Python Programming for Kids

(Ages 10-16)

Organized by Devopedia. Visit https://devopedia.org

0 **Installation**

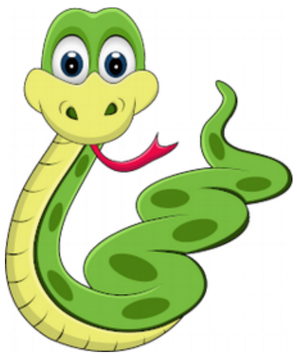
Install Python 3.5 or later from https://www.python.org/download/. This installation comes with pip, a tool to install additional Python packages. You can upgrade pip if required:

- Windows: `python -m pip install -U pip`
- Linux/Mac: `pip install -U pip`

The code for this workshop is at https://bit.ly/ppkidscode. Download the file. Unzip it into a folder. Let's call this folder ppforkids.

When working with multiple projects, managing different versions of packages can be difficult. The recommended procedure is to create a virtual environment for each project. Let's install virtualenv and create a virtual environment:

- Windows:
  ```
  pip install virtualenv
  cd ppforkids
  virtualenv --python=python.exe venv
  venv\Scripts\activate
  ```
- Linux:
  ```
  sudo pip install virtualenv
  cd ppforkids
  virtualenv --python=python venv
  source venv/bin/activate
  ```

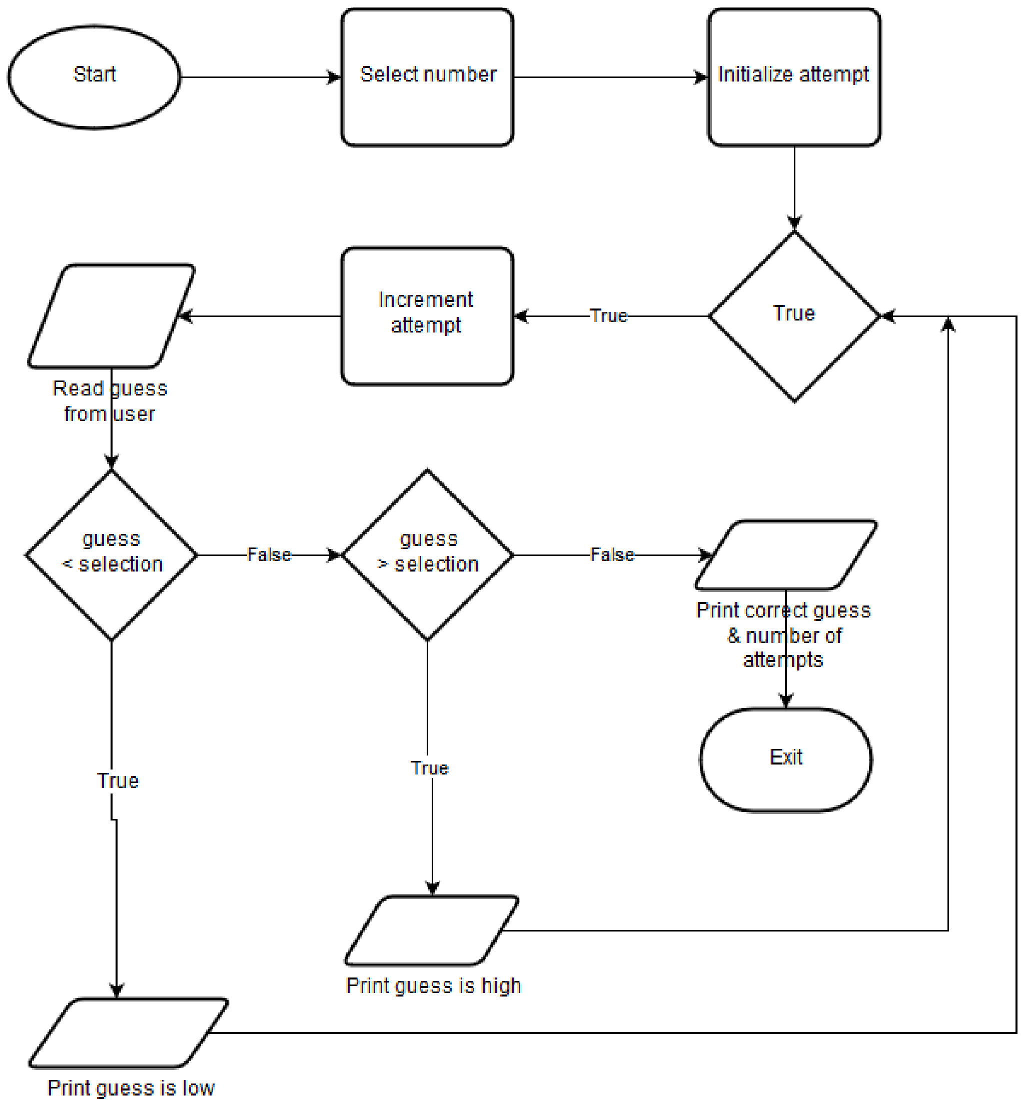
	Finally, install all dependent packages (within your virtual environment) for this workshop: `pip install -r requirements.txt` For editing and saving code, we recommend you to install Visual Studio Code.
1	**Basics**  Let's discuss the following: a) How computers help us in everyday life? b) Do we need to program computers and why? c) How do computers work? d) What is the Python programming language?
2	**Constants and Variables** Karan was born on January 2012. His sister Priya was born on February 2016. a) How old are Karan and Priya in May 2019? How old will they be in December 2030? b) What's the age difference between Karan and Priya? c) Let's model these as equations: Age(Karan) = Current Year – 2012 Age(Priya) = Current Year – 2016 Age Difference = Age(Karan) – Age(Priya) = (Current Year – 2012) - (Current Year – 2016) = 2016 – 2012 = 4 d) Let's write some Python code to compute the above. Which are the constants and which are the variables?

