Bridging the Gap: Storytelling Alice as a Precursor to Python
Computer Systems Lab 2009-2010

Amanda Gilbert

June 14, 2010

Abstract

The goal of my project is to determine whether or not Storytelling Alice is successful in preparing elementary school students for a career in computer programing. Essentially, I am testing to see how possible it is to bridge a gap between a visual and kid-friendly interface like Storytelling Alice and a non-visual, syntactically difficult interface like Python. If students are able to grasp concepts in Python after a foundational education in Storytelling Alice, it will show that this is a successful way to introduce students to computer science concepts. While Storytelling Alice is a respected way to teach programming to younger students, it is important to know how easy it is to transition into tougher environments. Tests have been done to prove that this language successfully teaches students programming concepts. However, applying the concepts learned in a more difficult language like Python would be a much bigger challenge. If students are unable to make the switch from Alice to other interfaces, their knowledge base in Alice is not helpful in regards to the success about their careers as computer programmers. We are looking to find out if Storytelling Alice can be used as a foundational start to a long-lasting career in programming with other languages (specifically Python).

1 Introduction

Though many people have studied drag-and-drop programming environments like Alice and Scratch, none have looked at the transition from this child-friendly environment to one that requires the user to type in their own code. Another problem this brings forth is the transition from a graphical environment (in this case Alice) to one like Python that is not centered on graphics. Transitioning to Python will require a high ability of individual thinking from the students that must be acquired both from their work with Storytelling Alice and with the work done in the classroom. The transition to Python will not begin until the fourth quarter. Up until this point, the main focus has been teaching concepts and testing their abilities to better focus my teaching. This research project is essential to the very nature of visual programming environments as a mean of teaching computer science to children. If students are not able to use the concepts learned in the environment with drag-and-drop methods to write code in a non-visual, difficult language, they will not be able to be successful in programming. With Storytelling Alice, there is the possiblility that students will become too dependent on the environment and not be able to reason independently and create code without the prompting of the draggable methods. If the students cannot make the connection between the two languages (in other words, if the languages are too different for the students to make a connection), it will prove that Storytelling Alice may not be the best introuction into programming. It has been proven that Storytelling Alice makes concepts more easily understood and enjoyable. However, if the transition is difficult and frustrating, the progress made with Storytelling Alice could be cancelled out and therefore the need for a language like Alice may not be existent.

2 Background

2.1 Development of Storytelling Alice

Storytelling Alice was created by a student at Carnegie Mellon University named Caitlin Kelleher. Last year, two girls did a research project at Cardinal Forest Elementary but focused on Scratch programming. This year, Mr. Allard decided to add Storytelling Alice as another language for students to study and I have been put in charge of the mentoring program for this language. This is the first year we will be testing Storytelling Alice at Cardinal Forest Elementary. However, it is not the first time Storytelling Alice has been used to teach young children to program.

Kelleher was interested in developing Storytelling Alice for many reasons. In her PhD dissertation, she recognized how important diversity is in the field of computer science. She felt storytelling Alice could help attract middle school girls to computer science and geared her language toward that group. Kelleher mentioned a study that found that boys' and girls' ideal technologies were significantly different. Because of this, having a more equal ratio of girls and boys in the field could drastically change the course of technological development.

2.2 Storytelling Alice and Women

One article I read about Alice programming dealt with girls and their ability to use a variation of Alice- Storytelling Alice- to learn programming. The article stated that girls have the same ability as boys do to program. This was encouraging for my project because not only am I a girl interested in programming as a career, I hope to spark an interest in programming in the girls I mentor. These researchers also stated that there were many reasons why girls were less likely to choose to pursue a career in programming. Not only are there social norms that encourage boys to program and not girls, at the middle school level, girls confidence in their abilities relating to math and science becomes deflated. This is actually a very promising fact for our study. Because we are working with Elementary School students, not only are we able to encourage programming for girls before they reach middle school level and lose confidence in science-related subjects, we are also working with them at an age when they are unlikely to know about social norms in programming. Hopefully, Alices easily under- stood set up will help give girls confidence and Alices storytelling nature will be attractive to girls.

