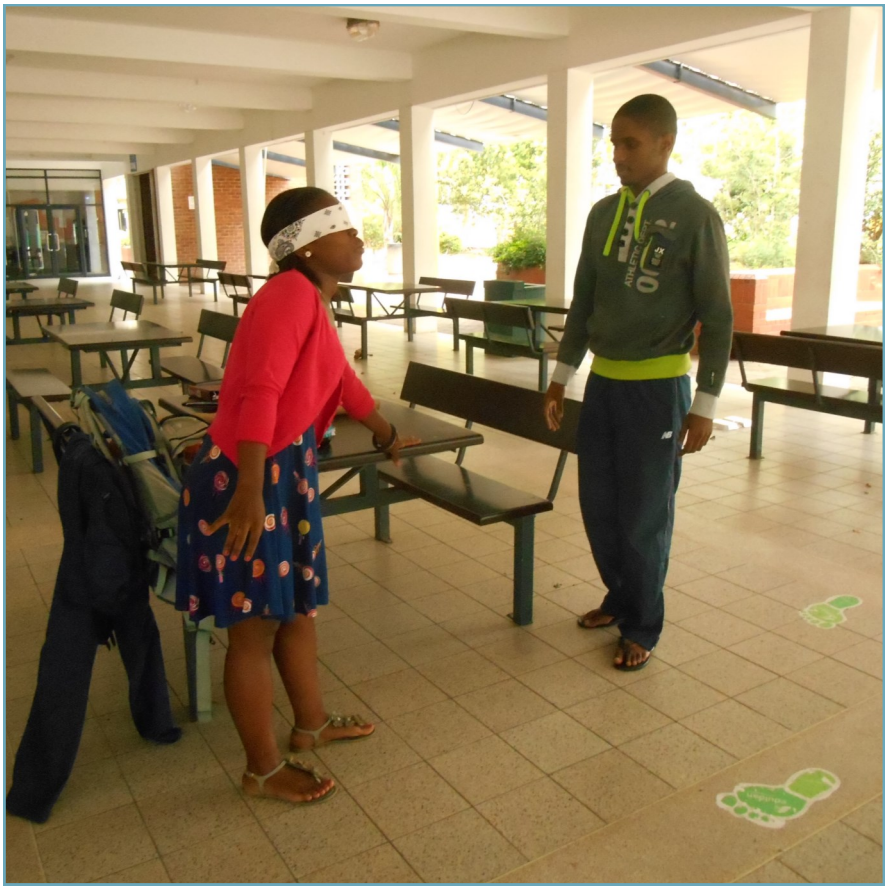
Computer Languages

Outreach Program Lesson Plan

[Picture]



WAAW foundation

Working To Advance STEM Education for African Girls

WAAW Foundation is non-profit organization dedicated to bringing hands-on STEM education to girls all over Africa.

Our Mission: To increase the pipeline of African women in Science, Technology, Engineering and Math (STEM) disciplines and to ensure this talent is engaged in African innovation.

Our Vision: To eradicate poverty in African through female education and science and technology innovation.

WAAW foundation

Working To Advance STEM Education for African Girls

P.O. Box 1691
Wylie, Texas 75098
1-972-763-5924
www.waawfoundation.org

"LIKE" us on facebook— www.facebook.com/waawfoundation
"FOLLOW" us on twitter— www.twitter.com/waaw_foundation
"SUBSCRIBE" to our newsletter— http://eepurl.com/ihwpU

Computer Languages

Class Description-

In this class, students will participate in activities to explore the basics of how computer languages work. They will work in groups to write their own "programs."

Class Outcomes-

-Students will understand the basics of how to give instructions to computers.
-Students will learn about the elements of communication that differ when speaking with a human vs. a computer.
-Students will be able to explain what a bug is in computer code, and how to approach "de-bugging" a program.

Materials List-

The kit to teach this class should include:
-Paper and pencils (one for each student)
-Blindfolds (one per every group of 3-5 students)

Computer Languages

Introduction (5 minutes)

Communication is a fundamental part of being human. We use words, actions, voice inflection, and facial expressions to convey messages to each other every day. But what if we don't need to communicate to another human? What if, instead, we were going to try to convey a message to a computer?

Today we are going to become computer programmers. We are going to explore how we can give instructions that a computer would be able to carry out.

Drawing Instructions (20 minutes)

Divide students into groups of 4-5. Hand one student in each group a simple image made up of basic geometrical shapes. (You could create your own image, or use one from the appendix at the end of this lesson plan.) Make sure that the other students do not see the image! Have that student describe how to draw the picture to the other students in their group. The other students should follow their instructions to recreate the image. Students are allowed to ask the describer questions, and the describer may use gestures, words, and give feedback to help the other students draw their pictures correctly. They can do anything except show the other students the image!

