

Introduction

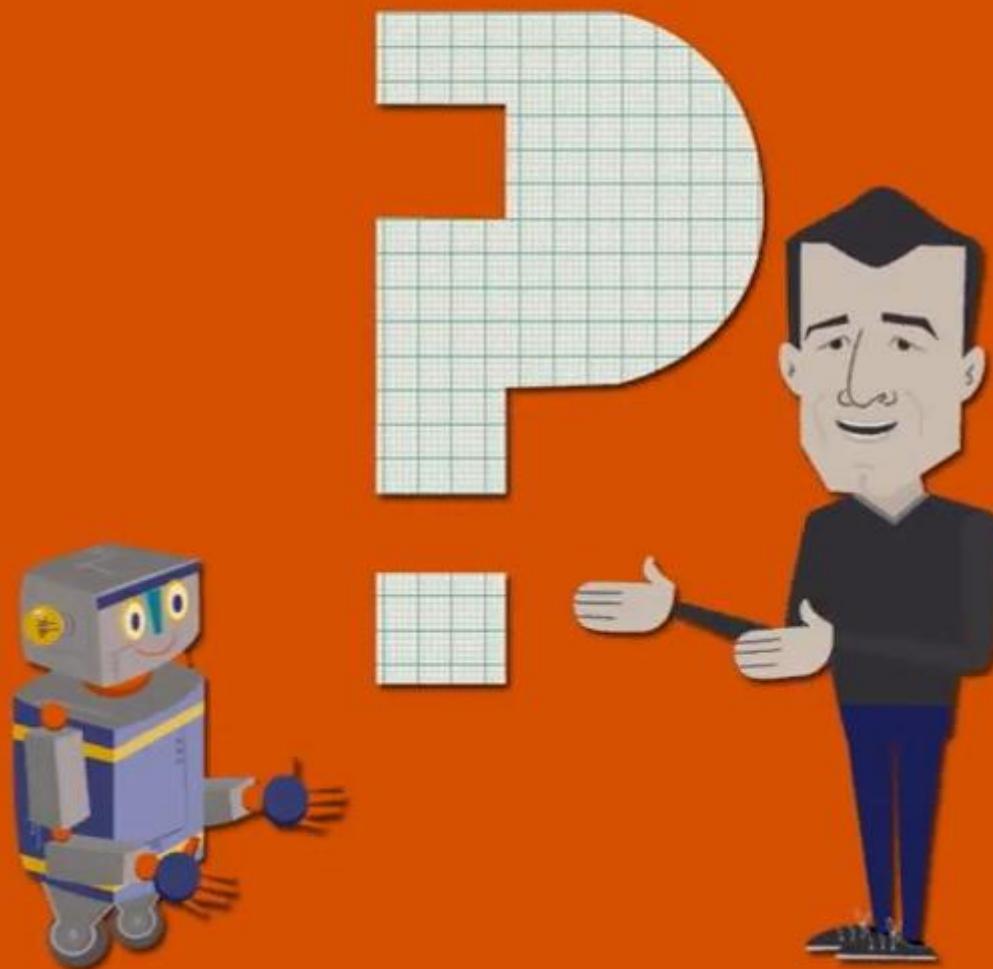
Struktur Data dan Algoritma Pemrograman

- ▶ Struktur Data
- ▶ Algoritma
- ▶ Pemrograman

History of Algorithm?

- ▶ Abu Ja'far Muhammad Ibnu Musa Al Khuwarizmi adalah seorang penulis buku Arab yang berjudul Kitab Al Jabar Wal Muqabala (Buku Pemugaran dan Pengurangan). Kata Al Khuwarizmi dibaca orang Barat menjadi algorism.
- ▶ Kata algorism berarti proses menghitung dengan angka Arab. Seseorang dikatakan algorist jika orang tersebut menggunakan angka Arab.
- ▶ Kata algorism lambat laun menjadi algorithm disebabkan kata algorism sering dikelirukan dengan kata arithmetic sehingga akhiran –sm berubah menjadi –thm. Kata algorithm diserap ke dalam bahasa Indonesia menjadi algoritma.

WHAT'S AN ALGORITHM



Analogi :

- } Jika seseorang ingin mengirim surat kepada kenalannya di tempat lain, langkah yang harus dilakukan adalah:

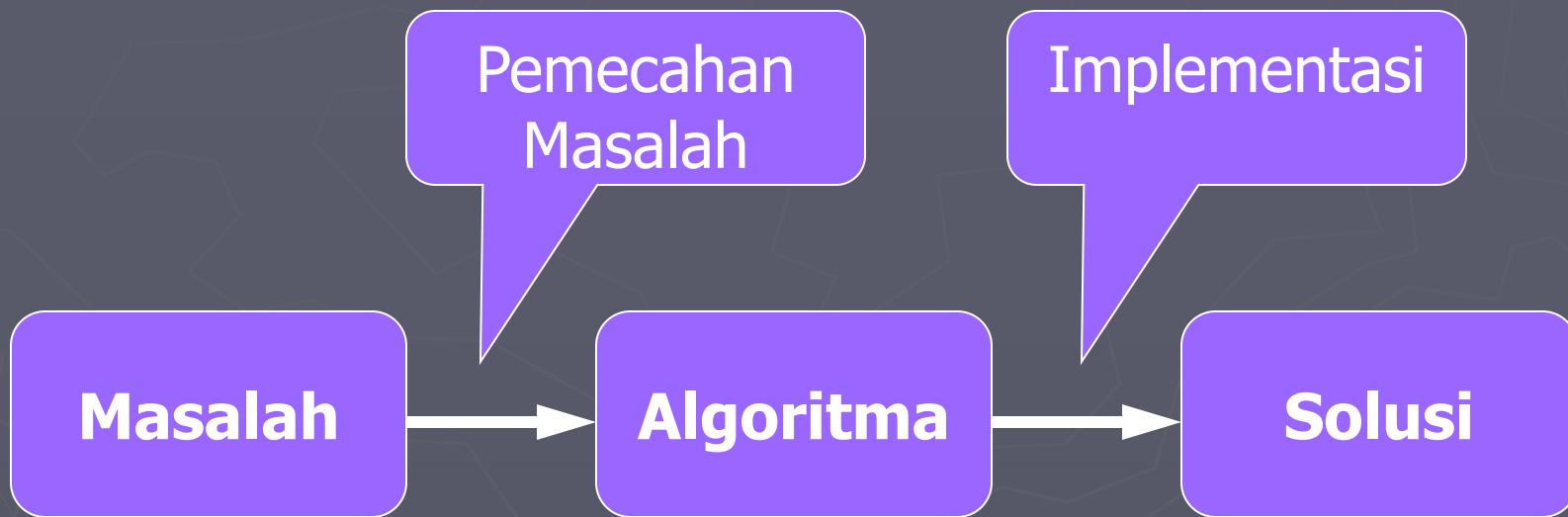
Langkah :

- } Menulis surat
- } Surat dimasukkan ke dalam amplop tertutup
- } Amplop dikasih alamat penerima dan pengirim
- } Amplop ditempeli perangko secukupnya.
- } Pergi ke Kantor Pos terdekat untuk mengirimkannya

What is Algorithm?

- ▶ Urutan langkah-langkah **logis** penyelesaian masalah yang disusun secara sistematis
- ▶ Urutan **logis** pengambilan keputusan untuk pemecahan masalah
- ▶ **Logis** : hasil dari urutan langkah tersebut harus dapat ditentukan benar atau salah
- ▶ Dalam bidang pemrograman,
 - Algoritma didefinisikan sebagai suatu metode khusus yang tepat dan terdiri dari serangkaian langkah yang terstruktur dan dituliskan secara sistematis yang akan dikerjakan untuk menyelesaikan suatu masalah dengan bantuan komputer.