3	Temperature can be expressed in Celsius or Fahrenheit scales. Given temperature in Celsius **C**, we can derive the Fahrenheit value **F** using the equation **F =** $(\mathbf{C} \times 9/5) + 32$ a) Write a program that will take a Celsius value and give us the Fahrenheit value. Which are the constants and variables in your program? b) Check that your program works correctly for the following: 0 C = 32 F (melting point of water) 37 C = 98.6 F (temperature of human body) 100 C = 212 F (boiling point of water) -60 C = -76 F (coldest on Mt. Everest) c) Can we change the program to take Fahrenheit value and give us Celsius value?
4	**If-Else** Execute the program guess-the-number.py. The program will select a number between 1 and 50. Your job is to guess the correct number. If your guess is wrong, the program will tell you if your guess is lower or higher than the correct answer. a) How many guesses do you need to guess the number correctly? b) Is there a way to guess in fewer attempts? Study the program and learn the following: a) while-condition: implements a loop b) if-elif-else: test for conditions and branch: we've used < c) += to modify values in variables d) exit(0) to exit the program. What happens if we remove this? e) Explain what is **Binary Search**?

5

Flowchart

Flowcharts help us to visualize the different steps of a program. It's a useful tool for beginners. Here's a flowchart for the guess-the-number program.



Modify the flowchart (on pen and paper) so that we first check if guess equals selected number. Change your program based on the modified flowchart.

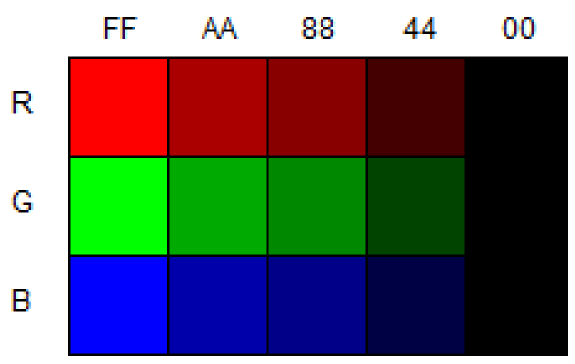
6	**Conditionals** There are many things we do in life that are dependent on conditions. For example, we switch on a light if it's dark. In this example, if it's dark is the condition and switch on a light is the action. Sometimes there could be multiple conditions. We switch on a light if it's dark **AND** we don't want to sleep. This is an example of two conditions that should both be True. We want to switch off the light if it's not dark **OR** we're going to sleep. This is an example of two conditions but only one of them needs to be True. Try running the program light_control_1.py. Note the following facts: a) True and True == True b) False and True == False c) True or True == True d) False or True == True e) False or False == False f) False and False == False
7	**Function** Run the code light_control_2.py. See how the conditions have changed. Variables is_dark and is_sleepy now store True or False instead of 'y' or 'n'. Run the code light_control_3.py. Notice how the code is shorter than light_control_2.py. How did we do this?

