Nested For Loops

Thomas Schwarz, SJ

Nesting

- We can use nested loops
  - E.g. to solve mathematical puzzles or diophantine equations
    - Where we just try out all possibilities
      - Nota bene: Diophantes was preceded by Indian mathematicians

Diophantine Equations

- A diophantine equation is a polynomial equation with variables in the integers
  - E.g. $x^2 - y^2 = z^3$
    - Has obviously trivial solutions
      - x any number, $y = x$, $z = 0$
    - We exclude those by restricting ourselves to $x > 0,\ y > 0,\ z > 0$

Diophantine Equations

- This still leaves us with an infinite number of number combinations
    - So, we *arbitrarily* limit ourselves to numbers < 1000
    - First attempt:

```
for x in range(1,1000):
    for y in range(1,1000):
        for z in range(1,1000):
            if x**2-y**2 == z**3:
                print(x, y, z)
```

Diophantine Equations

- Question:
  - How often are we executing the if-statement?
    - 999 times per y
    - 999×999 per x
    - $999 \times 999 \times 999$ total
    - which is a lot (almost 1000 million)

```
for x in range(1,1000):
    for y in range(1,1000):
        for z in range(1,1000):
            if x**2-y**2 == z**3:
                print(x, y, z)
```

Diophantine Equations

- We can reduce this by observing that for a solution
  - $0 < x < 1000$
  - $0 < x < y$
  - $0 < z < x$
- Now:

```
for x in range(1,1000):
    for y in range(1,x):
        for z in range(1,x):
            if x**2-y**2 == z**3:
                print(x, y, z)
```

Diophantine Equations

- Number of executions of the if-statement is now:
  - For each x
    - $(x-1)\times(x-1)$
  - Total number of times $\sum_{i=1}^{999}(x-1)^2$ is 331,835,499 or about a third of the previous value
  - But still takes noticeable time
    - And we get a bunch of solutions with absolutely no insight

Cryptarithmetic Puzzles

- A classic example:

```
  SEND
 +MORE
 _____
 MONEY
```

- `D, E, M, N, O, R, S, Y` are digits
  - In fact, they are different digits
  - And `S` and `M` are non-zero

Cryptarithmetic Puzzles

```
 SEND
+MORE
-----
MONEY
```

- We can of course deduce solution
  - E.g. M has to be one
  - Therefore, $S = 9$
  - …

Cryptarithmetic Puzzles

- Our "*brute force*" algorithm could just start out with
-

```
for d in range(10):
    for e in range(10):
        for n in range(10):
              …
              …
            #Test all conditions here
```

Cryptarithmetic Puzzles

- This just guarantees a lot of unnecessary tests
  - Easier to exclude cases as soon as possible
    - That all variables have to be different is a good candidate for early tests
    -

```
for d in range(10):
   for e in range(10):
      if e!= d:
         for n in range(10):
             if n!=e and n!=d:
                  …
                  …
                #Test here
                #Print out result
```

Cryptarithmetic Puzzles

- Now we need to deal with arithmetic!
  - Adding involves carries
    - E.g.:

```
 7
+8
15
```

    - Carry out of $a + b$ is $(a + b)//10$
    - Resulting digit is $(a + b)\ \%\ 10$

Cryptarithmetic Puzzles

```
for d in range(10):
    for e in range(10):
        if not e == d:
            for m in range(1,10):
                if not m==d and not m==e:
                    for n in range(10):
                        if not n==m and not n==e and not n==d:
                            for o in range(10):
                                if not o==n and not o==m and not o==e and not o==d:
                                    for r in range(10):
                                        if not r==n and not r==o and not r==m and not r==e and not r==d:
                                            for s in range(1,10):
                                                if s != n and s!=r and s!= o and s!=m and s!=e and s!= d:
                                                    y = (d+e)%10
                                                    if y!=s and y!= r and y!=o and y!=r and y!=s and y!= d and y!=e and y!=m:
                                                        c1 = (d+e)//10
                                                        if e == (n+r+c1)%10:
                                                            c2 = (n+r+c1)//10
                                                            if n == (e+o+c2)%10:
                                                                c3 = (e+o+c2)//10
                                                                if (s+m+c3)%10 == o and (s+m+c3)//10 == m:
                                                                    print(' ', s, e, n, d)
                                                                    print('+', m, o, r, e)
                                                                    print(m, o, n, e, y)
```