Proses Penyelesaian masalah dengan Algoritma



ROOM



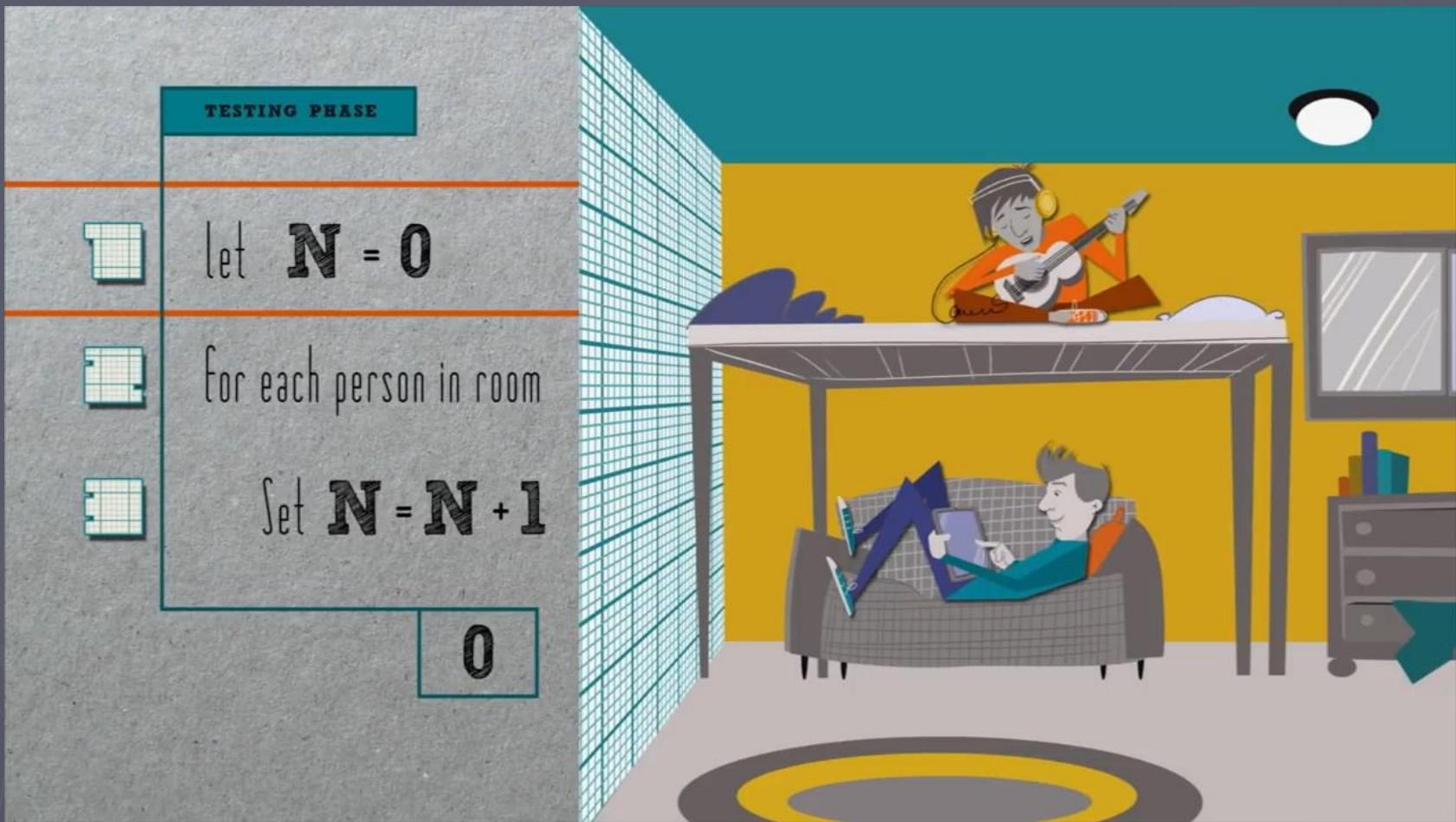
TESTING PHASE

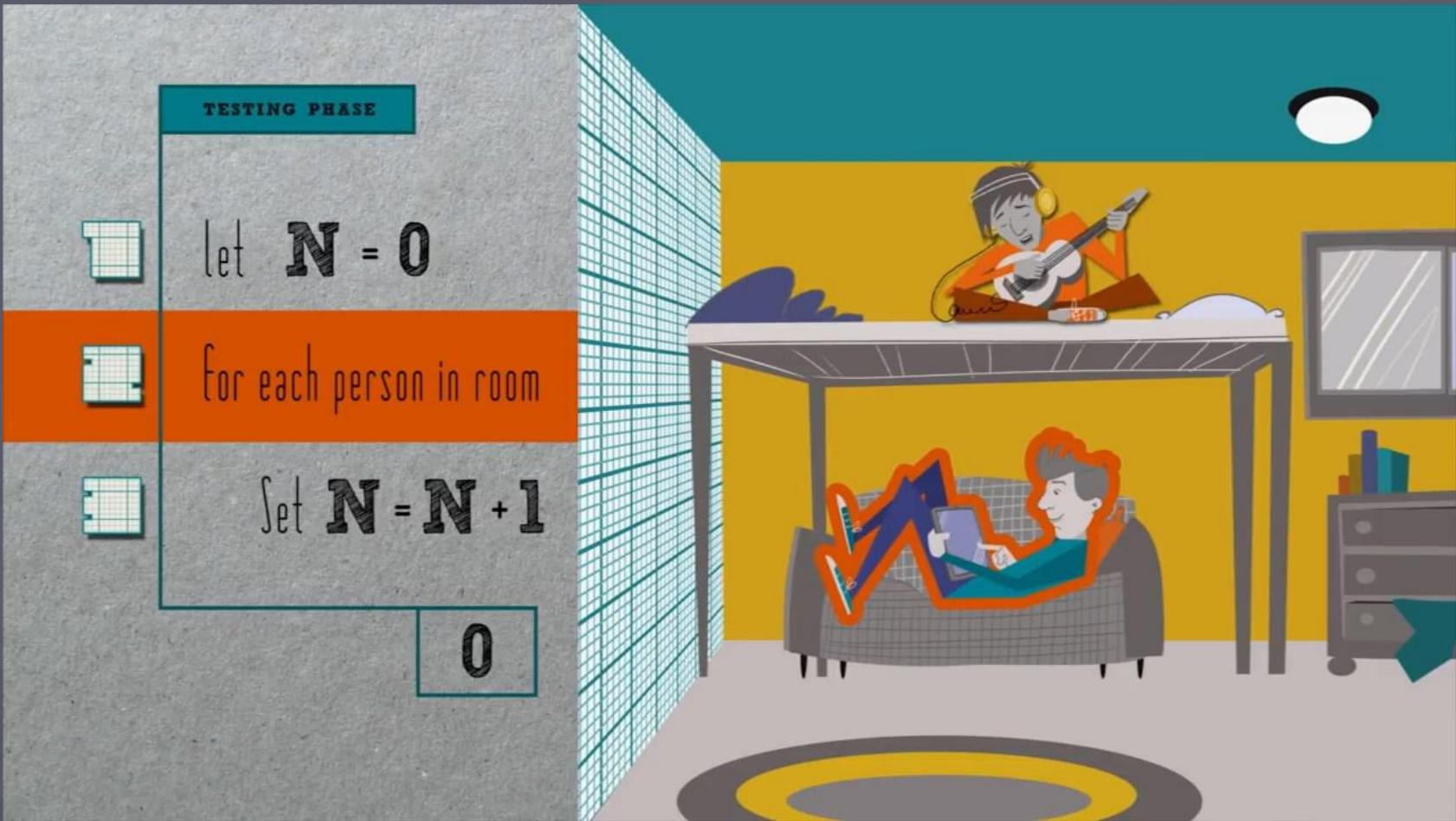
let **N** = 0

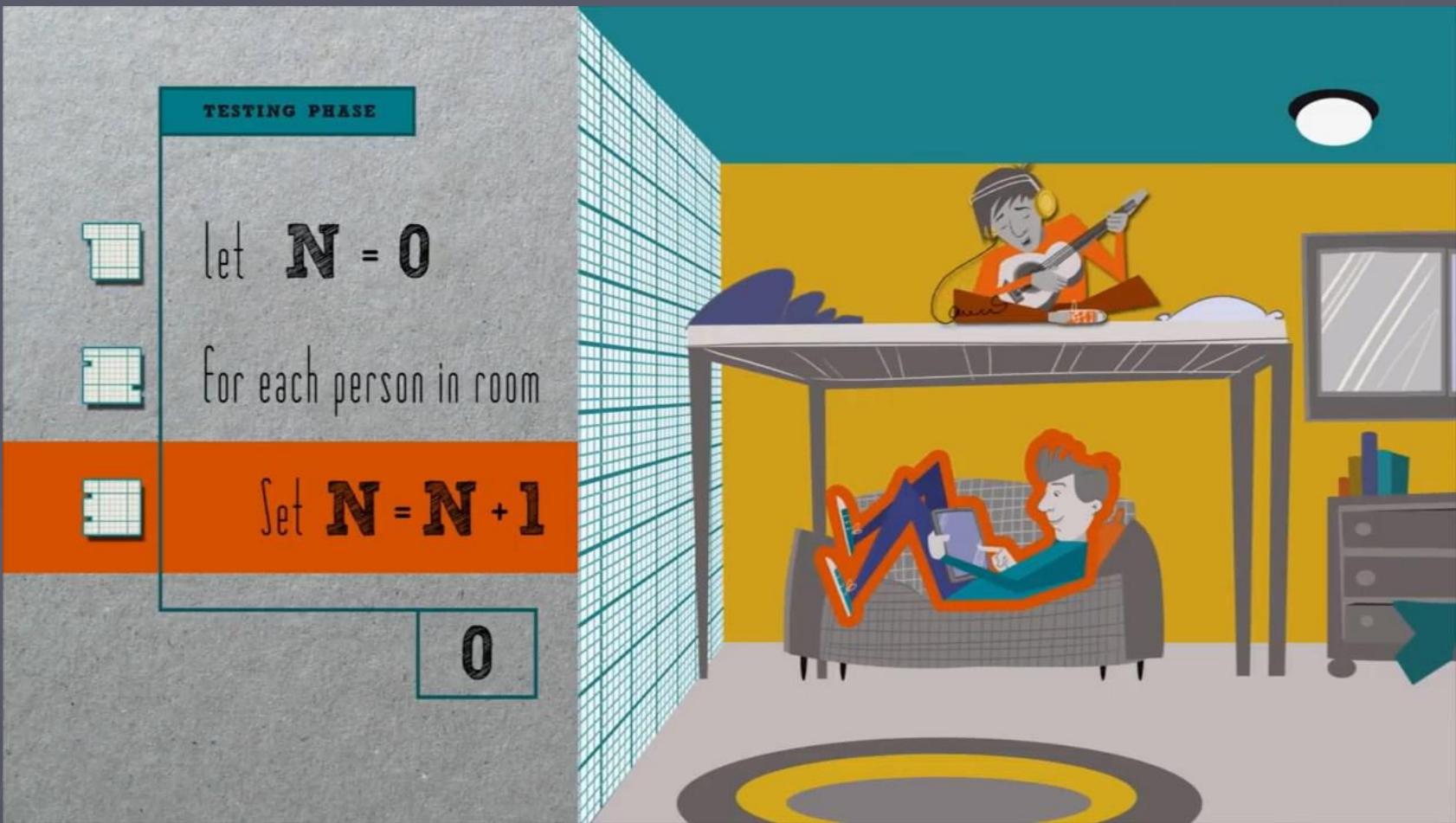
For each person in room

Set **N** = **N** + 1









TESTING PHASE

let **N** = 0

For each person in room

Set **N** = **N** + 1

