# DISCOVER

# 4-H CODE CLUBS

EXTENSION
UtahStateUniversity

# DISCOVER 4-H CODE CLUBS

Paul Hill | Stacey MacArthur
Utah State University Extension

## Description

The Discover 4-H Clubs series guides new 4-H volunteer leaders through the process of starting a 4-H club or provides a guideline for seasoned volunteer leaders to try a new project area. Each guide outlines everything needed to organize a club and hold the first six club meetings related to a specific project area.

## Purpose

The purpose is to create an environment for families to come together and participate in learning activities that can engage the whole family, while spending time together as a multi-family club. Members will experiment with new 4-H project areas.

## What is 4-H?

4-H is one of the largest youth development organizations in the United States. 4-H is found in almost every county across the nation and enjoys a partnership between the U. S. Department of Agriculture (USDA), the state land-grant universities (e.g., Utah State University), and local county governments.

4-H is about youth and adults working together as partners in designing and implementing club and individual plans for activities and events. Positive youth development is the primary goal of 4-H. The project area serves as the vehicle for members to learn and master project-specific skills while developing basic life skills. All projects support the ultimate goal for the 4-H member to develop positive personal assets needed to live successfully in a diverse and changing world.

Participation in 4-H has shown many positive outcomes for youth. Specifically, 4-H participants have higher participation in civic contribution, higher grades, increased healthy habits, and higher participation in science than other youth (Learner et al., 2005).

Updated June 2014

EXTENSION UtahStateUniversity

Utah State University is an affirmative action/equal opportunity institution.

## Utah 4-H

4-H is the youth development program of Utah State University Extension and has more than 90,000 youth participants and 8,600 adult volunteers. Each county (Daggett is covered by Uintah County) has a Utah State University Extension office that administers the 4-H program.

## The 4-H Motto

"To Make the Best Better!"

## The 4-H Pledge

I pledge: My HEAD to clearer thinking, My HEART to greater loyalty, My HANDS to larger service and My HEALTH to better living, For my Club, my Community, my Country, and my world.

## 4-H Clubs

What is a 4-H Club? The club is the basic unit and foundation of 4-H. An organized club meets regularly (once a month, twice a month, weekly, etc.) under the guidance of one or more volunteer leaders, elects its own officers, plans its own program, and participates in a variety of activities. Clubs may choose to meet during the school year, only for the summer, or both.

## Club Enrollment

Enroll your club with your local Extension office. Each member will need to complete a Club/member Enrollment form, Medical History form, and a Code of Conduct/Photo Release form (print these from the www.utah4h.org website or get them from the county Extension office).

## Elect Club Officers

Elect club officers during one of your first club meetings. Depending on how many youth you have in your club, you can decide how many officers you would like. Typical officers will include a president, vice president, pledge leader, and secretary. Other possible officers or committees are: song leader, activity facilitator, clean-up supervisor, recreation chair, scrapbook coordinator, contact committee (email, phone, etc.), field trip committee, club photographer, etc. Pairing older members with younger members as Sr. and Jr. officers may be an effective strategy to involve a greater number of youth in leadership roles and reinforce the leadership experience for both ages. Your club may decide the duration of officers–six months, one year, etc.

## A Typical Club Meeting

Follow this outline for each club meeting:

- ☐ Call to order–President
- ☐ Pledge of Allegiance and 4-H Pledge–Pledge Leader (arranges for club members to give pledges)
- ☐ Song–Song Leader (leads or arranges for club member to lead)
- ☐ Roll call–Secretary (may use an icebreaker or get acquainted type of roll call to get the meeting started)
- ☐ Minutes of the last meeting–Secretary
- ☐ Business/Announcements–Vice President
- ☐ Club Activity–arranged by Activity Facilitator and includes project, lesson, service, etc. These are outlined by project area in the following pages.
- ☐ Refreshments–arranged by Refreshment Coordinator
- ☐ Clean Up–led by Clean-up Supervisor

## Essential Elements of 4-H Youth Development

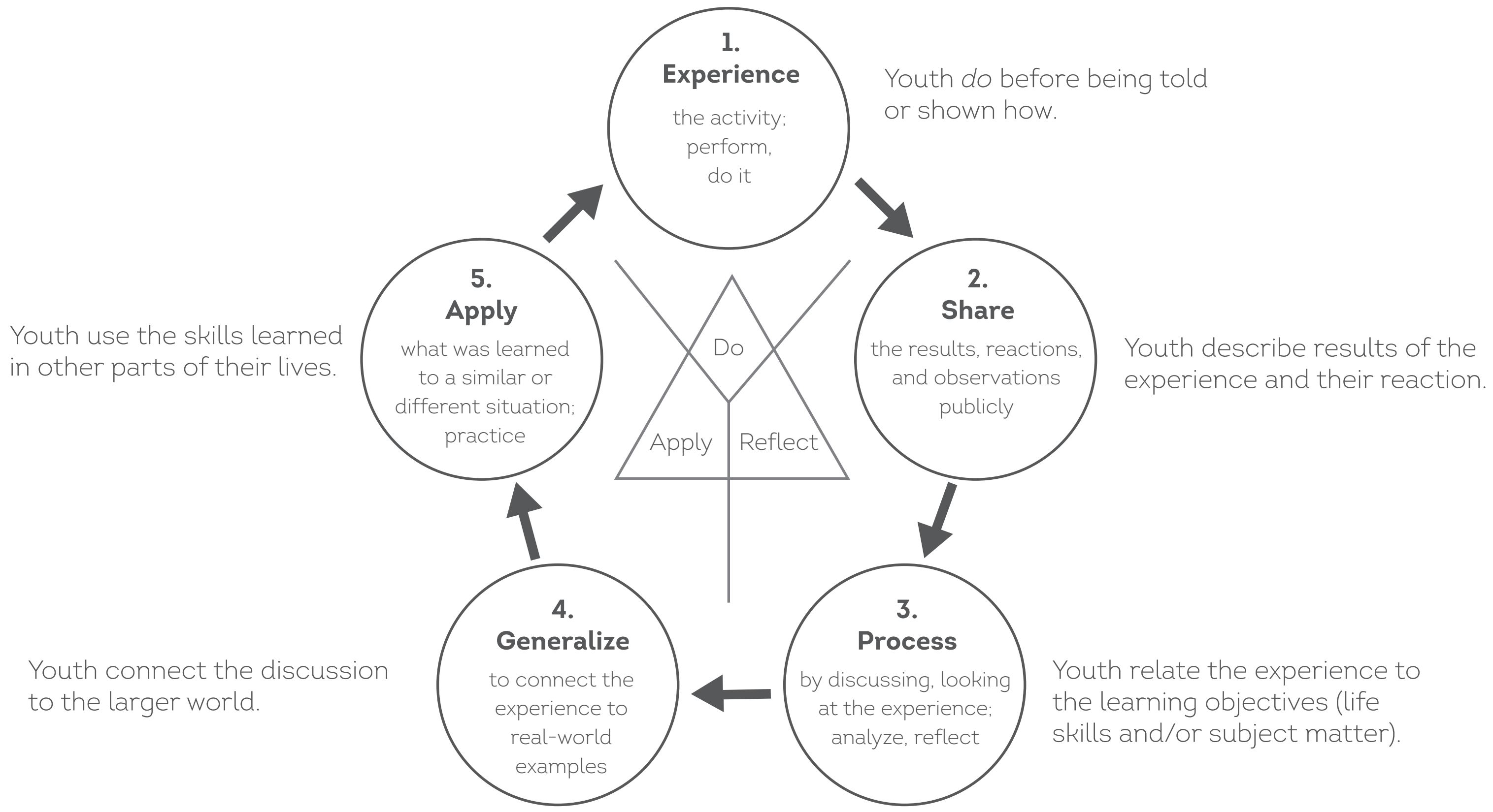
The essential elements are about healthy environments. Regardless of the project area, youth need to be in environments where the following elements are present in order to foster youth development.