Students should be aiming to recreate the picture exactly. Size, orientation, and placement on the paper all matter!

After the drawings are complete, have everyone reveal their pictures. Lead a quick discussion to share and reflect on results.

- How close were they to the original?
- What things happened that helped you to get the image right?
- What did your describer do in order to help you draw the image? Did they use just words, or were there other things involved?

When we talk to each other as human beings, we are not just using words to describe things to each other. Gestures, facial expressions, voice inflections, and reactions all allow us to communicate more effectively. So what would happen if you took away these elements of communication?

Computer Languages

Drawing Instructions continued…

Repeat the activity with a different image and the same student acting as the describer, but this time, only the describer is allowed to speak (the other students are not allowed to ask questions) and no one is allowed to see the describer (have them stand around a corner, behind a sheet, on the other side of a door, etc.) Once again, the describer should give instructions on how to draw the image, and the rest of their group should do their best to follow the instructions.

After they have finished, have the students reveal their drawings.

- How did the drawings turn out? Were they more or less accurate than the previous time?
- What was different this time that made the activity more difficult?
- Ask the describer: Was it more or less difficult to give instructions this time? What made it that way?

When we take away the other elements of communication and can only use words, relaying instructions becomes a lot more difficult. Now how does this relate to a computer? In this activity, students were not able to see facial expressions or gestures, or get feedback from the describer. Computers are not able to see or sense these things either! Computers can only follow very specific instructions from us, and only know certain words. They can't ask us questions if they don't understand, and they can't explain to us how our instructions are confusing.

It is important that you have the describer use a NEW image, not the same one from the previous exercise. If they were to use the same one, the students remember what it was supposed to look like, and the meaning of the exercise is lost.

Computer Languages

Writing a Program (40 minutes)

Now we are going to practice giving very specific instructions to each other, just as we would need to give them to a computer! Set a chair in the middle of the room . In their groups, the students' task is to write a set of instructions to walk to the chair and sit down in it. This may seem simple, but the students must write the instructions like they would write them for a computer– in this case, our computer will be represented by a student in a blindfold. The instructions will be read to the blindfolded student without any additional information– only what is written on the paper will be read to guide the student into the chair. Ask the students: How will you convey distance to walk? Direction to turn? How to sit down? What order will they put the instructions in? All of these things must be considered.

Encourage students to test out their programs as the go, as opposed to trying to write the entire program all at once.

Give groups time to write their instructions. Once each group has finished, have them blindfold one member of their group and test them out. The blindfolded student must follow the instructions exactly as they hear them. *Have an additional student walk along side the blindfolded student to stop them if they are about to run into something. Make sure the blindfolded students are safe!

Have groups work separately. They will naturally come up with their own language (or syntax) that they will use to give directions for each other. For example, one group might tell someone to turn by saying "rotate 90 degrees" while another group might say simply "turn left" or "turn right." This is a good opportunity to discuss that there are many different computer languages developed by different people for different functions.

After each group has had a chance to try them out, lead a discussion about what happened.

- Did your instructions work? What about your instructions worked well, and what, if anything, would you change if you could?
- What would we call the instructions we would give to a computer? (Programmers call these instructions "code." A complete set of code is a "program.")

When we as humans look at a task such as sitting in a chair, we don't have to think about the individual steps needed in order to do something. We have past experiences stored in our memories, so we can complete these tasks without thinking about how to do them. A computer does not have this intuition to rely on, so we must tell it specifically what it needs to do!

Computer Languages

Writing a Program continued...

Next, give the students a task that is a little bit more complex. Maybe it will involve turns, going through a doorway, going up or down stairs, etc. Tasks will vary on the room you have available to you. Again, students should write a program to guide a blindfolded student to complete the task.

Again, give students time to write their instructions, and try them out.

- As you add more instructions, does the task become more difficult? When we have more lines of code, we are much more likely to make mistakes.
- Did you write all of the instructions at once and try them all at once? Did any groups write their "code" in sections and test these sections independently? Real programmers do this all the time! It's easier to find errors in your code if you test it for errors section by section. Computer programmers call this "de-bugging." A "bug" is a mistake somewhere in the code.
- Did all groups come up with the exact same program? Probably not! You created your own "language" within your group to describe what to do. In the same way, there are many different computer programming languages used by programmers today.

Give students plenty of time to test, correct, and re-write their instructions. Have groups take turns running their programs for the rest of the class.

Conclusion (5 minutes)

At the end of class, bring the students together for a final discussion of the activities.

- Why do we have to write such specific instructions for computers?
- What is code? What is a program? What is a bug, and "de-bugging"?
- Why should you de-bug in sections?