1



TESTING PHASE

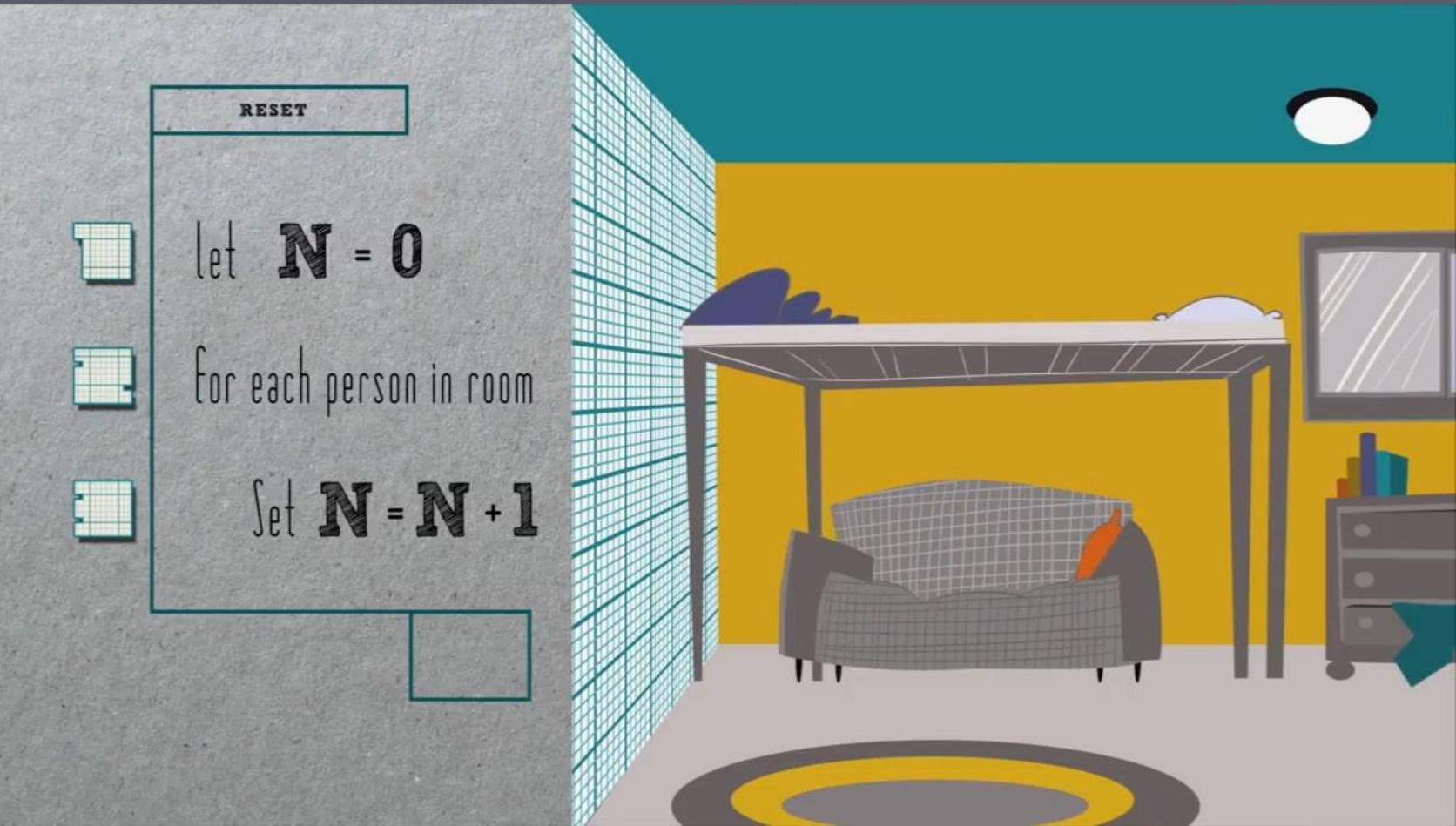
let **N** = 0

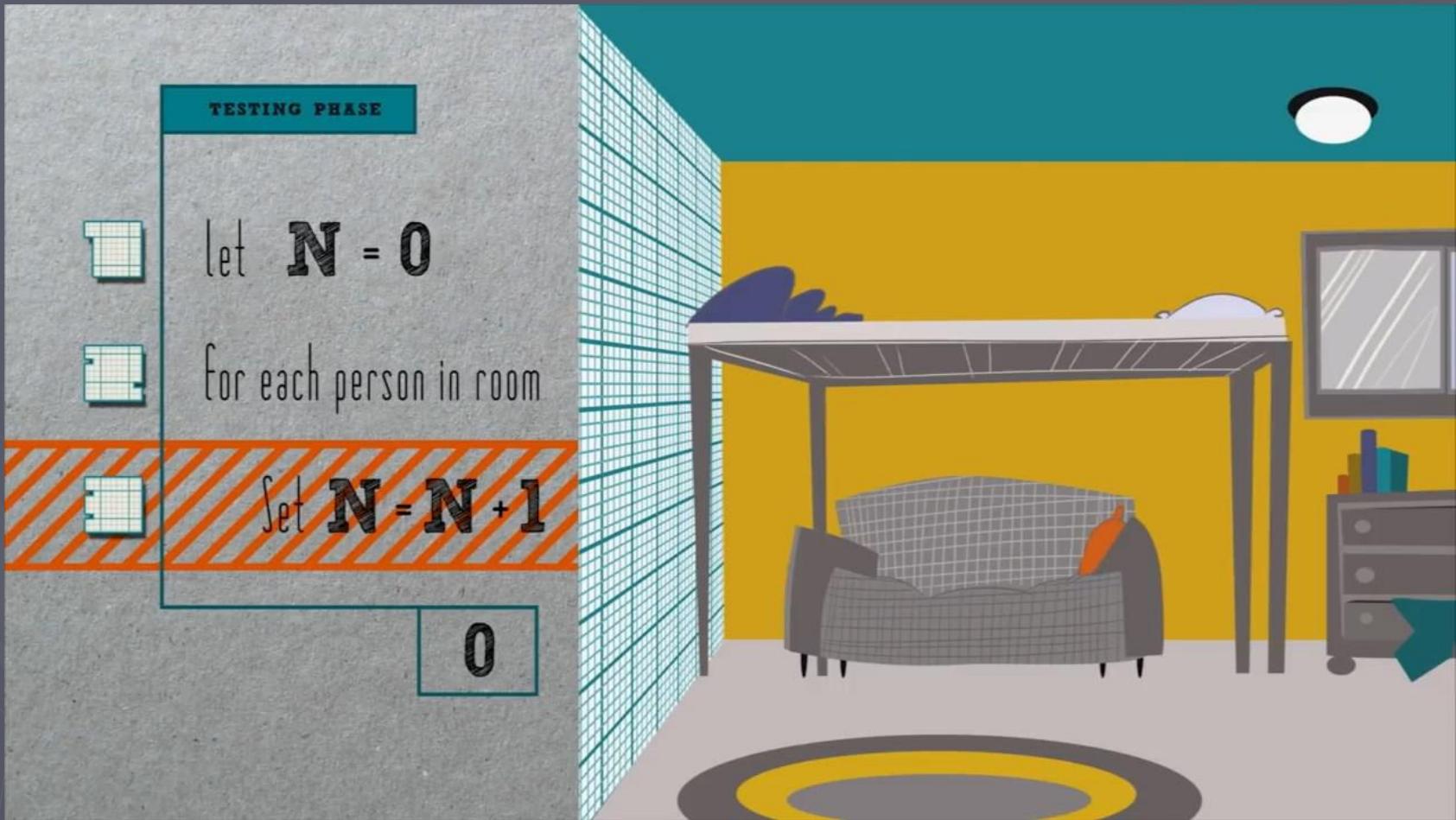
for each person in room

Set **N** = **N** + 1

2





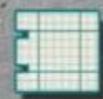




let **N** = 0



for each person in room



Set **N** = **N** + 1

ROOM



TESTING PHASE



let **N** = 0