1. **Belonging**: a positive relationship with a caring adult; an inclusive and safe environment.
2. **Mastery:** engagement in learning; opportunity for mastery.
3. **Independence:** opportunity to see oneself as an active participant in the future; opportunity to make choices.
4. **Generosity:** opportunity to value and practice service to others.

(Information retrieved from: http://www.4-h.org/resource-library/professional-development-learning/4-h-youth-development/youth-development/essential-elements/)

## 4-H "Learning by Doing" Learning Approach

The Do, Reflect, Apply learning approach allows youth to experience the learning process with minimal guidance from adults. This allows for discovery by youth that may not take place with exact instructions.


1. Experience
the activity; perform, do it
Youth *do* before being told or shown how.
2. Share
the results, reactions, and observations publicly
Youth describe results of the experience and their reaction.
3. Process
by discussing, looking at the experience; analyze, reflect
Youth relate the experience to the learning objectives (life skills and/or subject matter).
4. Generalize
to connect the experience to real-world examples
Youth connect the discussion to the larger world.
5. Apply
what was learned to a similar or different situation; practice
Youth use the skills learned in other parts of their lives.
Do
Apply
Reflect

## 4-H Mission Mandates

The mission of 4-H is to provide meaningful opportunities for youth and adults to work together to create sustainable community change. This is accomplished within three primary content areas, or mission mandates, - citizenship, healthy living, and science. These mandates reiterate the founding purposes of Extension (e.g., community leadership, quality of life, and technology transfer) in the context of 21st century challenges and opportunities. (Information retrieved from: http://www.csrees.usda.gov/nea/family/res/pdfs/Mission_Mandates.pdf)

1. **Citizenship:** connecting youth to their community, community leaders, and their role in civic affairs. This may include: civic engagement, service, civic education, and leadership.
2. **Healthy Living:** promoting healthy living to youth and their families. This includes: nutrition, fitness, social-emotional health, injury prevention, and prevention of tobacco, alcohol, and other drug use.
3. **Science:** preparing youth for science, engineering, and technology education. The core areas include: animal science and agriculture, applied mathematics, consumer science, engineering, environmental science and natural resources, life science, and technology.

## Getting Started

1. Recruit one to three other families to form a club with you.
   a. Send 4-H registration form and medical/photo release form to each family (available at utah4h.org)
   b. Distribute the Discover 4-H Clubs curriculum to each family
   c. Decide on a club name
   d. Choose how often your club will meet (e.g., monthly, bi-monthly, etc.)
2. Enroll as a 4-H volunteer at the local county Extension office (invite other parents to do the same)
3. Enroll your club at the local county Extension office
   a. Sign up to receive the county 4-H newsletter from your county Extension office to stay informed about 4-H-related opportunities.
4. Identify which family/adult leader will be in charge of the first club meeting.
   a. Set a date for your first club meeting and invite the other participants.
5. Hold the first club meeting (if this is a newly formed club).
   a. See *A Typical Club Meeting* section above for a general outline.
      i. Your activity for this first club meeting will be to elect club officers and to schedule the six project area club meetings outlined in the remainder of this guide. You may also complete a-d under #1 above.
   b. At the end of the first club meeting, make a calendar outlining the adult leader in charge (in partnership with the club president) of each club meeting along with the dates, locations, and times of the remaining club meetings.
6. Hold the six project-specific club meetings outlined in this guide.
7. Continue with the same project area with the 4-H curriculum of your choice (can be obtained from the County Extension Office) OR try another Discover 4-H Club project area.

## Other Resources

Utah 4-H website: www.Utah4-h.org
National 4-H website: www.4-h.org
4-H volunteer training:
To set up login:
http://utah4h.org/htm/volunteers/get-involved/new-volunteer-training
To start modules: http://4h.wsu.edu/volunteertraining/course.html
(password = volunteer)

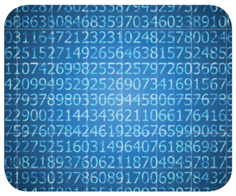
## References

Information was taken from the Utah 4-H website (utah4h.org), the National 4-H Website (4h.org), the Utah Volunteer Handbook, or as otherwise noted.

Lerner, R., M. et al., (2005). Positive youth development, participation in community youth development programs, and community contributions of fifth grade adolescents: Findings from the first wave of the 4-H Study of Positive Youth Development. Journal of Early Adolescence, 25(1), 17-71.

**We would love feedback or suggestions on this guide; please go to the following link to take a short survey:**
http://tinyurl.com/lb9tnad

# 4-H CODE CLUB *Meetings*



Paul Hill | Stacey MacArthur
Utah State University Extension

DISCOVER
4-H CODE CLUBS

# 4-H Club Meeting 1
## Welcome to Python

**Supplies**

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Writing code is the process of telling your computer how to solve problems with step-by-step instructions. This means you have to learn how to speak your computer's language. The language you'll be learning in these Discover 4-H activities is Python. Python is free, easy to download, and a relatively straightforward programming language to work with. For general Python information, visit www.python.org.

In learning computer science, you are exploring computational thinking (CT). CT involves a set of problem-solving skills and techniques that software engineers use to write programs that are the foundation for computer applications people use every day, like: search, email, calculators, and maps. Youth engaged in coding will begin to recognize the relationships between the different problems they're solving as 4-H clubs and how they apply to life in general.

