EuroPython 2021

Creating the Next Generation of Billionaires - Part 3

Python Programming for Kids, Mums',Dads', Grandpas' Grandmas', etc.

C.L.Nandi (Dr)

July 29th 2021

Who am I?

PhD, Member of BCS.

Interested in developing appropriate teaching methods for Programming in Python for School Children and encouraging them to take it up as a hobby like music.

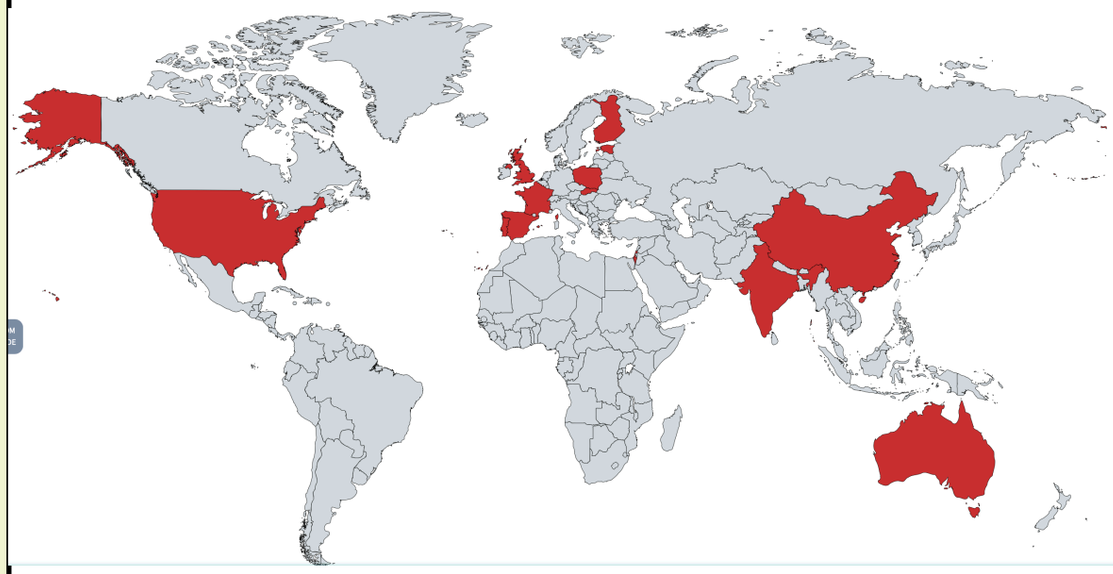
Building up a worthwhile Computer Science department in Schools

Interested Child

Governments Worldwide have dubbed:-

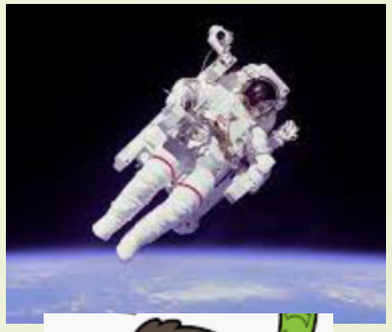
Computer pRogramming is the 4th ‘R’ along with Reading,wRiting and aRithmetic.

Introducing to Children Worldwide from Kindergarten



https://bulldogjob.com/news/82-how-computer-science-classes-are-conducted-around-the-world-5-key-conclusions