8	**List**
	
	So far we were working with numbers and strings. List is useful because we can store many values within a single variable. Let's make the following lists:
	a) `small_nums = [1, 4, 6, 9]`
	b) `words = ['go', 'come', 'in', 'out', 'today']`
	c) `nums_from_zero = list(range(10))`
	d) `nums_from_one = list(range(1, 11))`
	
	Try the following:
	a) Print the list: `print(words)`
	b) Print the items one by one:
	`for word in words:`
	`    print(word)`

9	**For Loop**
	
	On pen and paper, find the sum of all even numbers from 20 to 50. How long did it take you to find the answer?
	
	Let's now write a program so that the computer can give us the answer quickly:
	`total = 0`
	`for i in range(20, 51, 2):`
	`    total += i`
	`print(total)`
	
	Change the program to try the following:
	a) Find the sum of all 4-digit integers.
	b) Find the sum of every third integer from 100 to 200.
	c) Find the sum of all integers from -200 to 200.

10 **Colors**

Let’s learn about colours. Any colour can be expressed as a combination of three colours: Red, Green, Blue. By mixing these three colours in different proportions, we get different colours.



0xFF = 255
0xAA = 170
0x88 = 136
0x44 = 68
0x00 = 0

Each colour component R, G or B value is in the range 0 to 255. These are decimal values. Computers instead commonly use hexadecimal values. Hexadecimal system uses base 16: 0, 1, ... , 9, A, B, C, D, E, F. Thus, it has 16 characters.

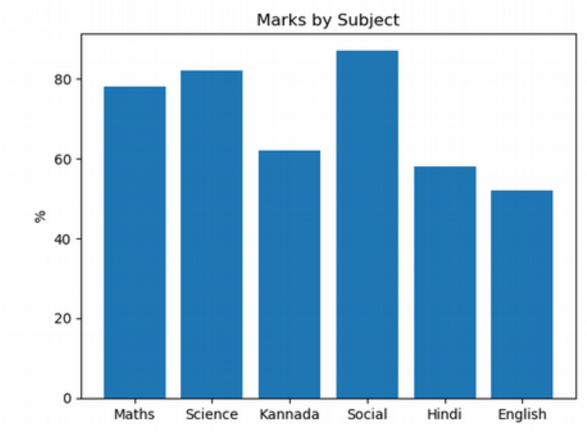
Python can help us convert:

a) Decimal to Hexadecimal: hex(170)