## Getting Started

### 1. INSTALL PYTHON

Get started by downloading Python 2.7 (which is free and open source) by following these simple instructions:

| Windows | Mac |
|---|---|
| 1. Click the link to download the installer: Windows.<br>2. Double click Python 2.7.msi to run the installer.<br>3. Follow the install dialog. By default, the program will be saved in a directory called Python27.<br>4. To start the Python interpreter, find "Python 2.7" in your Start Menu and start IDLE.<br>  a. Defaults to Interactive Mode.<br>5. (Optional) Once the interpreter is running, choose "File → New Window" to open the Python Editor. | 1. Click the link to download the installer: Mac OS.<br>2. Open the file and double click on the Python.mpkg icon (the open box icon).<br>3. Click through the installation instructions.<br>4. Open the Finder and click "Applications."<br>5. Double-click the Python 2.7 folder.<br>6. Double-click IDLE to start the Python interpreter.<br>  a. Defaults to Interactive Mode.<br>7. (Optional) Once the interpreter is running, choose "File → New Window" to open the Python editor. |

Now that you have Python installed, go into your Python 2.7 folder and open IDLE (this is the Python Interpreter). This file opens as Python Shell - now you can jump right in and start editing!

## 2. MATH SYMBOLS IN PYTHON

Python uses several symbols for mathematical operations that vary from what you usually see in math class at school. Experiment with these examples below while looking for patterns that indicate the meaning of each symbol.

| Operator | Symbol | Example 1 | Example 2 | Explanation |
|---|---|---|---|---|
| Exponent | ** | >>> 7**2<br>49 | >>> 2**4<br>16 | Repeated multiplication |
| Modulo | % | >>> 10%3<br>1 | >>> 24%5<br>4 | Remainder after division |
| Division | / | >>> 19/5<br>3 | >>> 21/4.0<br>5.25 | Divides, rounds down for integers |
| Floor Division | // | >>> 9//2<br>4 | >>> 9.0//2.0<br>4.0 | The division of operands where the result is the quotient in which the digits after the decimal point are removed |

You are probably familiar with these mathematical symbols already, but experiment with these as well.

| Operator | Symbol | Example 1 | Example 2 | Explanation |
|---|---|---|---|---|
| Addition | + | >>> 2 + 98<br>100 | >>> 'Hello' + 'World'<br>'Hello World' | Adds values on either side of the operator |
| Subtraction | - | >>> 1000 - 1<br>999 | >>> 95 - 4<br>91 | Subtracts values on either side of the operator |
| Multiplication | * | >>> 10 * 10<br>100 | >>> 'hello' * 3<br>'hello hello hello' | Multiplies values on either side of the operator |
| Equal/Not Equal | == | >>> 10 == 10<br>True | >>> 'apples' != 'oranges'<br>True | The comparison of two arbitrary figures |
| Greater Than/ Less Than | > / < | >>> 'z' > 'a'<br>True | >>> 10 < 10<br>False | A relation that holds between two values when they are different |

## 3. PARENTHESES IN PYTHON

Parentheses can change the result of an arithmetic expression. In the ***Python Shell***, enter the following expressions to understand how parentheses affect the order of operations.

| | | |
|---|---|---|
| >>> 6 + 10 / 2<br>11<br>>>> (6 + 10) / 2<br>8 | >>> 2 + 3 * 5<br>17<br>>>> (2+3) * 5<br>25 | >>> 6 + (8 / 4) - 2 * 3<br>2<br>>>> (6 + (8 / 4) - 2) * 3<br>18 |

Notes:

- Be sure to include all the parentheses you need to perform the correct order of operations.
- Be sure to close any parentheses you open. Examples >> Closed: (2+3)*5. Not Closed: (2+3*5.

## 4. TESTING FOR EQUALITY IN PYTHON

Python has three different symbols to represent different types of equality.

| Example 1 | Example 2 | Example 3 |
|---|---|---|
| >>> x = 7<br>>>> x * 4<br>28<br>>>> x ** 2<br>49<br>>>> x<br>7<br>>>> x = x * 4<br>28 | >>> x = 10<br>>>> y = 2<br>>>> x == y     #Equal to<br>False<br>>>> x != y     #Not equal to<br>True<br>>>> y = 10<br>>>> x == y<br>True | >>> x = 10<br>>>> y = 2<br>>>> x < y     #Less than<br>False<br>>>> x <= 10     #Less than or equal to<br>True<br>>>> x >= 11     #Greater than or equal to<br>False |

## 5. PROGRAMS IN PYTHON

A. To use Python in interactive mode, at the >>> prompt, type a command and press Enter.
Here's an example: **>>> print 'Hello World'**

```
Hello World
```

B. To use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

C. Open a Python editor (B2 above) and type or copy the code on the following page. Then run the code (F5).

```
#Asks the user to type in an answer and press enter.
```

```
temp_in_f = input ('What is the temperature in Fahrenheit: ')

temp_in_c = (temp_in_f - 32) * 5/9.0

#Displays the information
print temp_in_f, 'Fahrenheit is', temp_in_c, 'in Celsius.'
```

```
#Sample output
What is the temperature in Fahrenheit: 80
80 Fahrenheit is 26.6666666667 in Celsius.
```

Notes:

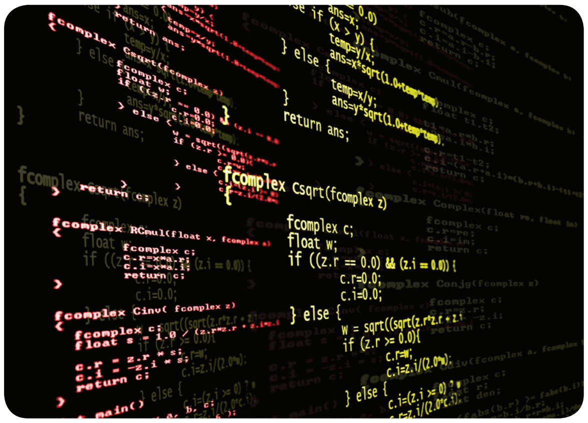
- Quotation marks and commas are **NOT optional** in Python. If Python gives you an error message, double check to make sure you included all quotation marks and commas.
- Comments start with a **#** and are ignored by Python. It is used to help others read their code.
- Variables in Python with more than one word are separated by an underscore ex: variable_name.
- To create an underscore, hold Shift and press the key to the right of the number 0.

## 6. LOOK UP THE FOLLOWING

- Source Code –
- Computer Programmer –
- Variable –

## 7. MAKE IT OFFICIAL

- Club name –
- Team colors –
- Mascot –



## Reflect

- What did you learn today that you did not know before?
- How does writing code (instructions) apply in other situations?
- How do parentheses affect the order in which Python performs a calculation? Why would it be important to use parentheses?
- How many values can you store in each variable? (A: 1)
- What is Python editor for? (A: to write your own programs)
- What are the symbols Python uses to evaluate mathematical expressions? (A: exponents (**), modulo (%), multiplication (*), division (/), and equality (==, >=, <))

## Apply

- How did you work together as a team on this activity?
- What is a career area that relies on teamwork? Or what is a career area that does not rely on teamwork?
- When do you work as a team at home? At school?
- Why is teamwork so important?