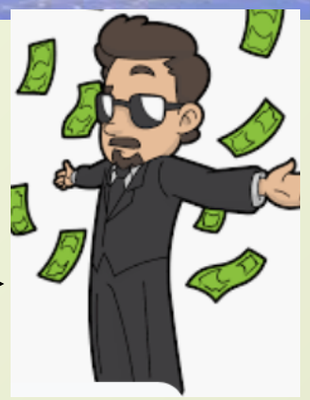
https://mapchart.net/world.html

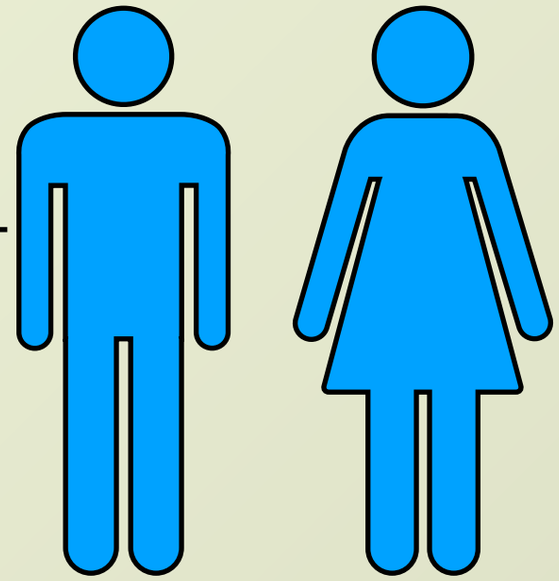


Me as Well!!

Interested Child

Curious Grown-Ups!!

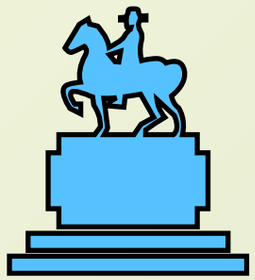




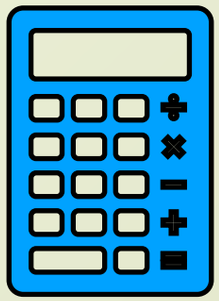
"The Subject is so young that teachers and curriculum designers have little pedagogical research to guide them".