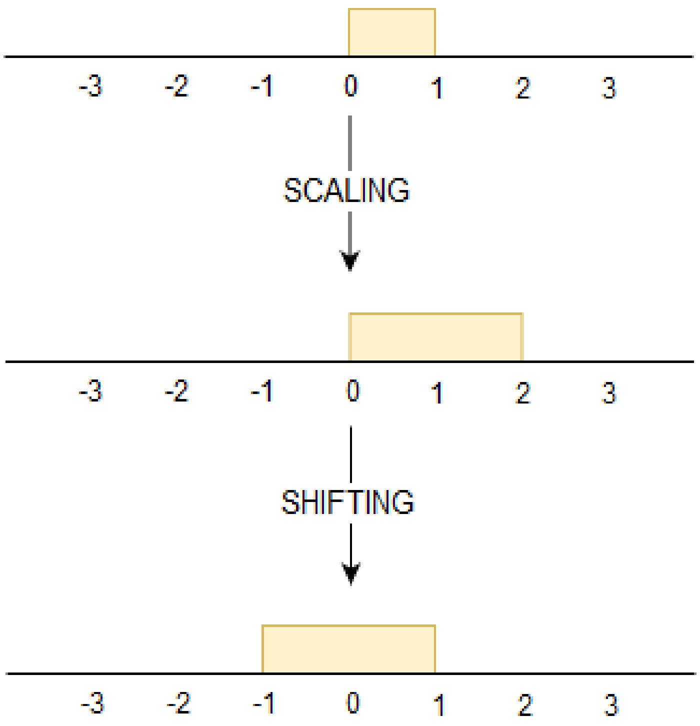
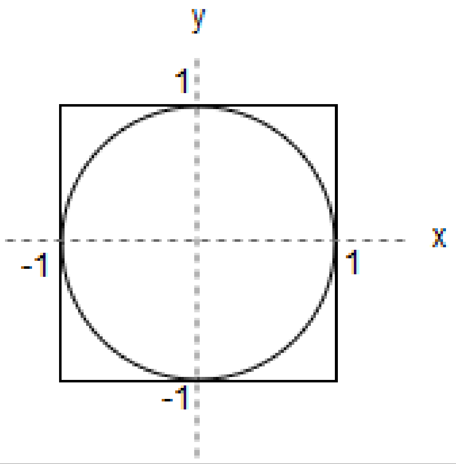
b) Hexadecimal to decimal: int('0xAA', 16)

11 **Make a Graph**

We usually find it hard to process numbers. Numbers are easier to “see” if we draw a graph. Graph helps us see relationships and patterns more easily.



Run the program marks_barchart.py. What can you make out from the chart?

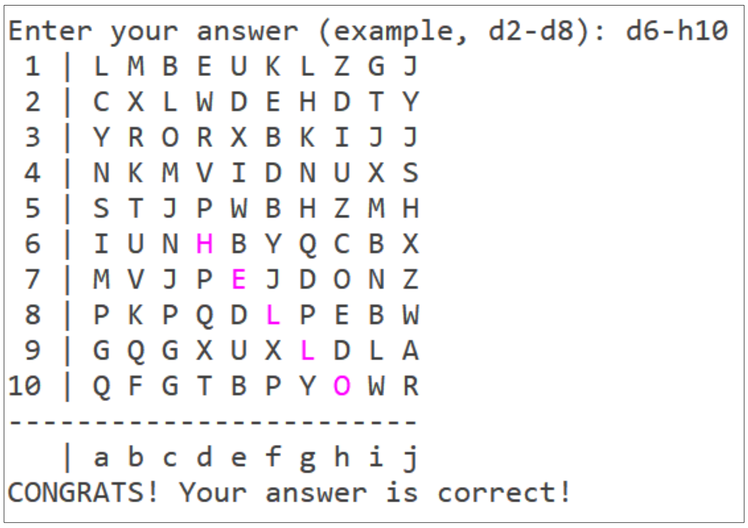
	a) Add color=(0.1, 0.2, 0.8, 1) to plt.bar()? What happens if 1 is changed to 0.4? b) Color can also be in hexadecimal form. Try this: color='#ff3d3d77' c) Change bar to barh. Change ylabel to xlabel. What happens? d) Change the style by adding mpl.style.use('seaborn') Execute marks_barchart_2.py. Try to understand the code. For more information, visit https://matplotlib.org
12	**Shifting and Scaling**  -3 -2 -1 0 1 2 3 SCALING -3 -2 -1 0 1 2 3 SHIFTING -3 -2 -1 0 1 2 3 Let’s assume we have numbers in the range 0 to 1. We can **multiply or divide** these numbers to get a new range. This is called **scaling**. We can **add or subtract** to get another range. This is called **shifting**.  We’ll use these ideas in calculating the value of pi or π. Run the program find_pi.py. Note how scaling and shifting are done in the code.

How does this program find the value of π? Assume we throw darts at the square. Assume that all darts will land within the square. Some darts will land inside the circle. By counting how many darts are inside the circle we can get π.