## Debriefing

Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

# 4-H Club Meeting 2
## Logic and Loops in Python

### Supplies

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Writing code includes telling your computer how to make decisions. As a programmer, you specify one or more conditions for the program to evaluate. Next, you write a statement(s) to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false. Loop statements, in particular, allow you to execute a statement, or group of statements, multiple times.

In this activity you will use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

## Getting Started

### 1. CONDITIONAL LOGIC IN PYTHON

Python uses conditional logic to "branch out" and handle all possible scenarios to solve a problem.
Copy this code into your IDLE Shell and run it:

```
secret_number = 7
guess = input ('Guess what number I am thinking of: ')

if guess == secret_number:
   print 'You got it!'
elif guess < secret_number:
   print 'Too low.'
else:
   print 'Too high.'
```

Change the secret number and have another club member try to guess it.

**Note:**
You must include a colon (:) at the end of each if, elif, and else statement, which tells Python to automatically indent the next line.

## 2. PROGRAMMING WITH LOOPS IN PYTHON

Python uses a "loop" to complete a repetitive task very quickly. A "while loop" performs a specified task over and over while a specified condition is true. Run this "blastoff" program below to see the loop in action.

```
number = input ('Enter a number between 1 and 50: ')

while number >= 0:
   print number
   number = number - 1
print 'Blastoff!'
```

Python can count the total number of multiples of 3 between 0 and 100 by adding a variable that we named counter in our program.

```
x = 3
counter = 0

while x < 100:
   print x
   counter = counter + 1
   x = x + 3
print 'There are', counter, 'multiples of 3 between zero and one hundred.'
```

Notes:

- We set counter equal to zero at the beginning of the program.
- We must "initialize" all variables before using them in a loop.
- This tells Python that it will need to save some space for the variable before it runs through the loop that follows.

## 3. EXCITING MATH IN PYTHON

The same programming skills that you use when solving math problems can actually create games too.

The problem with the previous "guess the number" program has two drawbacks: 1. It needs to be run over and over, and 2. The programmer knows the secret number and cannot play. Python has a built-in function called random that will pick a random number between any two values that you want. Now you can play too!

```
import random

guess = input('Guess the number between 1 and 25: ')
secret_number = random.randint(1, 25)
while guess != secret_number:
  guess = input('Try again: ')

print 'You got it!  My number is', secret_number
```

Add the conditional logic from above to help the user out.

```
import random
guess = input('Guess the number between 1 and 25: ')
secret_number = random.randint(1, 25)
while guess != secret_number:
  if guess < secret_number:
    guess = input('Too low, try again: ')
  else:
    guess = input('Too high, try again: ')

print 'You got it!  My number is', x
```

## Reflect

- What did you learn today that you did not know before?
- Can we use conditional logic to account for several possible cases of the same problem?
- Can we use “while loops” to perform a repetitive calculation quickly and accurately?

## Apply

- How was your teamwork in this activity?
- What is a career for a computer programmer?
- Why would a computer programmer need teamwork skills?

## Debriefing

Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

# 4-H *Club Meeting 3*

## Variables, Lists, and Functions in Python

**Supplies**

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Writing code includes telling your computer how to make decisions. As a programmer, you specify one or more conditions for the program to evaluate. Next, you write a statement(s) to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false. Loop statements, in particular, allow you to execute a statement, or group of statements, multiple times.

In this activity you will use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

## Getting *Started*

### 1. VARIABLES

A variable is a name that refers to a value, just a name for "something else" so you can use that name rather than the "something else" as you code. The purpose of naming variables is to make your code read more like English so you don't get lost when you go back and try to read it.

When naming variables follow these guidelines: 1. Make names easy to read. 2. Indicate use. 3. Start with a letter. 4. Do not use reserved words. Here are some good and bad examples:

| Good Variables | Bad Variables |
|---|---|
| description | it |
| project_name | this |
| long_variable_name | mine |

An **assignment statement** (variable_name = value) creates a new variable and gives it value:

```
>>> n = 15
>>> pi = 3.1415926535897931
```

To display the value of a variable, just use a print statement:

```
>>> print n
17
>>> print pi
3.14159265359
```

Make your own variable:

1. Assign the first initial in your name equal to your favorite number.
2. Assign the last initial in your name to 3.14 (pi).
3. Multiply the variables you have assigned.
4. Experiment with adding, subtracting, and dividing other variables.

Other things to remember:

Anything after a # sign on a line is ignored:

```
>>> F = m*a    #Force = mass * acceleration
```

Use 3 single quote marks for multiple line comments:

```
>>> '''This is a comment that
spans multiple lines.'''
```

## 2. LISTS (DOCUMENTATION)

The sequence is the most basic data structure in Python. Each element of a sequence is assigned a number, referred to as its position, or index. The first index is always zero, the second index is one, and so on.

There are six built-in types of sequences in Python. The most common sequences are **lists** and **tuples**.

The list is written as a list of comma-separated values (items) between square brackets. A **tuple** consists of a number of values separated by commas. Items in lists and tuples don't all have the same type.

Creating a list is as simple as putting different comma-separated values between square brackets:

| | |
|---|---|
| **Creating a list** | >>> a = ['cat', 'dog'] |
| **Referencing a list** | >>> a[ 1]<br>'dog' |
| **Finding an item in a list** | >>> a.index('dog')<br>1 |
| **Creating a tuple** | >>> p = 1983, 'R2-D2', 'C-3PO'] |
| **Referencing a tuple** | >>> p[ 2]<br>'C-3PO' |
| **Finding an item in a tuple** | >>> p.index(1983)<br>0 |

There are certain things you can do with all sequence types. This includes: indexing, slicing, adding, multiplying, and checking for membership. In addition, Python has built-in functions for finding the length of a sequence and for finding its largest and smallest elements:

| Function | Description |
|---|---|
| >>> len(p) | Gives the total length of the list. |
| >>> max(p) | Returns item from the list with max value. |
| >>> min(p) | Returns item from the list with min value. |

## 3. FUNCTIONS

A function is a block of organized, reusable code that is used to perform one action.

While Python gives you many built-in functions like print etc. you can also create your own functions! The functions you create are called user-defined functions.