2.3 The benefits of Storytelling Alice in Teaching Computer Programming

Another article I read simply studied Storytelling Alice and game making. Both articles mentioned that Alice is very successful in teaching algorithmic thinking and basic programming language and syntax to children. A third article I read discussed the difference between teaching programming with Storytelling Alice and other gaming centered program- ming languages. Because the gaming community is predominately male, a programming language with a gaming nature would be more attractive to males than females. Storytelling, however, is attractive to both males and females and, if taught at an early age, could help equalize the number of boys and girls interested in programming.

There are other benefits to using Storytelling Alice. Concepts that are hard for first-time programmers to understand like variables, parameters, and loops have to be introduced through assignments in other languages. In Alice, however, they are incorporated into the foundations of the language. This could save time in introduction to programming and could help beginners learn more advanced programming techniques at a younger age. Another great thing about Storytelling Alice is that due to its visual nature, parents and peers that do not know how to program can truly appreciate the codes created because they can watch the success through a video-like presentation. Kelleher stated that many probelms that beginning programmers encounter, like invisible state and syntax errors are not an issue for users of Storytelling Alice. She also stated taht in her work with college-level students she found that the retention rate almost doubled after a short introductory Alice class.

3 Development

3.1 Project One

The first project taught the students how to create a world in Alice. The main reason for this lesson was to allow the students to get comfortable with the interface. They learned how to add objects and position them. I also provided a checklist for them so they would be able to see how far along they were in their project and clearly see what they were missing. As I was a new teacher and they were new students of Storytelling Alice, this lesson was a learning experience. I learned from it how to revise my lesson plans to be simpler and easier for the kids to understand and the students learned about objects and how to operate the Storytelling Alice environment.

3.2 Project Two

The second lesson we did in class focused on the children's ability to create methods. This was an important lesson because they were introduced to the drag and drop capabilities of Storytelling Alice. Because they were creating a method, the students were also taught about two control statements: do in order and do together. I also provided them with a checklist to determine if their program was successful. After this project to test the students' understanding of programming, I gave them their first quiz. This quiz expected them to take a story about two girls walking together to class and be able to write the code on paper. Results can be found in the Results and Conclusions section.

3.3 Project Three

This lesson was by far the most complicated and difficult for the kids to understand. Project three focused on the storytelling aspect of Alice. This sole project took up most of the second quarter because it involved many complicated and intricate control statements and methods. The project combined everything we had learned in the year so far. After the project was over, I taught about variables– numbers, booleans, strings and objects. The kids seemed to grasp these concepts very well. I then gave them the second quiz of the year. I learned that the kids became stressed about the prospect of a quiz and therefore the results did not always convey how much they had learned in class. The second quiz caused me to take another look at my way of teaching and rethink my methods of assessment.

3.4 Project Four

When I decided to try to take on teaching the kids Python, I knew that I would have to come up with a program that could be easily completed in both Alice and Python. I decided to create a lesson called "Guess that number". Unlike the other projects we had created in class, this one focused less on the visual aspect of Storytelling Alice and more on the concepts necessary to advance to Python. Many important concepts were taught using this lesson, including; Loops, If/Else Statements, user input and manipulation of variables. In order to assess the students understanding of this code, I began with asking the students to write journal entries. Though it is difficult to assess the journal entries as quantitive data, they were incredibly helpful in seeing where each student was individually and decided what needed to be done in order to be more effective as a teacher. Sample entries can be found in the Appendices. I then created a worksheet that would serve as both a test of understanding and a learning experience. I decided that it was imperative that the kids learn how to write comments for their codes in Storytelling Alice before advancing to Python. By doing so, they would be able to use comments when programming in Python to make it less frustrating and more easy to understand. The results from the comments worksheet were very promising and can be found in the Results and Concųlsions secion. The final thing we completed in the third quarter was a cummulative quiz of everything learned in the year so far. Instead of being Storytelling Alice centered, it was centered on the knowledge of general and basic programming concepts. The results to this were very encouraging in regards to advancing to Python.

3.5 Guess that Number

Guess that Number was the last project created in Storytelling Alice (See Project Four), but also the first project created in Python. The original code was given to the students as a shell, which was missing three pieces of code. I had dialogue with the students back and forth about what needed to go into the code to make it complete and why. I helped the students understand the new syntax by creating a poster that showed them the transition from Storytelling Alice to Python with different blocks of code as examples. The students were all able to successfully create the code and run it, although some did encounter errors at the start (due to spelling, usually). Many of the kids were receptive to the new format and even enjoyed typing in the code better than the previous drag-and-drop interface.