Economist

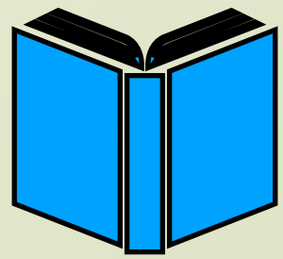
History



Maths



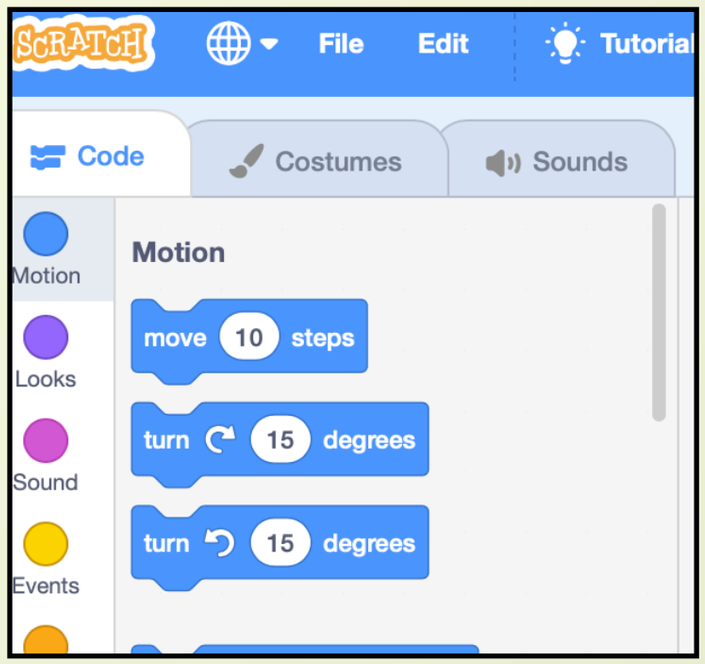
English



I developed my own framework

The Great Debate

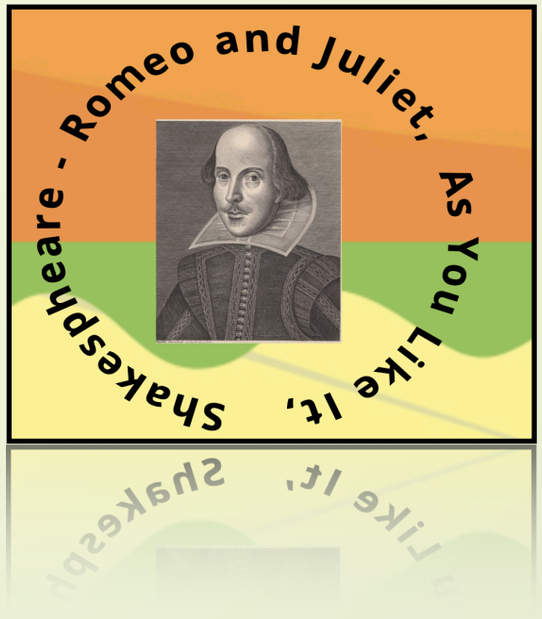
To Teach a Block-Based Language First



To Teach a Textual Language such as Python First

Introduce Textual Programming Languages such as Python from the Very Beginning & as Early as Possible (as Opposed to Block-based Languages such as Scratch, etc.)

English



Maths

$$x = y + 3$$

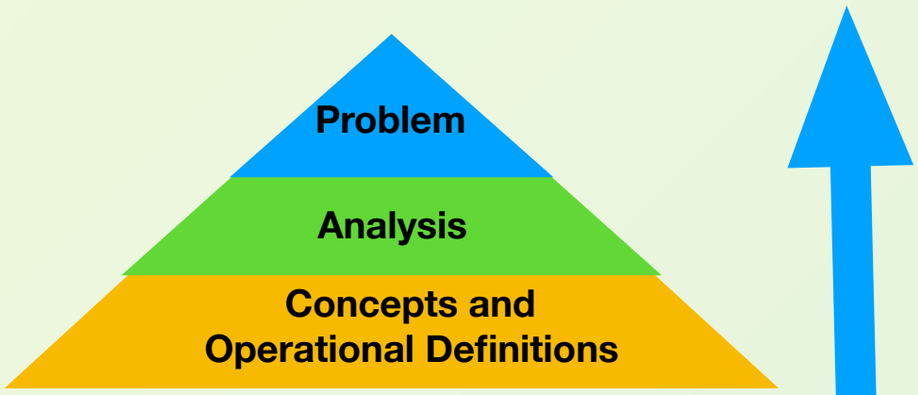
Geography

Discuss the advantages and disadvantages of Brexit? (8 marks)

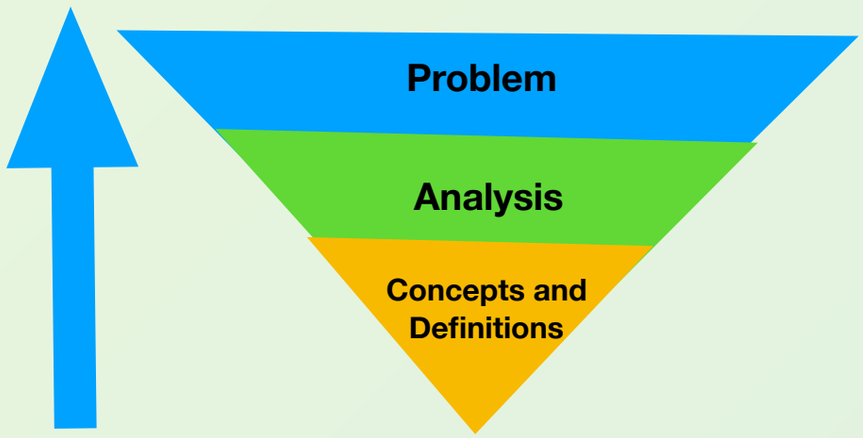
Children are accustomed to processing complex textual data in a proficient manner.