These are the rules for defining functions in Python:

1. Functions begin with the keyword def followed by the function name and parentheses:
    a. example: def function_name()
2. The input parameters should be placed within the parentheses. You can also define parameters inside these parentheses.
3. The first statement of a function can be an optional statement - the documentation string (or "docstring") of the function.
4. The code block within every function starts with a colon (:) and is indented.
5. The statement return exits a function.

| | | |
|---|---|---|
| **Defining Functions** | def function_name(input): | def circle_area(radius): |
| |     #indented code |     area = 3.14*(radius**2) |
| |     #(4 spaces or 1 tab) |     return area |
| |     return output | |
| **Calling Functions** | >>> function_name(input) | >>> circle_area(5) |
| | Output | 78.5 |
| **Input/Output** | >>> print(5+5) | >>> age = input("How old are you?") |
| | 10 | How old are you? 12 |
| | >>> print 'What is pi?' | >>> print age |
| | 'What is pi?' | 12 |
| **Formatting** | #Convert to an integer | #Convert into a string of characters. |
| | >>> int(3.14) | >>> 'I am ' + str(age) + 'years old' |
| | 3 | 'I am 12 years old' |

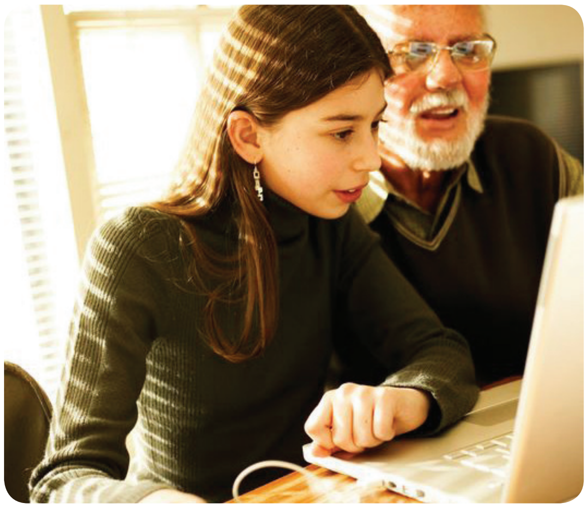
Make your own function:
Define the function as circle_area(diameter)
The parameter is (diameter)

## 4. CONTROL FLOW DOCUMENTATION

| | | |
|---|---|---|
| **While** | #initialize test variable | number = 0 |
| | while (test is true): | while (number < 5): |
| |     #run indented code |     print number |
| |     #increment test variable |     number = number + 1 |
| **For** | for variable in list: | for number in range (0,5): |
| |     print variable |     print variable |
| **If/Else** | if a < b: | if 1 < 2: |
| |     #do this indented code |     print "Yes" |
| | else: | else: |
| |     #do this indented code |     print "No" |

## 5. OTHER USEFUL MODULES

| Random Documentation | Math Documentation | Turtle Documentation |
|---|---|---|
| >>> from random import * | >>> from math import * | >>> from turtle import * |
| >>> randint(1,10) | >>> factorial(5) | >>> forward(100) #pixels |
| 5 | 120 | >>> goto(100,100) #(x,y) |
| >>> randint(1,10) | >>> log(10) #log e | >>> right(180) #angle |
| 8 | 2.30 | >>> home() #(0,0) |
| >>> choice(('H', 'T')) | >>> log(1000,10) #log 10 | |
| 'H' | 3 | |
| >>> choice(('H', 'T')) | >>> sin(pi/2) #Radians | |
| 'T' | 1.0 | |



## Reflect

- What did you learn today that you did not know before?
- Can we use conditional logic to account for several possible cases of the same problem?
- Can we use "while loops" to perform a repetitive calculation quickly and accurately?

## Apply

- How was your teamwork on this activity?
- What is a career area that relies on teamwork?
- When do you work as a team at home? At school?
- Why is teamwork so important?

## Debriefing

Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

# 4-H *Club Meeting 4*
## Functions and Algorithms in Python

### Supplies

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Most math problems can be solved by a computer in less than a second. However, some problems can take far longer, and others exist that we aren't sure how to solve at all. In this activity, youth will learn how to measure the complexity of a function, create different types of functions by using algorithms, all while learning how computer science applies to real-world situations.

In this activity you will use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

Prerequisites: Youth should know how to factor a number and how to use scientific notation to describe very small numbers.

## Getting *Started*

### 1. THE COMPLEXITY OF FUNCTIONS

- Write down a function that you might be able to solve in a minute or so by hand. Answers could include $6^2$ or $8^6$ or 182 + 347 depending on the abilities of the club. Have youth solve them and time them.
- Have youth make predictions on how much longer it would take to calculate the same function if the numbers were 10 or 100 times larger.
- Explain that the time it takes for a computer to solve a problem is affected by its complexity, similar to the way larger problems take more time to solve by hand.
- In general, multiplication is more complex than addition. Python can be used to show that this is true:

Copy and paste or enter this sample Python code for **addition**. The example output numbers may vary by machine:

```
from time import time

a = 27
b = 59

for trial in range (5):
    start = time ()
    a + b
    print time () - start
```

This is how long it takes for the computer to calculate 27 + 59 = 86, 5 times, in seconds:

```
86
0.0730860233307
86
0.0331969261169
86
0.0337069034576
86
0.0481669902802
86
0.0328640937805
```

Copy and paste or enter this sample Python code for **multiplication**. The example output numbers may vary by machine:

```
from time import time

a = 27
b = 59

for trial in range (5):
    start = time ()
    a * b
    print time () - start
```

This is how long it takes for the computer to calculate 27 * 59 = 1593, 5 times, in seconds:

```
1593
0.126674175262
1593
0.0521900653839
1593
0.0816390514374
1593
0.082288980484
1593
0.0668818950653
```

Note:

With a calculation that takes such a small fraction of a second, it is only by using a computer that we can actually see the difference between the two operations. This effect will be more obvious later for larger inputs. With the multiplication calculations taking almost 10 times longer, it might be worth pointing out the parallel to their calculations or have them imagine a process that takes 10 times longer than another process.

## QUESTIONS

1. Why did we use a loop to run this scenario multiple times?
   A: It may surprise you to discover that there are factors that can slow down a process (even one going extremely fast) so it is a good idea to run multiple trials to get a good average.

2. Run the above code again using two larger numbers for a and b, by how much did the number change?
   A: Computers are able to process quickly and have large amounts of memory. For this type of problem, you will not see a significant change in the amount of time it takes to calculate.

3. Think about the ways many people typically add and multiply. What makes multiplication the longer of the two?
   A: For 49 + 49 the process is exactly what you see. For 49 * 49, the process is 49 + 49 + 49 + 49 + 49 + 49.... The process is often shortened with algorithms like the partial product algorithm where 49 is multiplied by 9 then added to the product of 49 * 40.

## 2. CREATING FUNCTIONS WITH ALGORITHMS

- Functions take input data like numbers, words, pictures, etc., and return an output. Even if you are not aware of it, functions are doing some kind of calculation with the data.