3.6 Guess that Number Modification

The final project we completed for the year was a modification to the code Guess that Number in Python. The change was to have the computer tell the user whether they are guessing too high or too low. This required the students to understand the concept of the game, along with how they would create code to make this happen. I also created a shell for these modifications to make it easier. I worked with shells mostly because I did not want to deal with the confusion that comes with code allignment in Python. I wanted to see first how well they understood the programming concepts. Caroline, one of my better students, called me over to ask how she would modify her code so that it looped nine times instead of three. I challenged her to find which lines she would change and after some thinking, she was able to change her code without my help. This assured me that some of the students truly had gained an understanding about their code, its purpose, and grasped the concept of modifying code to change current processes and add new ones.

4 Discussion of Results and Conclusions

4.1 Results from Assessments

On the first quiz, there were five perfect papers out of the sixteen that were turned in. This was a promising number because this showed me that many thoroughly understood the concepts being taught. I believe the second quiz was very intimidating to the kids because, despite their obvious abilities in class, the average score was a 47.7 percent (see Appendices for graph of distribution of scores). These results forced me to consider other methods of assessments when reviewing the kids' understanding. I knew that I needed a way to allow the students to show me, unprompted, what they were learning without getting frustrated. The journals and the comment worksheets were both very successful. Most kids took the journals seriously but were not threatened by them. I learned which kids were enjoying and understanding Alice and which were struggling. On the comments worksheet, the average grade was an 88.125 percent. It was also very encouraging to see that all grades were relatively high (between a 60 percent and a perfect score). Seven students recieved perfects on this worksheet. Detailed results can be found in the Appendices. Finally, the cummulative quiz we took in class allowed me to see that the students truly had grasped the concepts taught to them in class. The vast majority of the class was able to correctly answer each question with the exception of one extremely complex problem.

4.2 Conclusions

I knew from my testing that my students had learned the concepts necessary to advance to Python from their lessons in Storytelling Alice. However, I still was unsure if they would be able to make the transition. After studying their receptiveness to the new language and their ability to code old ideas in a new language, I have concluded that Storytelling Alice is an effective starting language for beginning programmers. I do think, however, that Storytelling Alice should be used for a longer period of time and non-visual languages like Python should not be introduced until the students are a little older and have an even stronger foundational background in computer science. I think if I had more time with the kids, we would have been able to spend more time on drilling the concepts of computer science in Storytelling Alice and therefore they would have been even more successful with Python. That being said, I do think that using Storytelling Alice as a stepping stone to Python works because the students are less intimidated by the language once they understand the concepts behind programming, and because they do not seem to be bothered by a transition from Storytelling Alice to Python.

References

[1] C. Kelleher, R. Pausch, S. Kiesler,“Storytelling Alice Motivates Middle School Girls to Learn Computer Programming” ,
April 28- May 3, 2007.

[2] L. Werner, J. Denner, M. Bliesner, P. Rex, “Group behaviors for systems with significant dynamics”

[3] C. Kelleher, R. Pausch, “Using Storytelling to Motivate Programming”

[4] C. Kelleher, “Motivating Programming: Using Storytelling to Make Computer Programming Attractive to Middle School Girls”