Adopt a Bottom-Up Approach (as opposed to the ever popular Top-Down Approach Method of Teaching)

Bottom-Up Approach



Top-Down Approach



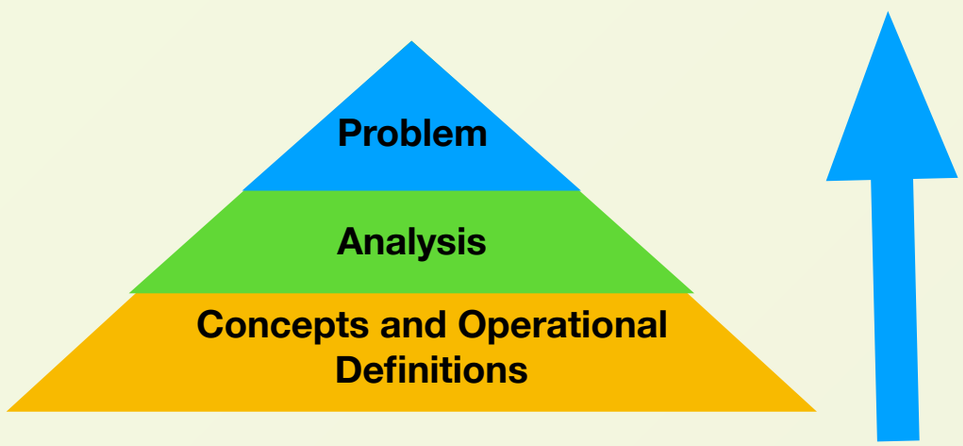
We found that children/young people/students embraced the Bottom-Up Approach.

Treat the teaching/learning of Computer Programming Languages in a similar fashion to teaching/learning Human Programming Languages.

Human Languages		Computer Programming Languages
Word	→	KeyWord
Sentence	→	Statement
Paragraph	→	Module
Essay	→	Program

This approach is to strengthen the grammar and fundamental building blocks

Bottom-Up Approach



Modifications to the Approach

(1) Primary Emphasis is on Correct/Expected Output

```
# Program 1
Apples = 10
Bananas = 12
Total = Apples + Bananas
print(Total)

Output
22
```

Advantage is People Start Testing and Debugging from the beginning and this empowers them and frees the teacher!!

(2) Fundamental Unit is a Useful., Working Block of Code.

```
For j in range(1,10):
    print(j)
```

```
Sum = 0

While j < 5:
  Number = int(input("Enter a number "))
  Sum = Sum + Number
  print(Sum)

Average = Sum/5

print(Sum)
print (Average)
```

(3) Teacher-Led Examples & Importance of Repetition & Memory

We memorise poetry, pieces of music, multiplication tables.

(4) Introducing the "Wow Factor"

Any Working Program is sufficient to enthuse the learner.

Curious Grown-Ups Codes

EuroPython 2021

Worked Example

```
1 # Program 5
2 for j in range(1,10,2):
3     print(j)
```

```
1
3
5
7
9
```

Output of 1st 5 odd numbers

Programs Created of Own Volition

```
1 for x in range(7,59,9):
2     print(x)
```

```
7
16
25
34
43
52
```

Output

```
1 for x in range(1,110,7):
2     print(x)
```

```
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
106
```

Output

"For Loop - I like it particularly, because it does all the laborious calculations so quickly and saves a lot of time….also codes eliminate human mistakes which could arise out of boredom of doing the same task over and over again"

Curious Grown-Ups Codes

EuroPython 2021

Worked Example

```
1 # Program 2
2 def Addition2(a,b):
3   Total = a + b
4   print(Total)
```

```
1 # Calling Program 2
2 Addition2(100,200)
```

```
00
```

```
1 Addition2(32,28)
```

```
0
```

Programs Created of Own Volition

```
1 def add_travel(b_number):
2   return b_number+20
```

```
1 add_travel(70)
```

```
90
```

```
1 def add_apartments(j_number):
2   return j_number+2000
```

```
1 add_apartments(900)
```

```
2900
```

```
1 def deduct_cars(H_number):
2   return H_number-40
```

```
1 deduct_cars(700)
```

```
660
```

Examples with Applications

Curious Grown-Ups Comment

"I like Selection and Conditional statements - I find it interesting that if, elsie and else take into account different situations and come up with with different options - which could help with making small or big decisions"