- Algorithms are the instructions, steps, or recipe to get from the inputs to the output. Some examples of functions they use every day:
  - It's hot outside today so I will wear shorts and sandals.
  - The drink costs $1.25 dollars and I have 5 quarters.
  - I have a research report due in a month so I need to spend 45 minutes each night for 1 month to finish it in time.
  - The board in class is fuzzy, I'm going to put on my glasses.

- Write out the steps to make a grilled cheese sandwich so someone who has never made one before would know what to do.

- Functions can be used over and over again. By learning a few basic types of functions, it is possible to perform a wide variety of algorithms and simulations.

The simplest functions are those that take an input, perform a calculation and return an output. These functions only run one time.

```
Lines that start with a # are comments and are ignored by Python.
>>> 2 + 5
7
>>> r = input ("What is the radius of your circle? ")
What is the radius of your circle? 10
>>> a = 3.14*(r**2) #pi times r squared
>>> print "Area = " + str(a)
62.8
>>> name = input ("What is your name? ")
What is your name? Arthur
>>> print "Hello " + name
Hello Arthur
```

## QUESTIONS

1. There are 5 functions in the above examples. Can you find them all?
   A: Addition "+" (of numbers and strings), input (), multiplication "*", print, and str ().

2. Why did we need to convert a into a string of characters in print "Area = " + str (a)?
   A: The number a and the words Area are not the same type and needed to be converted to show on the same line.

3. Rewrite the third example to calculate circumference instead.
   A: c = 3.14*2*r and print "Area = " + str (c).

## 3. FUNCTIONS THAT REPEAT OR LOOP

- Functions are helpful for making decisions or calculations. How do you deal with activities or chores in your life that repeat? What are examples of functions that repeat?
    - Alarm clocks or calendar reminders
    - Music (singing the chorus over and over)
    - A car's combustion cycle (intake, compression, combustion, exhaust)

- In the above three examples there are three types of repeating functions. They are:
    - Loops forever (every day or every year)
    - Loops a certain number of times (sing the chorus 3 times)
    - Loops while something is true (while there is gasoline and the car is on)

**Note:**

With loops, after the colon ":" the next line requires four spaces of indenting to show the code which is inside the loop and the code which is not. Other languages use brackets { } to do this. In IDLE, Python will automatically space the code as long as the colon is used.

Enter this function of a loop that continues forever (infinite loop):

```
>>> count = 0
>>> while 1 == 1: #The loop will repeat forever until you restart Python.
      count += 1 #Adds 1 to count each time it loops
            print count
1
2
3
... #and so on forever
```

## QUESTIONS

1. What are other ways to create an infinite loop with the while loop?
   A: while 3 == 3, while 1, while True, etc. As long as the test is always true.

2. Why does count = 0 use only one equals sign but while 1 == 1: uses two?
   A: The first is storing zero in the variable count, the second is a test to see if one is equal to one.

3. What is another way count += 1 could be written?
   A: count = count + 1

Enter this function of a loop that only repeats a few times:

```
>>> count = 3
>>> while count > 0: #Checks each time to see if count is greater than zero.
          print count
          count -= 1 #Subtracts 1 from count each time it loops
3
2
1
>>> for count in range (1, 4): #A quicker way to do the same thing as above
         print count
```

## QUESTIONS

1. In the second version of the loop above, why is the range from 1 to 6 when the loop prints out the numbers 1,2,3,4?
   A: The function is range (start,end) including start but excluding end.

2. For the second loop, change it to for count in range (1,6,2): what does the third number do? What would it do if it was changed to 3?
   A: The third number is the step or how many numbers to skip each time. If it was changed to 3 then the output would be 1 and 4.

## 4. FUNCTIONS THAT USE LOGIC FOR DECISION MAKING

1. The functions so far use calculations. Functions can also use logic to make decisions and take different actions \ based on these decisions. When you combine repeating algorithms and logic, there is enough intelligence to make decisions and that is the power of computational thinking.
2. Humans use their own logic in the functions they use everyday:
    a. Which shirt matches my socks?
    b. How much sugar and cream should I add to my coffee to make it perfect?
    c. What is the fastest way to school while staying dry in the rain?

Enter this function of logic that determines which name is first in alphabetical order:

```
# are comments and are ignored
>>> name1 = input ("What is the first name? ")
What is the first name? "Alice"
>>> name2 = input ("What is the second name? ")
What is the second name? "Bob"
>>> if name1 < name2:
        print name1
else:
            print name2
Alice
```

Enter this function of a Guess The Number Game (you may wish to save the code so you can play it again):

```
>>> from random import *
>>> correct_number = randint (1,10) #Computer picks a number from 1 to 10
>>> guess = 0 #This is set to zero so the loop begins.
>>> while guess != correct_number:
        guess = input ("What number am I thinking of (between 1-10)? ")
         if guess > correct_number:
            print "Too high"
         if guess < correct_number:
            print "Too low"
         if guess == correct_number:
            print "You got it!"
```

Example output:

```
What number am I thinking of (between 1-10)? 5
Too High
What number am I thinking of (between 1-10)? 3
Too Low
What number am I thinking of (between 1-10)? 4
You Got it!
```

Experiment with this function of a loop that acts as a thermostat:

```
>>> set_temp = 70
>>> current_temp = 70
>>> while 1 == 1:
      set_temp = input (“What do you want the temperature to be? “)
      if current_temp > set_temp:
         print “Turning on the air conditioning!”
         while current_temp > set_temp:
            current_temp -= 1
            print “Current Temp: “ + str(current_temp)
      if current_temp < set_temp:
         print “Turning on the heater!”
         while current_temp < set_temp:
            current_temp += 1
            print “Current Temp: “ + str(current_temp)
```

## QUESTIONS

1. Why does the first loop put the inputs in alphabetical order with the < sign? Isn't < only for numbers?
   A: Operator overloading is when the < sign can compare 1 < 2 and 'a' < 'b'. Another example of operator overloading is 2 + 3 and 'Py' + 'thon'.

2. What would happen if you ran the "Guess the Number" game again but skipped the second and third lines?
   A: The game would not pick a new random number, guess would not be reset to 0, therefore the condition to run the loop would not be met.

3. Instead of having the infinite loop in the beginning of the code, how could you modify the code to only run once?
   A: There are many possible options but the easiest one would be to modify the first loop to say while current_temp != set_temp.

4. Imagine you had a more powerful air conditioning or heater that added two degrees each time through the loop instead one. What might be a possible issue?
   A: If nothing else was changed in the code, you might overshoot the set_temp. This already happens in your homes but negative feedback loops and sensors work to prevent this.