In the figure, area of circle is $\pi r^2 = \pi*1*1 = \pi$. Area of square is 2*2 = 4.
Darts inside circle / Total darts thrown
= area of circle / area of square
= π / 4
Thus, π = (Darts inside circle / Total darts thrown) * 4
        = (inside_count / count) * 4

Computer scientists call this type of approach to problem solving as **Monte Carlo Simulation**.

13 **Hidden Word**

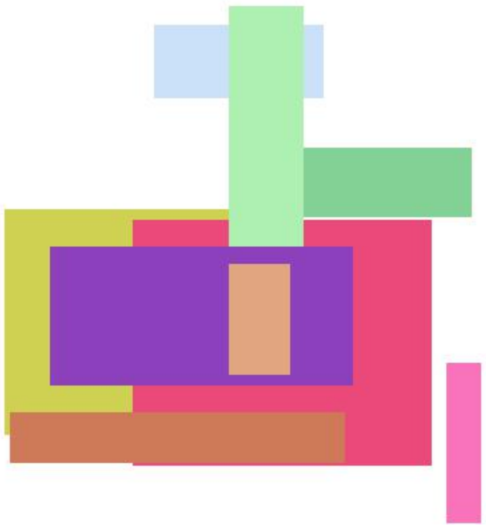


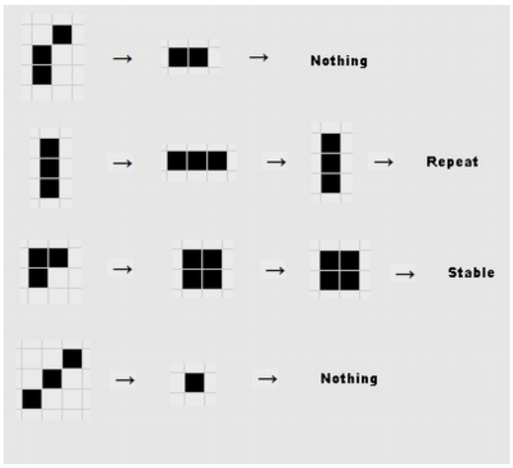
```
Enter your answer (example, d2-d8): d6-h10
 1 | L M B E U K L Z G J
 2 | C X L W D E H D T Y
 3 | Y R O R X B K I J J
 4 | N K M V I D N U X S
 5 | S T J P W B H Z M H
 6 | I U N H B Y Q C B X
 7 | M V J P E J D O N Z
 8 | P K P Q D L P E B W
 9 | G Q G X U X L D L A
10 | Q F G T B P Y O W R
------------------------
   | a b c d e f g h i j
CONGRATS! Your answer is correct!
```

Let's write a program that inserts a word into a grid of random letters. Your job is to find this word quickly. Run the program hidden_word.py

Try the following:

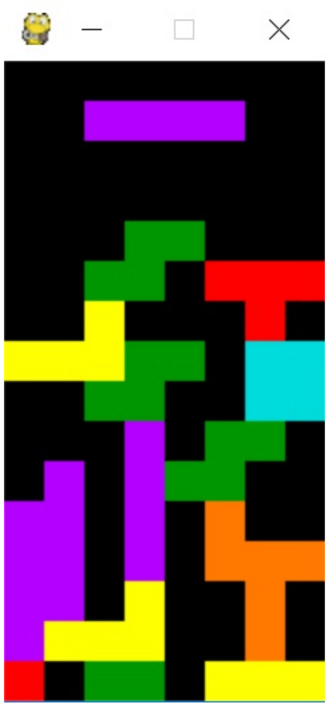
a) Use sys.argv to pass the word from command line.
b) Modify the program so that player is allowed 3 tries?
c) Print how long player took to solve the puzzle.
d) What happens when you remove random.seed in the code?
e) Discuss how you can improve this program.
f) Understand the use of class in the code.