Interested Children Codes

EuroPython 2021

Worked Example

```
# Program make a simple calculator

# This function adds two numbers
def add(x, y):
  Total = x+y
  return(Total)

# This function subtracts two numbers
def subtract(x, y):
  Total = x-y
  return(Total)

def main():
  print("Select operation.")
  print("1.Add")
  print("2.Subtract")


  while True:
    # Take input from the user
    choice = input("Enter choice(1/2) or Enter Q to Quit: ")

    if choice =='Q':
      break

    # Check if choice is one of the four options
    if choice in ('1', '2'):
        num1 = int(input("Enter first number: "))
        num2 = int(input("Enter second number: "))

        if choice == '1':
            Result = add(num1,num2)
            print(num1, "+", num2, "=", Result)

        elif choice == '2':
            Result = subtract(num1,num2)
            print(num1, "-", num2, "=", Result)

    else:
        print("Invalid Input. Try again")

   # EndWhile

  print("Program has Ended")

main()
```

Question 1 - Now extend the above program to:-

1. Include an Option which allows users to choose Multiplication
2. Include an Option which allows user to choose Division
3. Allow the user to input both real and integers
4. Include an option which calculates the MOD of the numbers entered.

- List item

Answer to Question

```
# Put your code here
# Program make a simple calculator

# This function adds two numbers
def add(x, y):
  Total = x+y
  return(Total)

# This function subtracts two numbers
def subtract(x, y):
  Total = x-y
  return(Total)

def multiplication(x, y):
  Total = x*y
  return(Total)

def division(x, y):
  Total = x/y
  return(Total)

def mod(x, y):
  Total = x/y
  return(Total)

def main():
  print("Select operation.")
  print("1.Add")
  print("2.Subtract")
  print("3.Multiplication")
  print("4.Division")
  print("5.Mod")
  while True:
    # Take input from the user
    choice = input("Enter choice(1/2/3/4/5) or Enter Q to Quit: ")

    if choice =='Q':
      break

    # Check if choice is one of the four options
    if choice in ('1', '2', '3', '4', '5'):
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))

        if choice == '1':
            Result = add(num1,num2)
            print(num1, "+", num2, "=", Result)

        elif choice == '2':
            Result = subtract(num1,num2)
            print(num1, "-", num2, "=", Result)

        elif choice == '3':
             Result = multiplication(num1,num2)
             print(num1, "x", num2, "=", Result)

        elif choice == '4':
            Result = division(num1,num2)
            print(num1, "/", num2, "=", Result)

        elif choice == '5':
            Result = mod(num1,num2)
            print(num1, "%", num2, "=", Result)

        else:
          print("Invalid Input. Try again")

   # EndWhile

  print("Program has Ended")

main()
```

Tests

```
Select operation.
1.Add
2.Subtract
3.Multiplication
4.Division
5.Mod
Enter choice(1/2/3/4/5) or Enter Q to Quit: 5
Enter first number: 20
Enter second number: 3
20.0 % 3.0 = 6.666666666666667
```

1. Added the extra functions at the correct places

2. Studied the structure of the program and knew where and how to add these extra functions

Interested Children - Guess the number program

Question 7 - Write a Function where the Computer Randomly generates a number between 1 and 100. And you as the user have to guess the number until you get it right. The Program will report on whether the number you have guess is too high or too low or if you can guess it correctly.

Program Created

```
# Put Code Here
import random

def Geuss_the_number():
  computer = random.randint(1,100)
  print(computer)
  Active = True
  while Active == True:
    User_input = int(input("Enter a number between 1-100 "))
    if User_input == computer:
      print("you got it right")
      Active = False

    elif User_input > computer:
      print("too high")

    else:
      print("too low")

Geuss_the_number()


```

```
92
Enter a number between 1-1001
too low
Enter a number between 1-10080
too low
Enter a number between 1-100100
too high
Enter a number between 1-10090
too low
Enter a number between 1-10092
you got it right
```