## Reflect

- What is a function? (**A:** A function performs calculations or follows a set of instructions.)
- Fill in the blank: Functions take input _____ like numbers, words, pictures, etc., and return an _____. (**A:** data, output)
- What is an algorithm? The instructions or logic that convert a function's inputs into outputs.
- What is Boolean Logic? (**A:** Compares two objects and determines whether the statement is true or false. Some examples: <,>, ==, !=.)
- What is the Python Interpreter/Shell? (**A:** In this mode, Python handles one line of instruction at a time as opposed to creating a multiline program and running it all as one.)
- How will you know you are in the Python Interpreter/Shell? (**A:** You will know you are in Python's interpreter mode if you see >>>.)

## Apply

- How can you create algorithms for everyday situations?
- Why might it be useful to know how to create functions and algorithms when working with a computer? A: Computers repeat instructions and calculations much more quickly than we can.
- Is y=3x+2 a mathematical function? Why is it useful to be able to identify a function?
- Why should you save a record of all the code you write?
- Do you think you could modify your algorithms to adapt to new situations?

## Debriefing

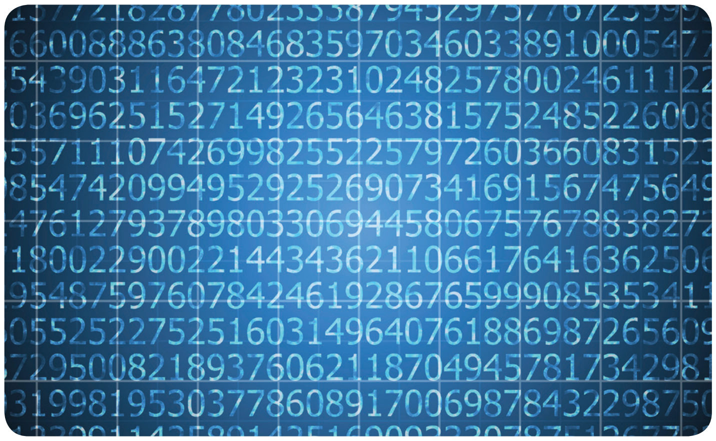
Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

## 4-H *Club Meeting 5*
## Mathematical Analyzation in Python



**Supplies**

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Analytical skills are critical to solving real-world problems with creativity and innovation. Working through each of these activities will help your club members think computationally while gaining a stronger understanding in using a programming language to problem solve.

In this activity you will use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

## Getting *Started*

### 1. COMPLEMENTARY AND SUPPLEMENTARY ANGLES

- Download Python Complement Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/complement.py.zip
- Open and save the complement.py file from the IDLE Shell.
- Run the module (click F5).

Use this program to apply your club member's knowledge of geometry and automatically compute the complement and supplement of a given angle. Have youth analyze and edit the program to reinforce their understanding.

### 2. LONG MULTIPLICATION

- Download the Python Long Multiplication Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/long-mult.py.zip
- Open and save the long-mult.py file from the IDLE Shell.
- Run the module (click F5).

Use this program to apply your club member's knowledge of how to calculate a long multiplication problem on two-digit numbers, for example, 49 x 87. Have youth analyze and edit the program or use the program to check results of other long multiplication problems.

### 3. PROPORTIONALITY

- Download the Python Proportionality Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/proportionality.py.zip
- Open and save the proportionality.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to help your club members check whether or not two fractions are proportional. Have youth analyze, edit, or use the program to check the proportionality when comparing other fractions.

### 4. EVALUATING EXPRESSIONS

- Download the Python Evaluating Expressions Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/evaluate-expression.py.zip
- Open and save the evaluate-expression.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to explain to your club members how a basic calculator functions. Have youth analyze, edit, or use the program to check results to other math problems. This program will introduce youth to Python's eval function as a means of computing expressions containing variables a, b, and c while providing values for each.

### 5. RATIOS

- Download the Python Ratios Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/ratios.py.zip
- Open and save the ratios.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to help your club members understand ratios by solving for x in the equation a/b = c/d, where x can be in any location in the two fractions. Have youth analyze, edit, or use the program to check solutions of other calculations.

### 6. PYTHAGOREAN THEOREM

- Download the Python Pythagorean Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/pythagorean.py.zip
- Open and save the pythagorean.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to apply your club member's knowledge of the Pythagorean Theorem to calculate a third side of a right triangle given the other two sides. Have youth analyze, edit, or use the program to check solutions of other calculations.

### 7. SURFACE AREA

- Download the Python Surface Area Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/surfacearea.py.zip
- Open and save the surfacearea.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to test your club member's knowledge of surface area formulas to automatically calculate the surface areas of the following geometric objects: cube, rectangular prism, cylinder, and sphere. Have students analyze, edit, or use the program to check results of other calculations.

## Reflect

- What did you learn today that you did not know before?
- Can we use these programs to solve other problems outside of school and 4-H?
- How did you edit these programs?

## Apply

- How did you work together as a team on this activity?
- What is a career area that relies on teamwork?
- When do you work as a team at home or school to solve problems?
- How can you use Python in your daily life to analyze and solve problems?

## Debriefing

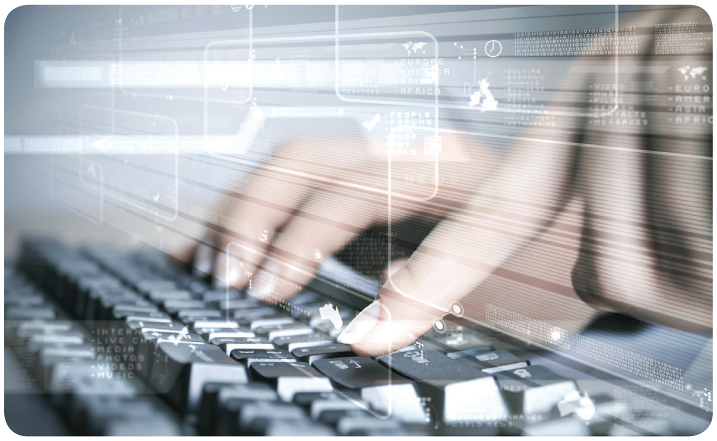
Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

# 4-H *Club Meeting 6*
## Problem Solving in Python



**Supplies**

- Computers (PC or Mac, Laptop or Desktop
- Python 2.7 Installed

Problem solving skills are critical to scientific breakthroughs and innovation. Working through each of these activities will help your club members think computationally while gaining a stronger understanding of using a programming language to solve real-world problems.

In this activity you will use the Python editor to write multi-line programs:

1. To write code: From the IDLE Shell: **Ctrl-N/Command-N** or **File** → **New Window**
2. To run code: Press **F5** or choose **Run** → **Run Module** → **Save with .py extension**

## Getting *Started*

### 1. HOW TALL IS THE CLIFF?

- Download the Python Similarity-Cliff Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/similarity-cliff.py.zip
- Open and save the similarity-cliff.py file from the IDLE Shell.
- Run the the module (click F5).