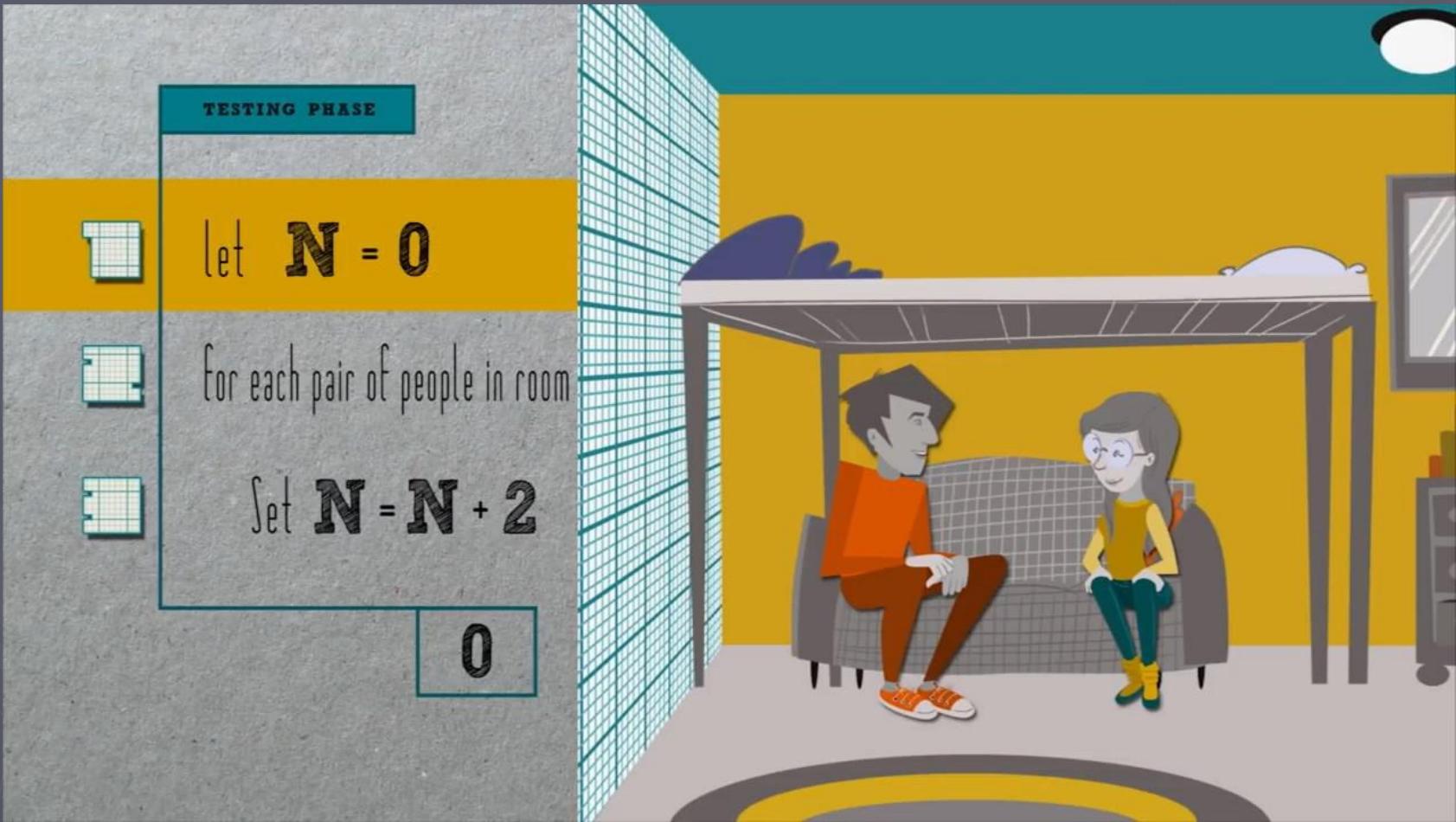


For each pair of people in room



Set **N** = **N** + 2





TESTING PHASE

let **N** = 0
For each pair of people in room
Set **N** = **N** + 2

0

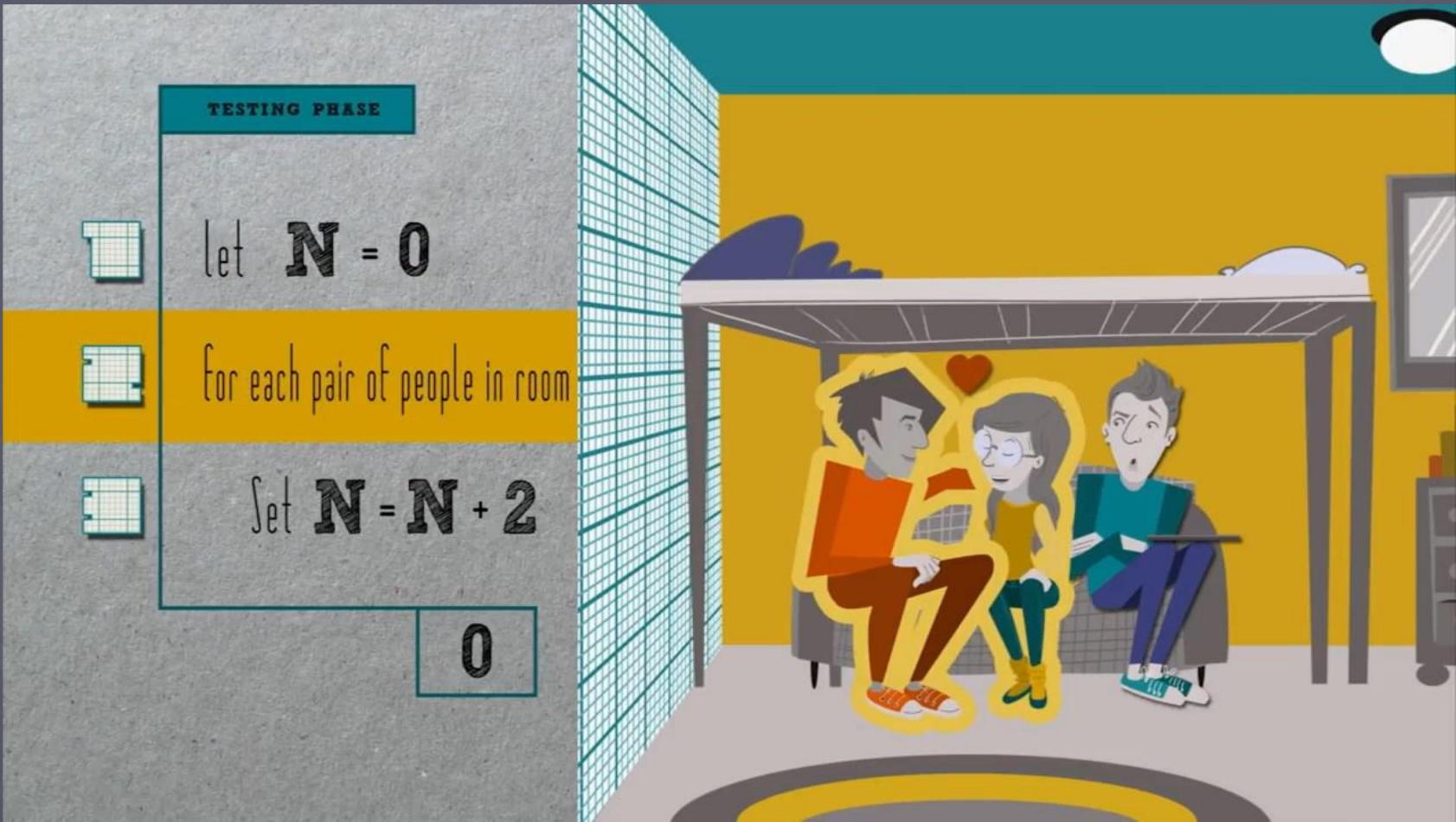


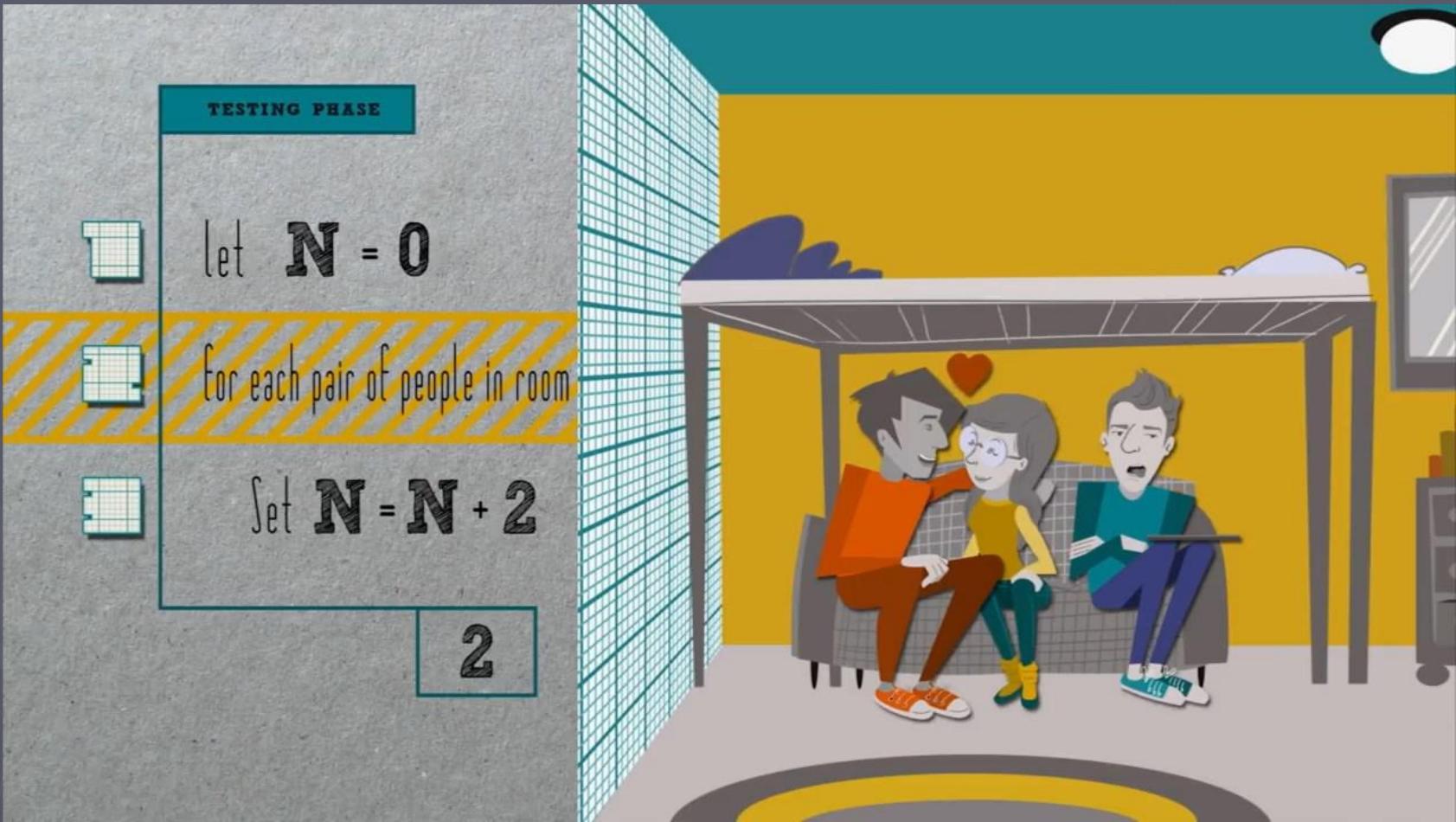
TESTING PHASE

```
let N = 0
for each pair of people in room
    Set N = N + 2
```

