Back to the Future:
A Line-Numbered Python Interpreter with BASIC Flavor

PyBas: A toy interpreter with Retro look and Python under the hood

Abdul Arfan @ PyCon Indonesia 2025

Overview

```
10 About Me
20 But Why?
30 Features And Sample Code
40 How It Works
50 IF WE HAVE TIME GOTO 70
60 END
70 DEMO
```

About Me

```
10 Abdul Arfan
20 Tech Enthusiast
30 Backend/Fullstack Engineer
40 Educator -> Teacher, Lecturer
50 github.com/arfan
```

But Why?

```
10 Nostalgia
    -> First time I learned coding in
       Junior high school using BASIC
20 Teaching tools
    -> learn control and understand
       loops with just IF and GOTO
    -> similar concept with assembly language
       i.e. PC (program counter)
30 Learn to build simple interpreter
40 Add python power to BASIC
    -> Using python data structure
       Inside basic-like syntax
```

Features And Sample Code

10 Interpreter
    $ ./pybas hello.pybas
    -> Will run the program
20 REPL
    $ ./pybasrepl

30 Formatter
    $ ./pybasfmt
    -> Will format existing source code

Features And Sample Code (2)

Hello World (infinite loop):

```
10 Print "Hello World!"
20 GOTO 10
```

Features And Sample Code (3)

Loop using control

```
10 REM Loop Control
20 PRINT "Counting from 1 to 5:"
30 LET I = 1
40 PRINT "Number " + str(I)
50 IF I >= 5 GOTO 80
60 LET I = I + 1
70 GOTO 40
80 END
```

Features And Sample Code (4)

Loop using FOR NEXT

```
10 PRINT "Testing FOR loops"
20 FOR I = 1 TO 3
30    PRINT "Loop iteration: " + str(I)
40 NEXT I
50 PRINT "Loop finished"
60 END
```

Features And Sample Code (5)

User Input

```
10 PRINT "Welcome to the Personal Information Program!"
20 INPUT "Enter your name: ", NAME
30 INPUT "Enter your age: ", AGE
40 INPUT "Enter your favorite number: ", FAVNUM
50 PRINT "Hello, " + NAME + "!"
60 PRINT "You are " + str(AGE) + " years old."
70 PRINT "Your favorite number is " + str(FAVNUM)
80 LET DOUBLE_FAV = FAVNUM * 2
90 PRINT "Double your favorite number is " +
str(DOUBLE_FAV)
100 LET BIRTH_YEAR = 2025 - AGE
110 PRINT "You were born around " + str(BIRTH_YEAR)
120 END
```

Features And Sample Code (6)

Subroutine

```
10 PRINT "Main program starts"
20 LET X = 5
30 GOSUB 100
40 PRINT "Back in main program"
50 PRINT "Main program ends”
60 END

100 PRINT "In subroutine, X ="
110 PRINT X
120 LET X = X + 1
130 RETURN
```

Features And Sample Code (7)

Python style function

```
10 PRINT "Simple function test"
20 RESULT = DOUBLE(5)
30 PRINT "Double of 5 is:"
40 PRINT RESULT
50 END

100 DEF DOUBLE(X)
110   RETURN X * 2
120 ENDDEF
```

Features And Sample Code (8)

Modern power in retro syntax

```
10 LET NUMBERS = [1, 2, 3, 4, 5]
20 PRINT NUMBERS
30 PRINT NUMBERS[2]    ' prints 3

40 LET AGES = {"Alice": 30, "Bob": 25}
50 PRINT AGES["Bob"]    ' prints 25


60 LET SQUARES = [x*x for x in range(5)]
70 PRINT SQUARES        ' [0, 1, 4, 9, 16]
```

How It Works

Architecture at a glance

- Program stored as python dictionary {line_number: statement}
- Program counter (PC): current line
- Stacks:
  - gosub_stack -> remembers subroutine returns
  - for_stack -> manages loops
- Variables: dictionary self.vars

How It Works

Loading the program & PC

- Load lines -> ignore comments -> map line -> statements
- PC starts at lowest line number
- Next line = jump target OR next higher line number
- Python snippet:
  - next_pc = self.execute (stmt)
  - self.pc = next_pc or min (n for n in program if n> self.pc)
- Python **eval** under the hood

How It Works

Control flow: GOTO & GOSUB

- GOTO N: jump to line N
- IF conditions THEN GOTO N: conditional jump
- GOSUB N:
  - Push return address
  - Jump to subroutine
- RETURN: pop address and continue
  - Subroutines are the ‘functions’

NEXT

- Translate the syntax
- Create ‘real’ compiler
- Create IDE (sample: vscode plugin)
- More...

Thank You