1. Wrote Program from Scratch

2. Provided Slightly different structure from given answer

3. Inadvertently highlighted that there is more than 1 solution to a problem.

4. Also, tested the program.

Suggested Answer

```
def Guess_Number():
    import random
    Computer_Number = random.randint(1,10) # Computer Generates a Random Number between 1 and 10
    print("The Computer has generated the Number ", Computer_Number)
    while True:  #Notice a different way of putting the condition here
        User_Guess = int(input("Enter a Number "))
        if User_Guess == Computer_Number:
            break   #The break statement allows you to come out of the WHILE block
        elif User_Guess < Computer_Number:
            print("Try Again. Your Guess is Too Low ")
        else:
            print("Try again. Your Guess is Too High ")


    print("Well done. You have guessed the Correct Number")

```

```
Guess_Number()
```

```
The Computer has generated the Number  6
Enter a Number 5
Try Again. Your Guess is Too Low
Enter a Number 7
Try again. Your Guess is Too High
Enter a Number 10
Try again. Your Guess is Too High
Enter a Number 3
Try Again. Your Guess is Too Low
Enter a Number 6
Well done. You have guessed the Correct Number
```

Interested Children Codes

Exercise 9 - Please see the lists below for the number of people vaccinated in various different countries (data taken from John Hopkins website)

1. [China, USA, India, Germany, UK, Brazil]
2. [223299000, 161473715, 85453618,38843476,36099727]
3. Write a Program which calculates the total number vaccinated in this group. Use a FOR loop for this.
4. Extend the above program to calculate the average number of vaccinations in this group.

Covid-19 Vaccination Data

Program Created

```
1 vaccines = [223299000, 161472715, 85453618, 38843476, 36099727]
2 print("Sum using inbuilt function ", sum(vaccines))
3
4 def number_vaccines():
5   sum = 0
6   for j in range (len(vaccines)):
7     sum = sum + vaccines [j]
8   average = sum / len(vaccines)
9   print("\nSum", sum, "\nAverage ", average)
10 number_vaccines()
11
12
13
```

```
Sum using inbuilt function  545168536

Sum 545168536
Average  109033707.2
```

545 million

109 million

(1) He has solved the problem in 2 different ways:-

(i) Using Built-In Sum Function

(ii) Creating the Function

(2) He has developed the Program from Scratch

(3) He is using Real World Covid-19 Data from a Respected Scientific Source

(4) He has created the Program after studying lots of Worked Examples and playing with them.

They liked these large numbers!!

Conclusions & General Observations

(1) Interested Children had a more analytical approach whereas the curious grown-ups exhibited a more creative approach.

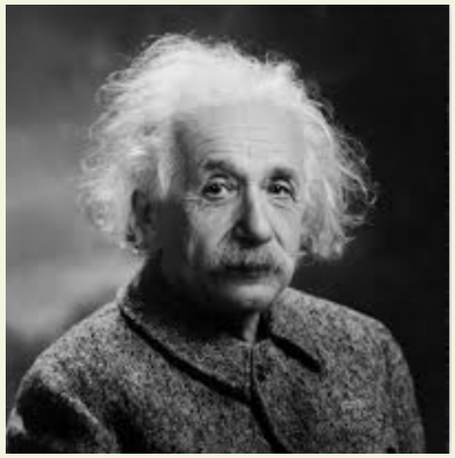
(2) Interested Children were more comfortable with the concept of errors and correcting them whereas the grown-ups placed more emphasis on precision.

(3) Interested Children had a more intuitive grasp of coding concepts whereas grown-ups had to be explicitly taught. However, the grown-ups seemed to appreciate the ideas better.

(4) Curious Grown-Ups voiced more interest in what would be the real-life applications of all of this.

(5) This framework of a teacher-led explanatory approach with an emphasis on memory, repetition and blocks of code appeared to be successful.

(6) Both curious adults and interested children felt they were making good progress and felt happy.



“I never teach my pupils, I only attempt to provide the conditions in which they can learn”

(Albert Einstein)

- Any Questions?
- Thank you for your time.

- (My) YouTube Channel: lil anonymous
- Website: https://computersciencegcealevel.wordpress.com
- Email: demo999@yahoo.com