2

The slide features a sidebar on the left with three icons: a clipboard, a document with a grid, and another document with a grid. The main area contains pseudocode for a 'TESTING PHASE'. The code starts with 'let N = 0', followed by a loop 'for each pair of people in room' which contains the statement 'Set N = N + 2'. At the bottom right of the main area is a large, bold number '2'.





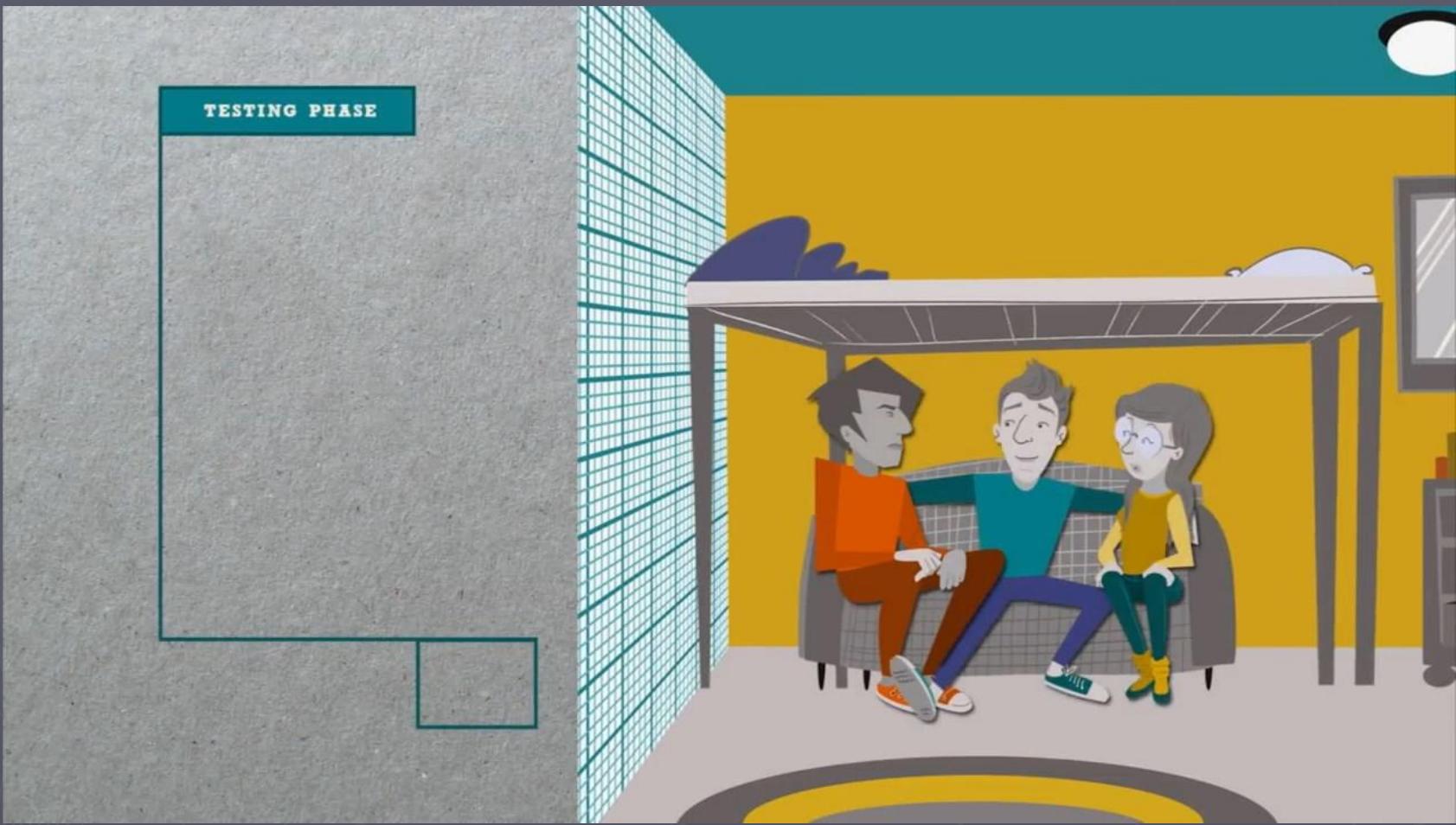
TESTING PHASE

```
let N = 0  
for each pair of people in room  
    Set N = N + 2
```

