200 Benchers Only

By: Connor Kippes, Luke Laurie, and Ryan Bullard

About

Is this you using an unintuitive language like *Python*??

About

- Well worry not 200BencherOnly is here!
- Only people who can bench 200+ pounds can effectively use this language
- It is **STRONGLY** typed and uses type annotations
- It is imperative and 3rd generation like common languages such as Java and Python
- Could also be considered a scripting language because it is *very* readable and fast to code in.
- Similar to Java and translated into Python

Types

- Integers:
  - lightWeight: integers under 200
  - weight: integers over 200 but less than 1000
  - samSulek: ints over 1000
  - ryanBullard: ints < 0
- Booleans: tryBench
  - True: gotItUp
  - False: failed
- Strings: cables
- longs: pr
- floats: smallPlate - only 2.5 value allowed

Variable names and assignment

- Can only be these muscle groups followed by any number of digits:
  - pecs, delts, lats, biceps, triceps, abs, obliques, quads, hamstrings, glutes, calves, forearms
  - Could also be pecs1, delts4321, etc.
  - This is because you are usually assigning weight to a variable so you are training that muscle
- Var assignment (=): loadBar
- End of statements (;): pump
- Ex- int a = 2000; → samSulek glutes loadBar 2000 pump

Integer operations

- Addition (+): creatine
  - Ex- 1 + 1 ⇒ 1 creatine 1
- Subtraction (-): restDay
- Multiplication (*): steroids
- Division (/): vegan
- Modulus (%): muscleMass
- Increment (_++): superset
  - Ex- i++ = lats superset

Boolean and comparison operations

- And (&&): crushed
- Or (||): settle
- Not (!): spotter
- greater than (>): biggerThan
- less than (<): smallerThan
- equal to (==): sameSize

Loops and conditionals

Loops:

python:

```
for x in range(1,3):
```

If statements:

java:

```
If (__) {
} else {
}
```

200BenchersOnly:

```
set pecs, 1 to 3 leftWeightClip
rightWeightClip
```

```
canYouLift (__) leftWeightClip
rightWeightClip yourAFailureSo leftWeightClip
rightWeightClip
```

Misc features

- Functions: workout <type> <name> (<params>)
- Braces for scoping
  - {: leftWeightClip
  - }: rightWeightClip
- Return: gains
- Print: showoff()
- Command Line arguments:
  - If none then run in debug mode
  - otherwise 1st arg is file name to read lang where we will parse and translate to Python
  - Remaining args are set variables in python- integers are converted
- Commenting:
  - sayToGymBro <comment here>

Added features

- Interactive system to type in commands and show parsing process and result
- Scoping system
- Some generic error messages
- Custom names for everything
- Have more integer subtypes
- Floats and longs (technically)
- Increment
- Commenting
- Type checking
- Function names and calls

Process

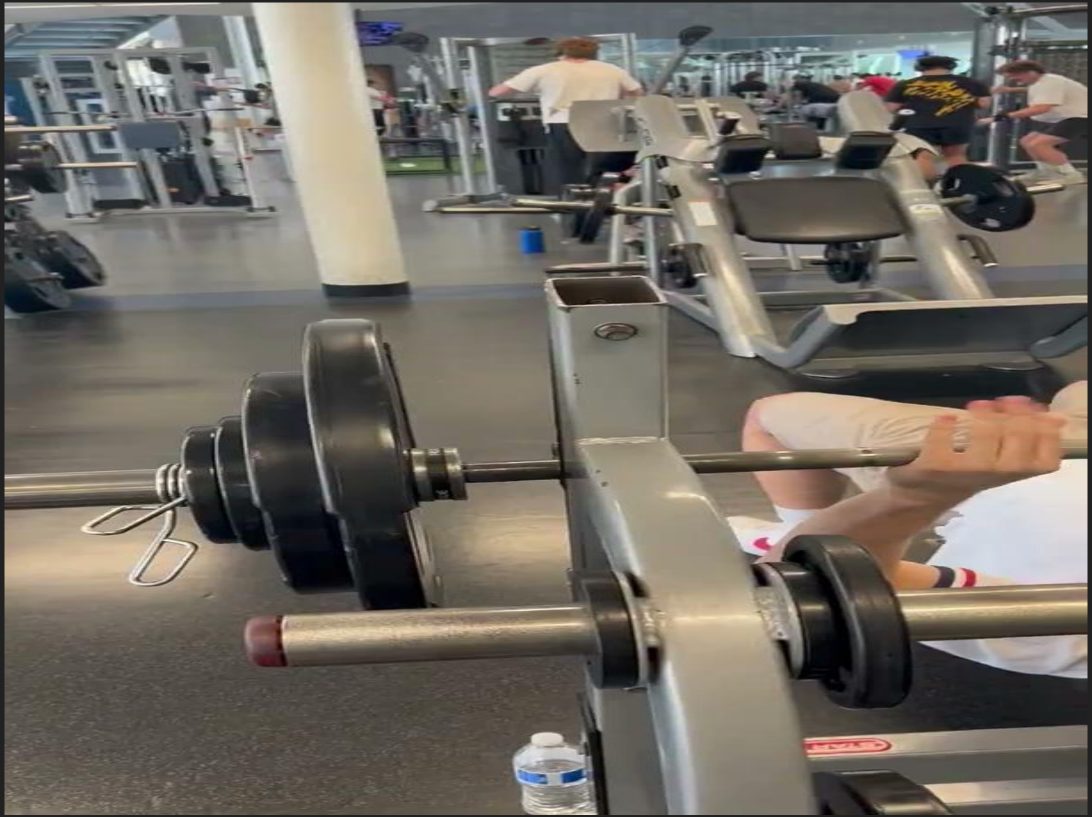
- First based code off of the example code given and had to learn the regex
- Then we added a bunch of patterns, wrote functions to match them, and had a main class to go through commands and see if it was variable assignment, an if statement, loop, etc.
- Then we changed some of the parser functions to return the corresponding code on the way back up of the parcer and throw exceptions up if invalid so we could do the translating all in one class.
- Then we added some additional features, fixed bugs, tested, and polished it.

Challenges

- Handling block scoping
- Reworking parser to throw exceptions and return python code strings
- Handling indentation of Python lines
- Problems with regex where we had to separate expressions
- Type checking was difficult to implement because we added it after everything

Coding demo

demo



Code demo- open eclipse

What we would change or add

- Add more type checking cases that we missed
- Add more custom error messages
    - For example, if someone tries to assign 100 to a weight variable we would say “that's lightweight buddy”
- Add else if statements
- Clean up the code a bit
- Add some more miscellaneous features to use more gym puns