5 Appendices

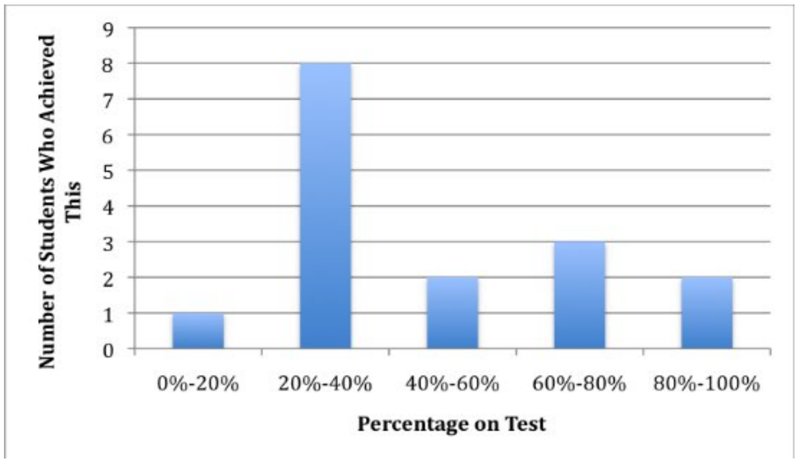
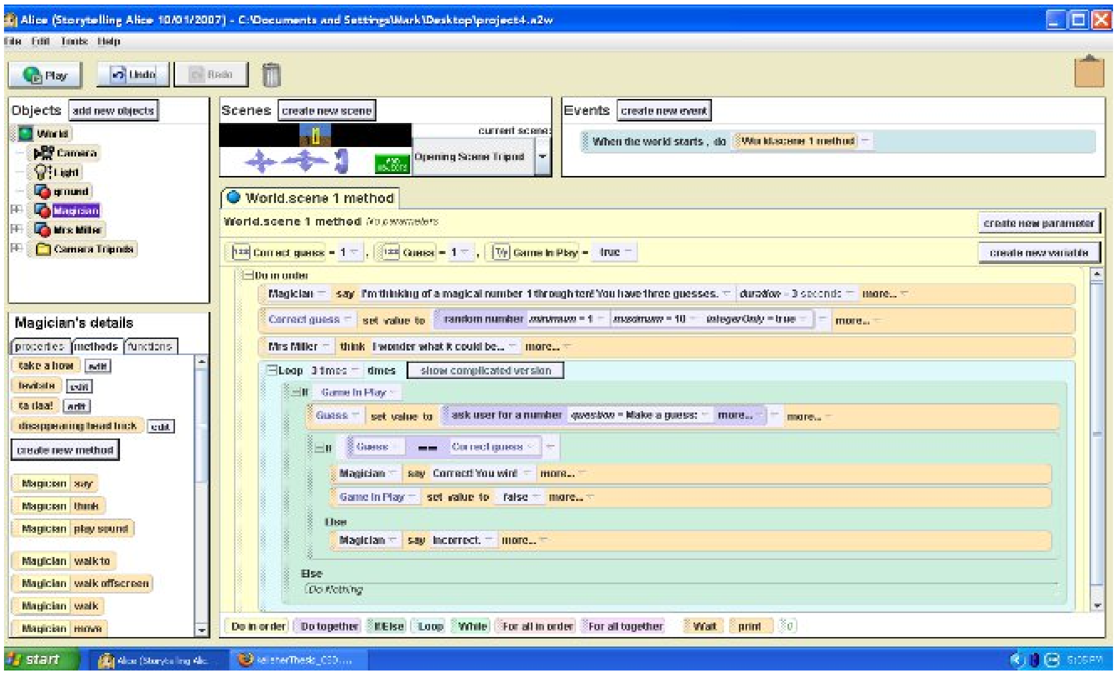
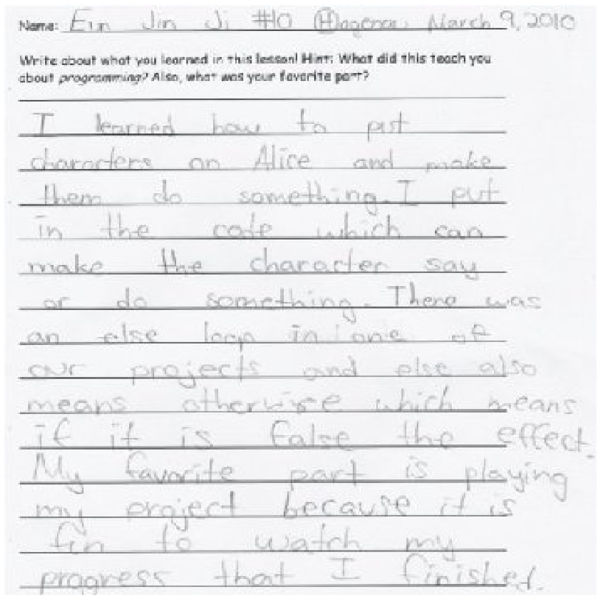


Figure 1: Quiz Two Results



```
World.scene 1 method   No parameters
Correct guess = 1 , Guess = 1 , Game In Play = true
Do in order
    Magician say I'm thinking of a magical number 1 through ten! You have three guesses. duration = 3 seconds more...
    Correct guess set value to random number minimum = 1 maximum = 10 integerOnly = true more...
    Mrs Miller think I wonder what it could be... more...
    Loop 3 times times
        If Game In Play
            Guess set value to ask user for a number question = Make a guess: more... more...
            If Guess == Correct guess
                Magician say Correct! You win! more...
                Game In Play set value to false more...
            Else
                Magician say Incorrect. more...
        Else
            Do Nothing
```