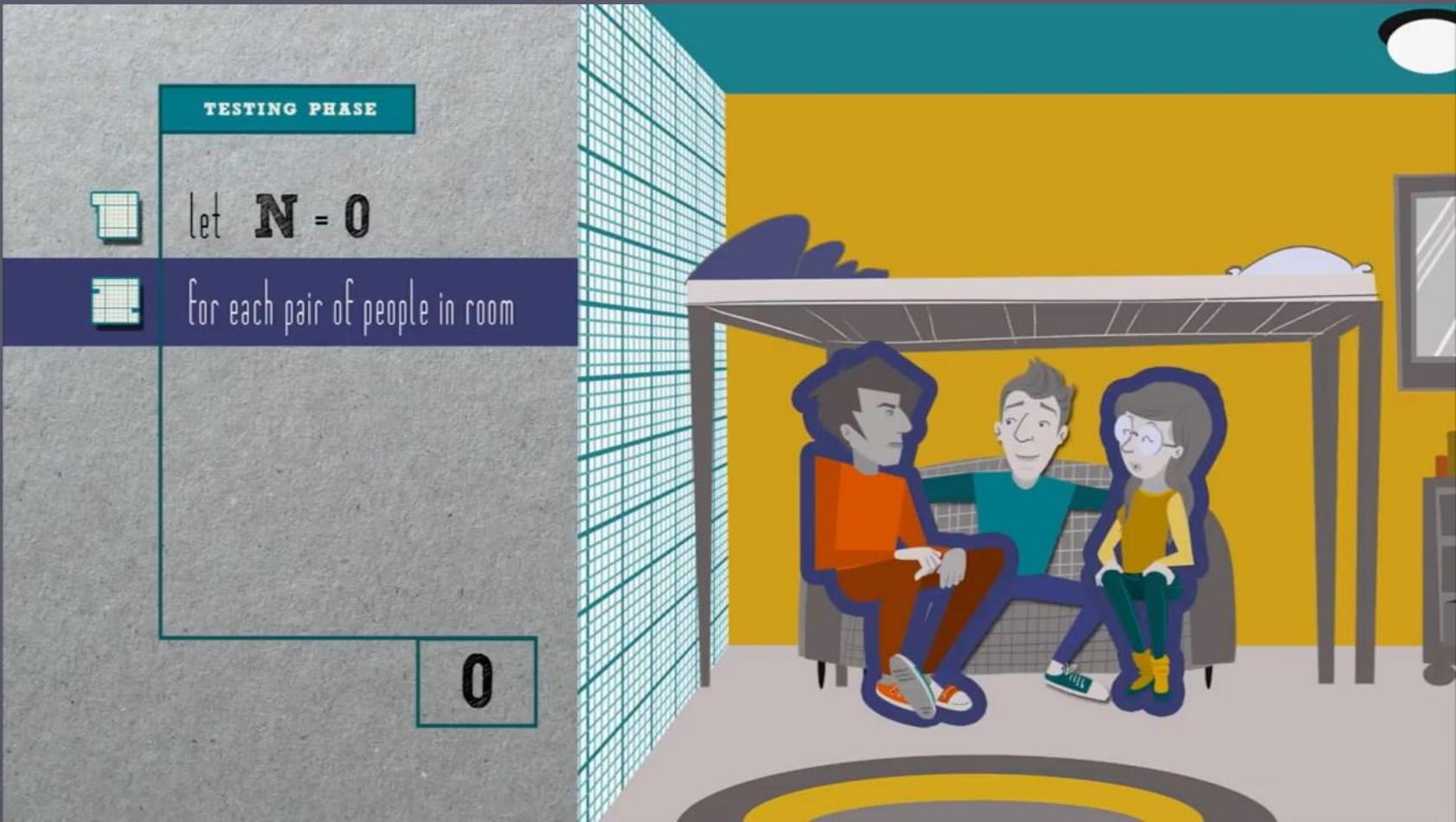
2

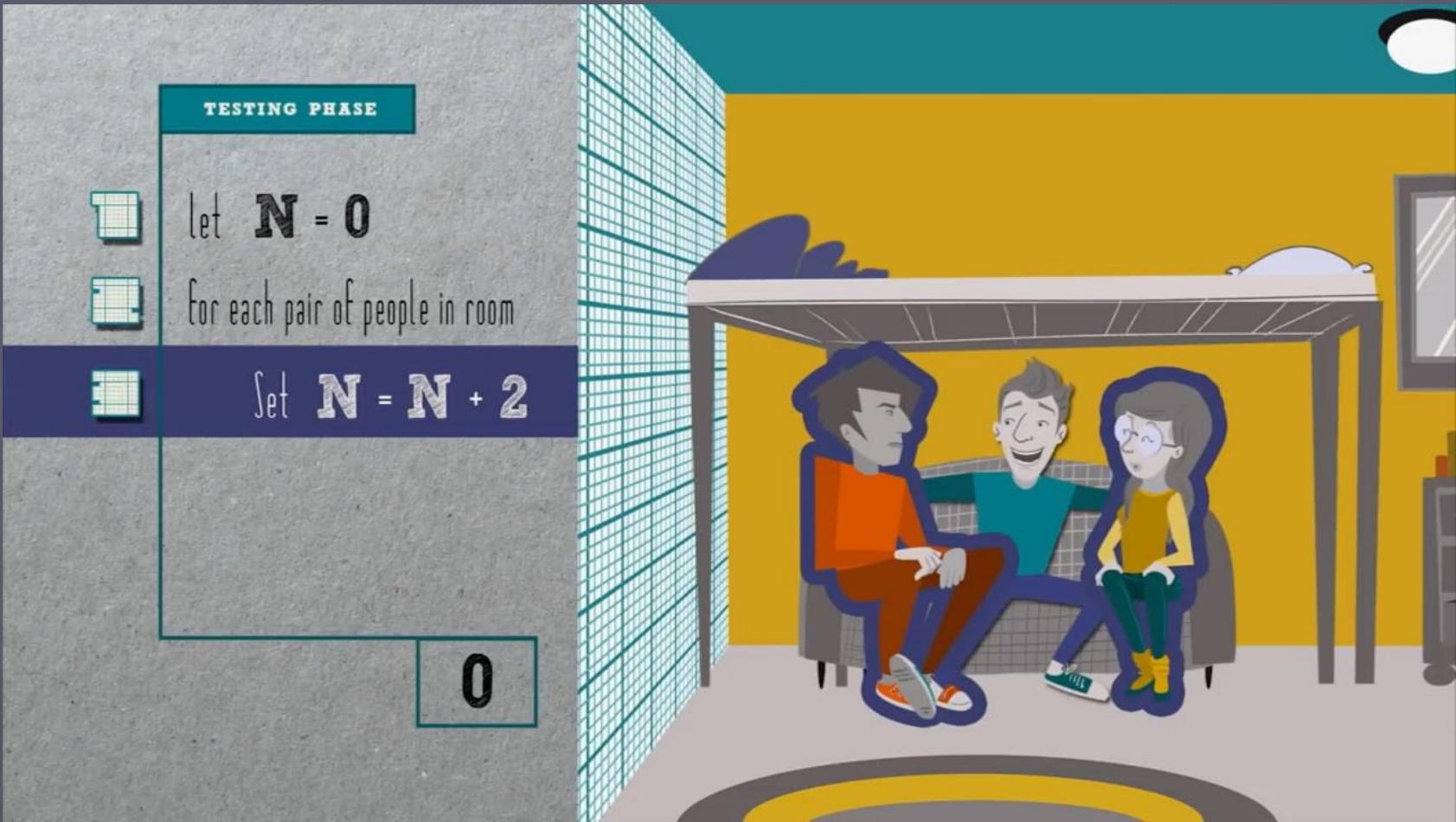












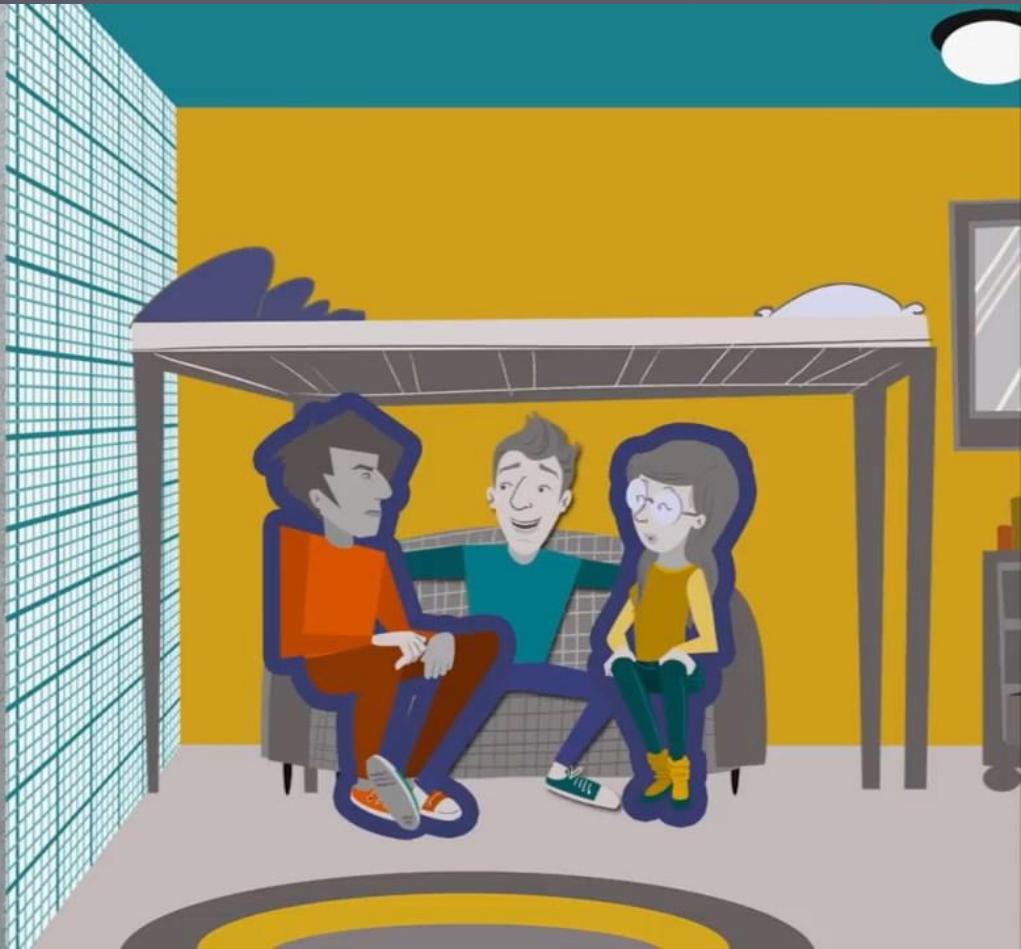
TESTING PHASE

let **N** = 0

For each pair of people in room

Set **N** = **N** + 2

2



TESTING PHASE

let **N** = 0

For each pair of people in room

Set **N** = **N** + 2

If 1 person remains then

2



TESTING PHASE

let $N = 0$

For each pair of people in room

Set $N = N + 2$

If 1 person remains then

Set $N = N + 1$

2



TEST COMPLETE



let **N** = 0

For each pair of people in room

Set **N** = **N** + 2

If 1 person remains then

Set **N** = **N** + 1

3



Why..??



Contoh

- ▶ Ibu Tati Mengupas Kentang

Ruang Lingkup

- ▶ Apakah kentangnya harus dibeli dulu atau sudah ada di dapur ?
- ▶ Apakah yang dimaksud dengan mengupas kentang untuk makan malam berarti sampai kentang terhidang?
- ▶ Ketika kentangnya terhidang, jadi sup, digoreng atau direbus saja?