A rock climber wants to know the height of a cliff. The climber measures the shadow of her friend, who is 5 feet tall and standing beside the cliff, and measures the shadow of the cliff. If the friend's shadow is 4 feet long and the cliff's shadow is 60 feet long, how tall is the cliff?

### 2. HOW MUCH LEMONADE?

- Download the Python WordProblem-Lemonade Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/wordproblem-lemonade.py.zip
- Open and save the wordproblem-lemonade.py file from the IDLE Shell.
- Run the the module (click F5).

Sam has a jar with 5 cups of fresh lemonade. Jack has some glasses which hold 1.5 cups each of liquid. How many glasses of lemonade can Jack serve of Sam's lemonade?

## 3. HOW LONG TO REACH A TARGET POPULATION?

- Download the Python Birth-DeathRate Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/birth-deathrate.py.zip
- Open and save the birth-deathrate.py file from the IDLE Shell.
- Run the the module (click F5).

How long will it take to reach a certain target population, given a starting population, birthrate, and death rate? Have your club members analyze, edit, or use the program to check results of different inputs.

## 4. HOW FAST CAN TWO WORKERS FINISH THE JOB?

- View the Python WorkRate-2People Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/workrate-2people.py.zip
- Open and save the workrate-2people.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to help club members conceptualize work problems by solving word problems with two people working together at different rates. Have students analyze, fill in parts of, or enhance the program to solve more technical work problems.

## 5. HOW FAST CAN THREE WORKERS FINISH THE JOB?

- View the Python WorkRate-3People Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/workrate-3people.py.zip
- Open and save the workrate-3people.py file from the IDLE Shell.
- Run the the module (click F5).

Use this program to help club members conceptualize work problems by solving word problems with three people working together at different rates. Have students analyze, edit, or enhance the program to solve more technical work problems.

## 6. HOW TALL IS THE BUILDING?

- View the Python Similarity-Basketball Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/similarity-basketball.py.zip
- Open and save the similarity-basketball.py file from the IDLE Shell.
- Run the the module (click F5).

A basketball rim 10 ft high casts a shadow 15 ft long. At the same time, a nearby building casts a shadow that is 54 ft long. How tall is the building?

## 7. WHAT IS THE WAITING TIME IN LINE?

- View the Python WordProblem-Line Program File (zip).
  http://www.google.com/edu/computational-thinking/programs/wordproblem-line.py.zip
- Open and save the wordproblem-line.py file from the IDLE Shell.
- Run the the module (click F5).

There are 90 people in line at a theme park ride. Every 5 minutes, 40 people get on the ride and 63 join the line. Estimate how long it would take for 600 people to be in line.

## Reflect

- What did you learn today that you did not know before?
- Did you find the solution to the problems right away?
- What ways did you use Python to solve these problems?

## Apply

- How did you interact with others during this activity?
- Who demonstrated teamwork?
- Who asked for help? Who gave help?
- What kind of problems do you solve at home? At school?
- Why is teamwork so important?

## Debriefing

Explain that being a part of a code club requires a great deal of responsibility. Not only is a computer very expensive, but there are many files of code that each youth must keep organized in folders. Allow time for questions and answers. Mention that proper file management is necessary for the code club to function properly and to save programs from previous club meetings. Instruct the youth to become familiar with the 4-H portfolio by going to www.utah4h.org. At this point, youth should also know that they can print off and enter their programs at their local County Fair.

## References

Exploring Computational Thinking. (2013). Retrieved from http://www.google.com/edu/computational-thinking/

Python Programming Language - Official Site. (2013). Retrieved from http://www.python.org

## More to Discover

Congratulations on completing your Discover 4-H club meetings! Continue with additional curriculum in your current project area, or discover other 4-H project areas. Check out the following links for additional 4-H curriculum.

1. http://utah4h.org/htm/discover4hclubs
2. http://www.4-h.org/resource-library/curriculum/
3. http://utah4h.org/htm/resource-library/view-all-curriculum

### Become a 4-H Member or Volunteer

To **register** your Utah club or individuals in your club visit:

http://www.utah-4.org/htm/staff-resources/4-h-online-support

http://utah4h.org/htm/about-4-h/newto4h/

Non-Utah residents please contact your local 4-H office:

http://www.4-h.org/get-involved/find-4-h-clubs-camps-programs/

## Stay Connected

### Visit Your County Extension Office

Stay connected with 4-H activities and news through your county Extension office. Ask about volunteer opportunities and don't forget to register for your county newsletter. Find contact information for counties in Utah here:

http://extension.usu.edu/htm/counties

### Enjoy the Fair!

Enter your project or create a new project for the county fair. Learn about your county fair and fair judging here:

http://utah4h.org/htm/events-registration/county-fairs

## Participate in Local or State 4-H Activities, Programs, Contests or Camps

For Utah state events and programs visit:

http://utah4h.org/htm/events-registration

http://www.utah4h.org/htm/featured-programs

For local Utah 4-H events and programs, visit your county Extension office.

http://extension.usu.edu/htm/counties

Non-Utah residents, please contact your local 4-H office.

http://www.4-h.org/get-involved/find-4-h-clubs-camps-programs/

## Discover *Service*

### Become a 4-H Volunteer!

http://www.youtube.com/watch?v=UBemO5VSyK0

http://www.youtube.com/watch?v=U8n4o9gHvAA

To become a 4-H volunteer in Utah, visit us at:

http://utah4h.org/htm/about-4-h/newto4h/

### Serve Together as a 4-H Club or as an Individual 4-H Member

Use your skills, passions, and 4-H to better your community and world. You are needed! Look for opportunities to help in your area or participate in service programs that reach places throughout the world (religious groups, Red Cross, etc.).

### Hold a Club Service Project

USU Collegiate 4-H Club hosted "The Gift of Giving" as a club activity. Club members assembled Christmas stockings filled with needed items for CAPSA (Community Abuse Prevention Services Agency).

http://tinyurl.com/lu5n2nc

## Donate 4-H Projects

Look for hospitals, nursing homes, or other nonprofit organizations that will benefit from 4-H projects. Such projects include making quilts for CAPSA or Primary Children's Hospital, or making beanies for newborns. During Utah 4-H State Contests, 40 "smile bags" were sewn and donated to Operation Smile.

## Partner with Local Businesses

92,000 pounds of processed lamb, beef, and pork were donated to the Utah Food Bank in 2013 by multiple companies.

http://tinyurl.com/pu7lxyw

## Donate Money

Clubs or individuals can donate money gained from a 4-H project to a worthy cause. A nine-year-old 4-H member from Davis County donated her project money to help a three-year-old battle cancer.

http://tinyurl.com/mqtfwxo

## Give Us Your *Feedback*

Help us improve Discover 4-H curriculum. We would love feedback or suggestions on this guide; please go to the following link to take a short survey:

http://tinyurl.com/lb9tnad