Figure 2: Lesson Four Screenshot

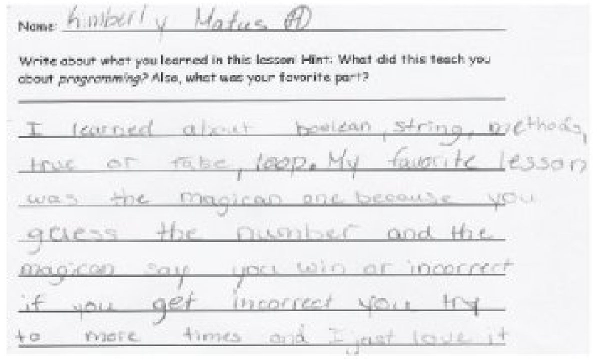


Name: Eun Jin Ji #10 Hagerman March 9, 2010

Write about what you learned in this lesson! Hint: What did this teach you about *programming*? Also, what was your favorite part?

I learned how to put characters on Alice and make them do something. I put in the code which can make the character say or do something. There was an else loop in one of our projects and else also means otherwise which means if it is false the effect. My favorite part is playing my project because it is fun to watch my progress that I finished.

Figure 3: Journal Example 1

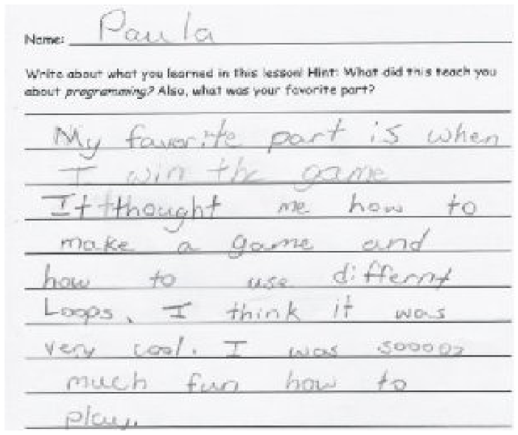


Name: Kimberly Matus

Write about what you learned in this lesson! Hint: What did this teach you about *programming*? Also, what was your favorite part?

I learned about boolean, string, methods, true or false, loop. My favorite lesson was the magican one because you guess the number and the magican say you win or incorrect if you get incorrect you try to more times and I just love it

Figure 4: Journal Example 2



Name: Paula

Write about what you learned in this lesson! Hint: What did this teach you about *programming*? Also, what was your favorite part?

My favorite part is when I win the game It thought me how to make a game and how to use differnt Loops. I think it was very cool. I was sooooo much fun how to play.