Constraint

- ▶ Initial State :
Kentang sudah ada di kantong kentang,
yang ditaruh di rak di dapur dimana ibu tati
akan mengupasnya
- ▶ Final State :
Kentang dalam keadaan terkupas di panci,
siap untuk dimasak dan kantong
kentangnya dikembalikan ke rak lagi

Sub-Aksi

- ▶ Ambil kantong kentang dari rak
- ▶ Ambil panci dari almari
- ▶ Kupas kentang
- ▶ Kembalikan kantong kentang ke rak

- ▶ Ambil kantong kentang dari rak
- ▶ Ambil panci dari almari
- ▶ Depend on warna baju
 - Berwarna muda : pakai celemek
 - Tidak berwarna muda : -
- ▶ Kupas kentang
- ▶ Kembalikan kentang ke rak

- ▶ Ambil kantong kentang dari rak
- ▶ Ambil panci dari almari
- ▶ Depend on baju
 - Berwarna muda : pakai celemek
 - Tidak berwarna muda : -
- ▶ While jumlah kentang terkupas belum cukup do
 - Kupas 1 kentang
- ▶ Kembalikan kantong kentang ke rak

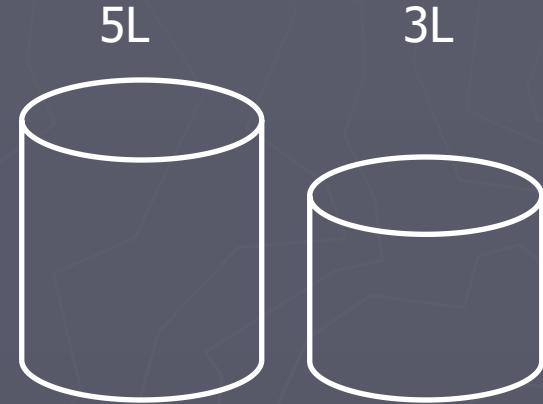
Kasus lain



Bagaimana Algoritmanya ??

Kasus lain

Misalkan terdapat dua buah ember, masing-masing mempunyai volume 5 liter dan 3 liter. Tuliskan algoritma untuk memperoleh air sebanyak 4 liter dengan hanya menggunakan kedua ember tersebut.



Bagaimana Algoritmanya ??

5 ciri penting algoritma

- ▶ Algoritma harus berhenti setelah mengerjakan sejumlah langkah terbatas
- ▶ Setiap langkah harus didefinisikan dengan tepat dan tidak memiliki dua arti (ambigu)
- ▶ Algoritma memiliki nol atau lebih masukan (input)
- ▶ Algoritma memiliki nol atau lebih keluaran (output)
- ▶ Algoritma harus efektif

Program dan Bahasa Pemrograman

- ▶ **Program** adalah formulasi sebuah algoritma dalam bentuk bahasa pemrograman sehingga siap untuk dijalankan pada mesin komputer
- ▶ **Bahasa pemrograman** adalah bahasa buatan yang digunakan untuk mengendalikan perilaku dari sebuah mesin, biasanya berupa mesin komputer, sehingga dapat digunakan untuk memberitahu komputer tentang apa yang harus dilakukan.

Klasifikasi Bahasa Pemrograman

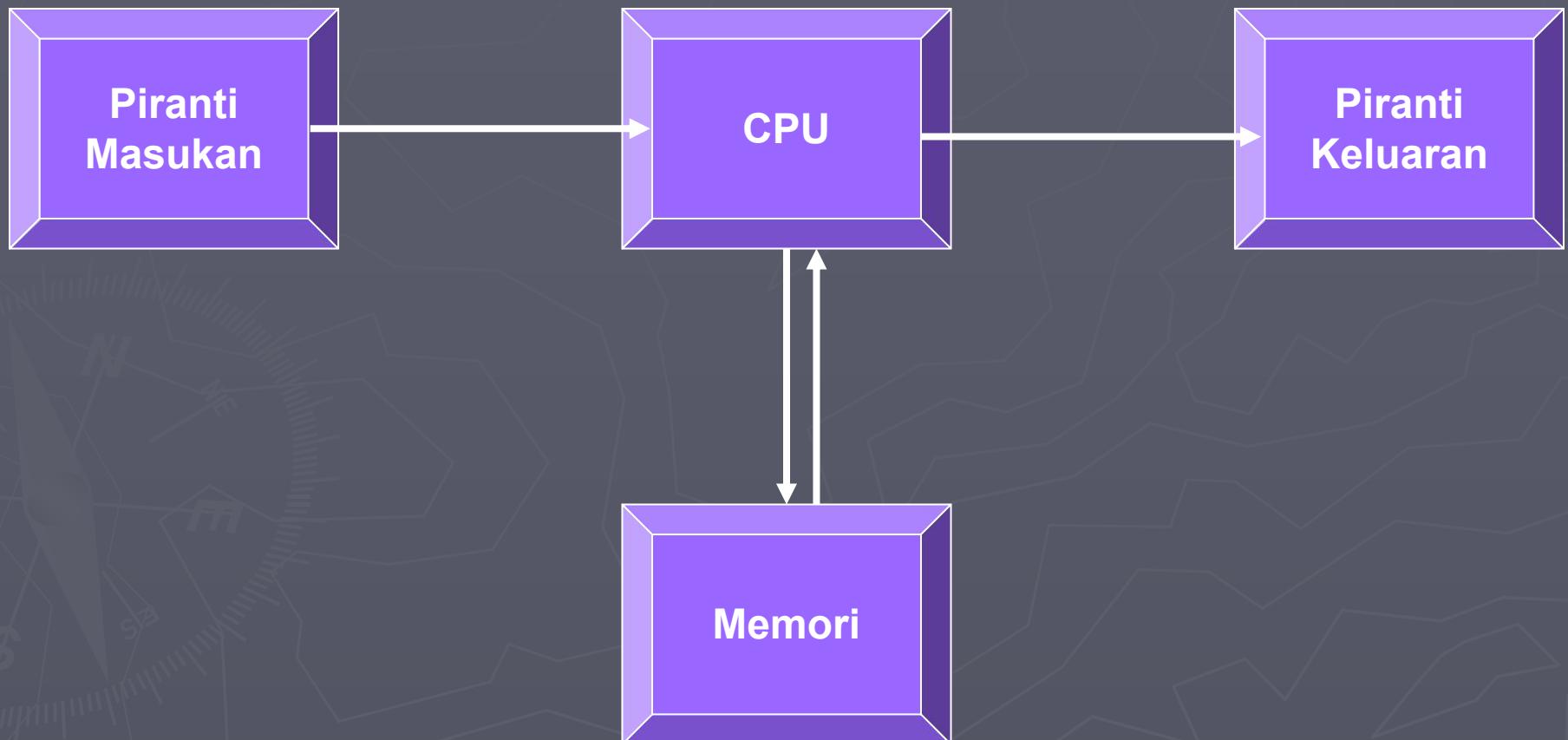
► Menurut Generasi

- First Generation Language (1GL), kode mesin
- Second Generation Language (2GL), bahasa assembly
- Generasi Ketiga, C,C++,Pascal,Java
- Generasi Keempat, SQL,PL/SQL,ABAP
- Generasi Kelima, Prolog,LISP → AI

► Menurut Tingkatan

- Low-level programming language, 1GL & 2GL
- High-level programming language (HLL), 3GL
- Very High-level programming language (VHLL), 4GL

Bagaimana Algoritma dapat menjadi program?



Belajar Memrogram Vs Belajar Bahasa Pemrograman

- ▶ **Belajar memrogram** adalah belajar tentang metodologi pemecahan masalah, kemudian menuangkannya dalam suatu notasi tertentu yang mudah dibaca dan dipahami.
- ▶ **Belajar bahasa pemrograman** berarti belajar memakai suatu bahasa, aturan-aturan tata bahasanya, instruksi-instruksinya, tata cara pengoperasian *compiler*-nya, dan memanfaatkan instruksi-instruksi tersebut untuk membuat program yang ditulis hanya dalam bahasa itu saja.

Belajar Memprogram

- ▶ belajar bahasa pemrograman
- ▶ belajar tentang strategi pemecahan masalah, metodologi dan sistematika pemecahan masalah kemudian menuliskannya dalam notasi yang disepakati bersama
- ▶ bersifat pemahaman persoalan, analisis dan sintesis
- ▶ titik berat : designer program

Belajar Bahasa Pemrograman

- ▶ belajar memakai suatu bahasa pemrograman, aturan sintaks, tatacara untuk memanfaatkan instruksi yang spesifik untuk setiap bahasa
- ▶ titik berat : coder

Bahasa Pemrograman

Berdasarkan kegunaannya :

- ▶ Bahasa pemrograman bertujuan khusus (specific purpose programming language). Yang termasuk kelompok ini adalah Cobol (untuk terapan bisnis dan administrasi), Fortran (aplikasi komputasi ilmiah), bahasa assembly (aplikasi pemrograman mesin), Prolog (aplikasi kecerdasan buatan), bahasa-bahasa simulasi, dan sebagainya.
- ▶ Bahasa pemrograman bertujuan umum (general purpose programming language) yang dapat digunakan untuk berbagai aplikasi. Yang termasuk kelompok ini adalah bahasa Pascal, Basic, C, C++, dsb.