14	**Create Art**  Computer programs can be used to create art. This example will help you learn the basics. You can use this knowledge to create more imaginative art. Run draw_art_random.py. Run this 9 times to generate 9 different images. For each run, change the configuration slightly: a) Change the count to a value between 50 and 1000. b) Generate using random colours. c) Generate using a single fixed colour but configure to any colour of your choice. d) Change the shape to line, rectangle, arc or pieslice. e) Use any combination of the above. Next, we'll combine the 9 images generated above to make a single image. Run make_collage.py to do this. A more complex example for your study is the code draw_art_pattern.py.
15	**Hungry Baby** A hungry baby comes to you for food. This baby doesn't drink milk. It eats only pancakes and pizzas! Your job is to make the baby's favourite foods quickly. The more food you

	feed the baby, the more coins you get. If the baby doesn’t get food fast enough, it cries and you lose coins. In this game, you produce food for the baby and put it on a plate. The baby consumes food from the plate. Computer scientists call this type of problem the **Producer-Consumer Problem.** How many items can you feed the baby? Run the program hungry_baby.py to find out. Answer the following questions: a) How many coins do you have when the game starts? b) How does the baby choose what to eat next? c) Once fed, how long does the baby wait before crying again? d) When the baby is fed, how many coins do you earn? e) When or how does this game end?
16	**Conway’s Game of Life**  In this game, a cell evolves based on the neighbouring cells. If a cell has < 2 or > 3 active cells next to it, it dies. If a cell has 2 or 3 active cells next to it, it lives. If an empty space is surrounded by 3 cells, a new cell is born is that space. Run the program conway_game_of_life.py to see how the game proceeds. Study the code and see if you can explain it. Can you apply this game to real life?

17

Tetris



This is a famous game created in 1984 by Alexey Pajitnov in Moscow, Soviet Union. Our code of this game uses PyGame, a useful Python package for creating games. While so far we’ve been creating applications on **Command Line Interface (CLI)**, this is the first application we’ll create with a **Graphical User Interface (GUI)**.

Run the program tetris.py. Play this game and understand how it works. Try the following in code:

a) Change the background colour to white. Change the colour of the pieces.
b) Add extra rows to the puzzle.
c) Make the game run faster.
d) What do you understand by the term FPS? What happens if maxfps is set to a low value?
e) Keep track of how many rows you have completed successfully. Display this when the game ends. Change the code to display this score during the game.
f) You learned about shifting and scaling earlier. In Tetris game, study the code to learn how shapes are rotated.

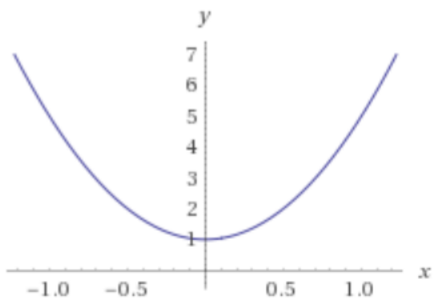
PyGame documentation is available at https://www.pygame.org/docs/

18 **Worksheet for Students**

To continue learning, try the following:

a) Take any formula from your Mathematics or Physics textbook and implement them as Python code. For example, take the equation of a parabola: y = $4x^2 + 1$. For a range of values of x, print the values of x and y.

b) Plot the above equation as a line graph using Matplotlib. Note that to get a smooth curve, x should not be limited to integers. Use this as reference: https://matplotlib.org/users/pyplot_tutorial.html



c) Take your school mark sheet for last year. Use Matplotlib to plot a suitable graph of the marks.

d) Look at this sequence of squares: 1, 4, 9, 16, 25, 36... Notice that this sequence is actually squares of integers: 1^2, 2^2, 3^2, 4^2, 5^2, 6^2... Write code to find the sum of squares of first 50 integers. To square an integer x in Python, we can write x*x or x**2.

e) Hidden word: recall that we wrote this code where a single word is hidden in the puzzle. Can you update the code so that three words of your choice can be hidden in the same puzzle? Can you update the code so that words can also go bottom-to-top or right-to-left order.

f) The Tetris code was written using functions. Can you rewrite the code using classes and objects? Study other examples (Hidden Word or Hungry Baby) to learn how to write code using classes and objects. Computer scientists call this **object-oriented programming**.

g) On paper, create simple art using only geometric shapes (square, rectangle, circle and ellipse). Now attempt to draw the same by writing a Python program.

h) Solving a maze is an interesting problem for programmers. Study the code at https://gist.github.com/a613/49d65dc30e98c165d567 and see if you can improve it. Or you can try to implement it in PyGame.