Year Study of IT/IS Knowledge and Skills in Demand by Employers. *Journal of Information Systems Education*, 31(3), 208-217.

Interactive: The Top Programming Languages 2018. (2018). https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages

Koong, K. S., Liu, L. C., & Xia, C. (2002). A Study of the Demand for Information Technology Professionals in Selected Internet Job Portals. *Journal of Information Systems Education*, 13(1), 21-28.

Koulouri, T., Lauria, S., & Macredie, R. D. (2014). Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches. *ACM Transactions on Computing Education*, 14(4), 1-28.

List of Business Schools in the United States. (n.d.). Wikipedia. https://en.wikipedia.org/wiki/List_of_business_schools_in_the_United_States

Mason, R., & Simon. (2017). Introductory Programming Courses in Australasia in 2016. *Proceedings of the Nineteenth Australasian Computing Education Conference* (pp. 81-89).

McMurtrey, M. E., Downey, J. P., Zeltmann, S. M., & Friedman, W. H. (2008). Critical Skill Sets of Entry-Level IT Professionals: An Empirical Examination of Perceptions from Field Personnel. *Journal of Information Technology Education: Research*, 7(1), 101-120.

Mehmood, E., Abid, A., Farooq, M. S., & Nawaz, N. A. (2020). Curriculum, Teaching and Learning, and Assessments for Introductory Programming Course. *IEEE Access*, 8, 125961-125981.

Murphy, E., Crick, T., & Davenport, J. H. (2017). An Analysis of Introductory Programming Courses at UK Universities. *Programming Journal*, 1(2), 1801-1822.

Orlikowski, W. J. (2001). Desperately Seeking the IT in IT research. *Information Systems Research*, 12(2), 121-134.

Parker, K. R., Chao, J. T., Ottaway, T. A., & Chang, J. (2006). A Formal Language Selection Process for Introductory Programming Courses. *Journal of Information Technology Education: Research*, 5(1), 133-151.

Prasad, C., & Li, X. (2004). Teaching Introductory Programming to Information Systems and Computing Majors: Is There a Difference? *Proceedings of the Sixth Conference on Australian Computing Education* (pp. 261-267).

PYPL PopularitY of Programming Language. (2021). PYPL Index. https://pypl.github.io/PYPL.html

Saldana, J. (2015). *The Coding Manual for Qualitative Researchers*. SAGE Publications.

Siegfried, R. M., Siegfried, J. P., & Alexandro, G. (2016). A Longitudinal Analysis of the Reid List of First Programming Languages. *Information Systems Education Journal*, 14(6), 47-54.

The RedMonk Programming Language Rankings. (2021). https://redmonk.com/sogrady/2021/03/01/language-rankings-1-21/

TIOBE Index. (2021). TIOBE. https://www.tiobe.com/tiobe-index/

Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K., Nunamaker, J. F., Sipior, J. C., & de Vreede, G. J. (2010). IS 2010: Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. *Communications of the Association for Information Systems*, 26, 359-428.

Zhang, X., Zhang, C., Stafford, T. F., & Zhang, P. (2013). Teaching Introductory Programming to IS Students: The Impact of Teaching Approaches on Learning Performance. *Journal of Information Systems Education*, 24(2), 147-155.

AUTHOR BIOGRAPHIES

Tim C. Smith is an assistant teaching professor of information systems and business analytics at Iowa State University. He earned his Ph.D. from Carleton University. His research work includes information systems pedagogy, organizational routines, dynamic capabilities, and technology adoption. Additionally, he has over 20 years of professional experience where he held executive positions in both software development and ICT services industries.

Leslie Jones is an associate professor of mathematics at the University of Tampa, where she recently served as the Assistant Dean of the College of Social Sciences, Mathematics and Education. She received her Ph.D. from Baylor University. Her research interests include population dynamics and applied mathematics education. Her work has been published in such journals as *Journal of Theoretical Biology*, *Gulf and Caribbean Research*, and *Mathematics in School*.

Information Systems & Computing Academic Professionals

Education Special Interest Group

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the ***Journal of Information Systems Education*** have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 2574-3872