Bahasa Pemrograman

Berdasarkan tingkatannya :

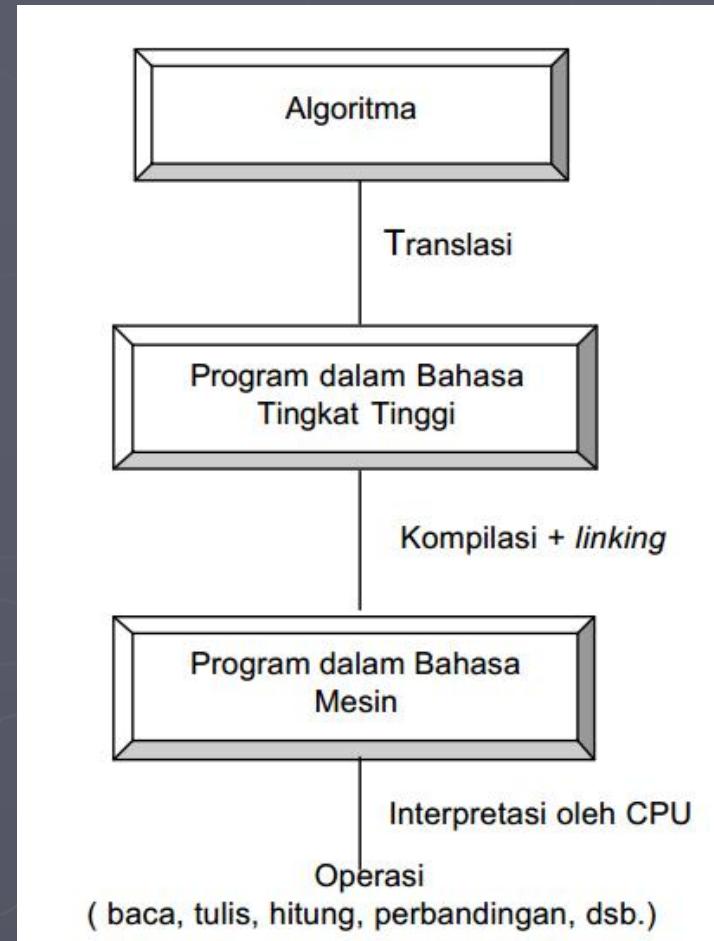
- ▶ **Bahasa tingkat rendah.** Bahasa jenis ini dirancang agar setiap instruksi langsung dikerjakan oleh komputer, tanpa harus melalui penerjemah (translator). Contohnya adalah bahasa mesin (machine language). Bahasa mesin adalah sekumpulan kode biner (0 dan 1). Setiap perintah dalam bahasa mesin langsung “dimengerti” oleh mesin dan langsung dikerjakan. Bahasa tingkat rendah bersifat primitif, sangat sederhana, dan relatif sulit dipahami manusia. Bahasa assembly dimasukkan ke dalam kelompok ini karena notasi yang dipakai dalam bahasa ini merupakan bentuk “manusiawi” dari bahasa mesin, dan untuk melaksanakan instruksi masih diperlukan penerjemahan (oleh assembler) ke dalam bahasa mesin. Bahasa tingkat rendah merupakan bahasa pemrograman generasi pertama yang pernah ditulis orang.

Bahasa Pemrograman

Berdasarkan tingkatannya :

- ▶ **Bahasa tingkat tinggi.** Bahasa jenis ini membuat program menjadi lebih mudah dipahami, lebih “manusiawi”, dan lebih dekat ke bahasa manusia (bahasa Inggris terutama). Kelemahannya, program dalam bahasa tingkat tinggi tidak dapat langsung dilaksanakan oleh komputer. Ia perlu diterjemahkan terlebih dahulu oleh sebuah translator bahasa (yang disebut kompilator atau compiler) ke dalam bahasa mesin sebelum akhirnya dieksekusi oleh CPU. Contoh bahasa tingkat tinggi adalah Pascal, Cobol, Basic, Fortran, C, C++, dan sebagainya.

-SDAP 2019-



Notasi Algoritmik

- ▶ Notasi algoritmik bukan notasi bahasa pemrograman
- ▶ Independen dari spesifikasi bahasa pemrograman
- ▶ Mudah dibaca dan dimengerti
- ▶ Dapat diterjemahkan ke dalam berbagai bahasa pemrograman

Notasi Algoritmik

Notasi I : Menyatakan langkah-langkah algoritma dengan untaian kalimat deskriptif

PROGRAM Tukar_Isi

Diberikan dua buah ember, A dan B; ember A berisi air berwarna merah, ember B berisi air berwarna biru. Pertukarkan isi kedua ember itu sedemikian sehingga ember A berisi air berwarna biru dan ember B berisi air berwarna merah.

ALGORITMA:

1. Tuangkan air dari ember A ke dalam ember C.
2. Tuangkan air dari ember B ke dalam ember A.
3. Tuangkan air dari ember C ke dalam ember B.

Notasi Algoritmik

Notasi I : Menyatakan langkah-langkah algoritma dengan untaian kalimat deskriptif

PROGRAM Euclidean

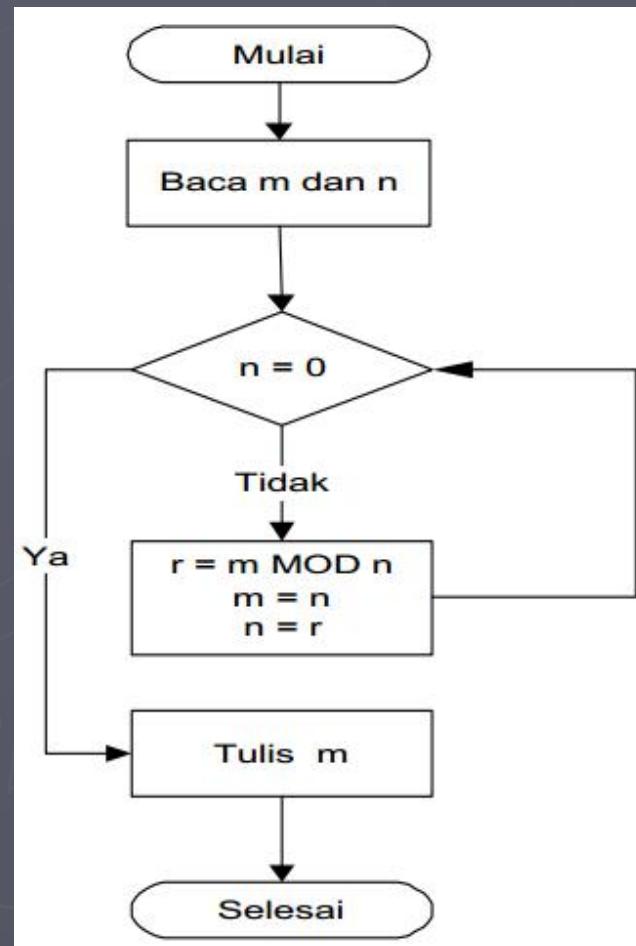
Diberikan dua buah bilangan bulat tak-negatif m dan n ($m \geq n$). Algoritma Euclidean mencari pembagi bersama terbesar, gcd, dari kedua bilangan tersebut, yaitu bilangan bulat positif terbesar yang habis membagi m dan n .

ALGORITMA:

1. Jika $n = 0$ maka
 m adalah jawabannya;
stop.
tetapi jika $n \neq 0$,
lanjutkan ke langkah 2.
2. Bagilah m dengan n dan misalkan r adalah sisanya.
3. Ganti nilai m dengan nilai n dan nilai n dengan nilai r , lalu ulang kembali ke langkah 1.

Notasi Algoritmik

Notasi II : Diagram alir (*Flowchart*)



Notasi Algoritmik

Notasi III : Menggunakan *Pseudocode*

PROGRAM Euclidean

Program untuk mencari gcd dari dua buah bilangan bulat tak-negatif m dan n ($m \geq n$). gcd dari m dan n adalah bilangan bulat positif terbesar yang habis membagi m dan n .

DEKLARASI:

```
m, n : integer      { bilangan bulat yang akan dicari pbt-nya}
r     : integer      { sisa hasil bagi }
```

ALGORITMA:

```
read(m,n)          {  $m \geq n$ }
while n ≠ 0 do
    r ← m MOD n    { hitung sisa hasil pembagian }
    m ← n
    n ← r
endwhile
{ kondisi selesai pengulangan: n = 0, maka gcd(m,n) = m }

write(m)
```

Kasus

- ▶ Algoritma untuk menentukan apakah suatu bilangan merupakan bilangan ganjil atau bilangan genap.

Algoritmanya :

- a. Masukkan bilangan yang akan ditentukan
- b. Bagi bilangan dengan bilangan 2
- c. Hitung sisa hasil bagi pada langkah b.
- d. Bila sisa hasil bagi sama dengan 0 maka bilangan itu adalah bilangan genap tetapi bila sisa hasil bagi sama dengan 1 maka bilangan itu adalah bilangan ganjil.

Kasus

- ▶ Pemimpin sebuah perusahaan otomotif perlu menentukan besarnya bonus yang akan diberikan kepada para pegawainya yang bekerja sebagai *account executive*. Jika terdapat pegawai yang dalam bulan ini telah menjual mobil lebih dari dua unit, maka akan mendapatkan bonus sebesar Rp 1.000.000,- kemudian pegawai yang bisa menjual mobil tepat dua buah maka, akan mendapatkan bonus Rp 500.000,- namun jika pegawai yang dalam bulan ini penjualannya kurang dari dua unit maka, pegawai tersebut tidak mendapatkan bonus.

Kasus

- ▶ Misalkan seorang pemuda tiba di tepi sebuah sungai. Pemuda tersebut membawa seekor kambing, seekor srigala, dan sekeranjang sayur. Mereka bermaksud hendak menyeberangi sungai. Pemuda itu menemukan sebuah perahu kecil yang hanya dapat memuat satu bawaannya setiap kali menyeberang. Situasinya dipersulit dengan kenyataan bahwa srigala tidak dapat ditinggal berdua dengan kambing (karena srigala akan memangsa kambing) atau kambing tidak dapat ditinggal berdua dengan sekeranjang sayur (karena kambing akan memakan sayur). Buatlah algoritma untuk membantu pemuda menyeberangkan seluruh bawaannya itu sehingga mereka sampai ke seberang sungai dengan selamat.