Figure 5: Journal Example 3

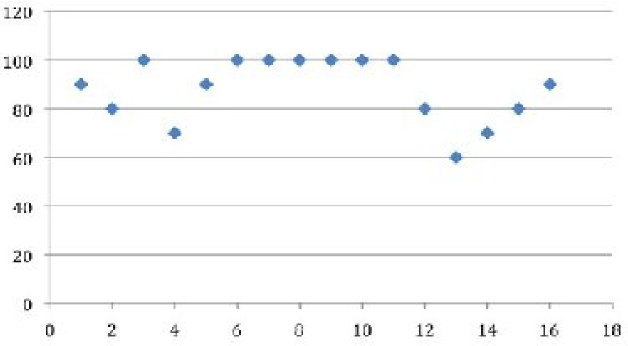
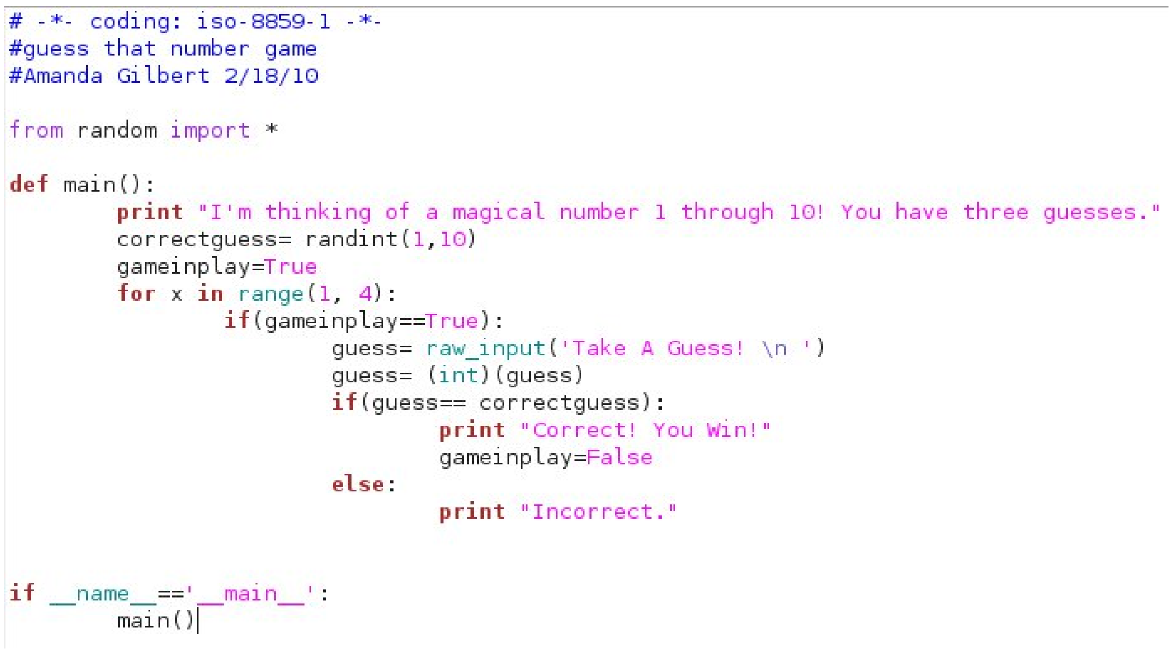


Figure 6: Comments Worksheet Results (Each Point Represents a Student's Score)



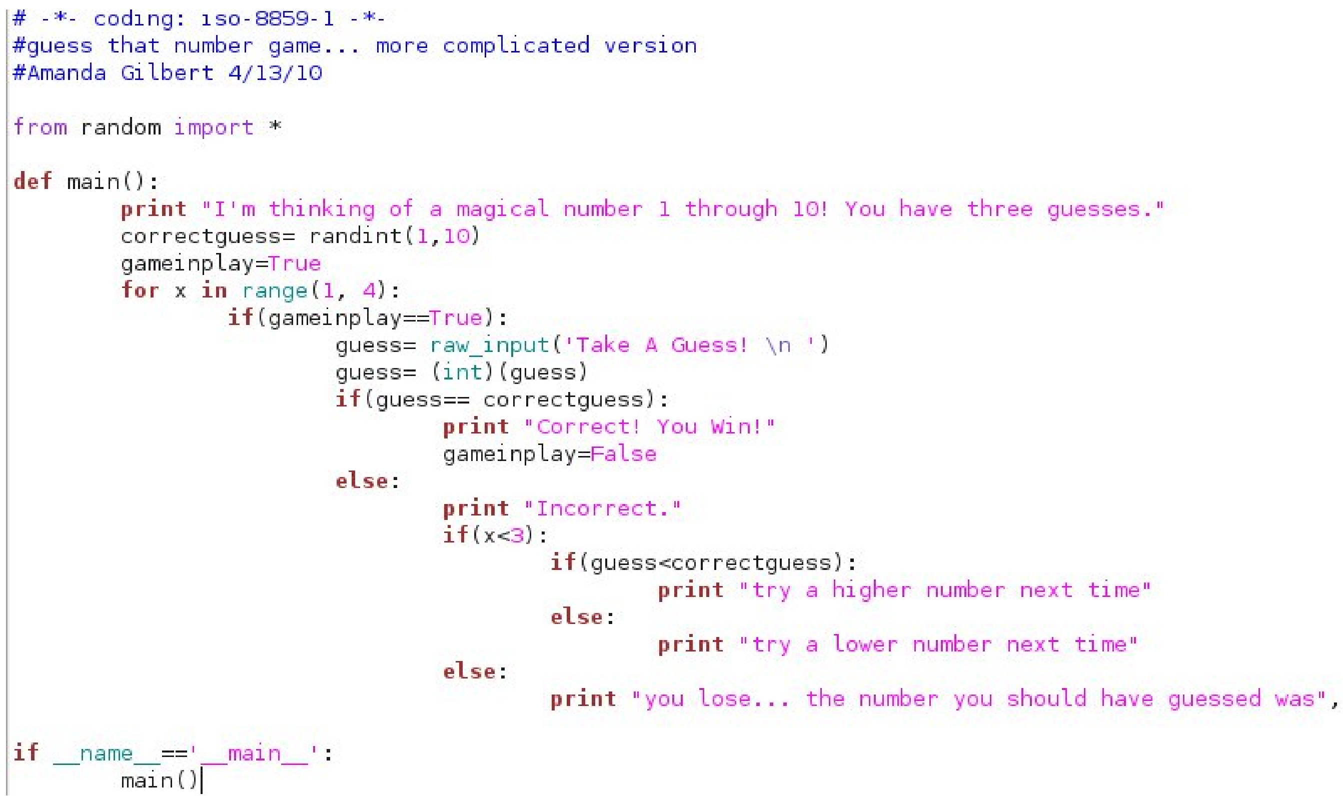
```
# -*- coding: iso-8859-1 -*-
#guess that number game
#Amanda Gilbert 2/18/10

from random import *

def main():
        print "I'm thinking of a magical number 1 through 10! You have three guesses."
        correctguess= randint(1,10)
        gameinplay=True
        for x in range(1, 4):
                if(gameinplay==True):
                        guess= raw_input('Take A Guess! \n ')
                        guess= (int)(guess)
                        if(guess== correctguess):
                                print "Correct! You Win!"
                                gameinplay=False
                        else:
                                print "Incorrect."


if __name__=='__main__':
        main()
```

Figure 7: Guess That Number (This game was the transitioning concept from Storytelling Alice to Python)



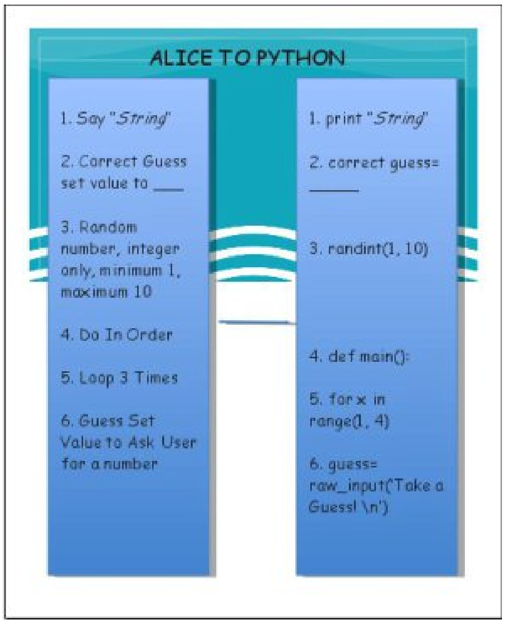
```
# -*- coding: iso-8859-1 -*-
#guess that number game... more complicated version
#Amanda Gilbert 4/13/10

from random import *

def main():
        print "I'm thinking of a magical number 1 through 10! You have three guesses."
        correctguess= randint(1,10)
        gameinplay=True
        for x in range(1, 4):
                if(gameinplay==True):
                        guess= raw_input('Take A Guess! \n ')
                        guess= (int)(guess)
                        if(guess== correctguess):
                                print "Correct! You Win!"
                                gameinplay=False
                        else:
                                print "Incorrect."
                                if(x<3):
                                        if(guess<correctguess):
                                                print "try a higher number next time"
                                        else:
                                                print "try a lower number next time"
                                else:
                                        print "you lose... the number you should have guessed was",

if __name__=='__main__':
        main()
```

Figure 8: Guess That Number Modification (This modification told the user whether or not they guesed too high or too low.)



ALICE TO PYTHON

1. Say "*String*"	1. print "*String*"
2. Correct Guess set value to ____	2. correct guess= ______
3. Random number, integer only, minimum 1, maximum 10	3. randint(1, 10)
4. Do In Order	4. def main():
5. Loop 3 Times	5. for x in range(1, 4)
6. Guess Set Value to Ask User for a number	6. guess= raw_input('Take a Guess! \n')

Figure 9: Alice To Python Poster (This poster showed the kids what code looked like in Python as compared to Storytelling Alice.)