PRESENTATION 8 · LEARNING WITH PYTHON

Two of each

Exactly two iron, two silver and two gold, counted by the simulator’s own built-in variables.

TheBotCup 2026 · AiBotCup Simulator



RECAP · PRESENTATION 7

We made our own counters

1 · Make

Top of the program → Iron, Silver, Gold = 0

2 · Read

In a condition: **and Iron < 2**

3 · Change

In a branch: **Iron = Iron + 1**

Result: 2 + 2 + 2, **190 points in 1:15**. Today the same idea, with less work.

PART 1

The mission: Session 8

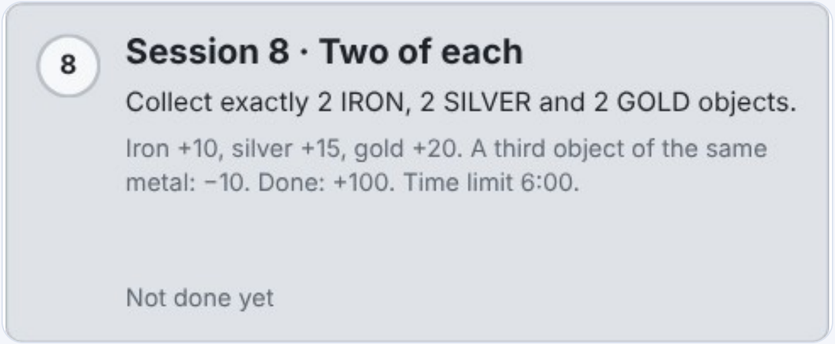
Exactly two of every metal

SESSION 8 · TWO OF EACH

2 iron, 2 silver, 2 gold. Not one more.

What happens	Points
Iron · silver · gold	+10 · +15 · +20
Two of each: done	+100
A third of the same metal	−10, refused

Time limit **6:00**.





THE FIELD

Six of each metal

- **6 iron** · black
- **6 silver** · grey
- **6 gold** · yellow

You need 2 of each: 12 objects must be **left behind**.

WE TESTED IT · NO COUNTING

Without counters: two penalties

What it did

It tried to collect a third of a metal **twice**. Both were refused: **−10, −10**.

Score

170 in 1:21, instead of 190. Each refused pickup also wasted about 3 seconds.

So we need counters again. This time we do not even have to make them.

PART 2

The simulator counts too

Built-in variables

BUILT-IN VARIABLES

Ready to use, nothing to make

Name	Means	In the panel
LoadedObjects	objects on the robot now (0–6)	Load 3/6
LoadedIron	iron on the robot	Fe
LoadedSilver	silver on the robot	Ag
LoadedGold	gold on the robot	Au

Fe, Ag and Au are the chemistry symbols for iron, silver and gold. You can **read** these, but not change them: the simulator does that.

SENSORS PANEL · ROBOT SECTION

They count only what is really on board

Robot

Wheels L **0.0** · R **0.0** · LED **2**
Load **3**/6 · Fe 1 Ag 2 Au 0
On **collecting 1.3 / 3 s**

35.8 s: collecting the 2nd iron, still **Fe 1**.

Robot

Wheels L **0.0** · R **0.0** · LED **2**
Load **5**/6 · Fe 2 Ag 2 Au 1
On **collecting 1.5 / 3 s**

73.4 s: collecting the 2nd gold, still **Au 1**.

The number goes up only **after** the 3-second collect has finished.

OWN COUNTER OR BUILT-IN?

Same rule, different way of counting

	Iron (P7, own)	LoadedIron (built-in)
Made by	you, at the top of the program	the simulator
Goes up	when collect() **starts**	when the object is **on board**
If a collect fails	would still go up	stays the same
Extra work	Iron = Iron + 1 in 6 branches	none

Own variables are still important: use them for anything the simulator does **not** count for you.

PART 3

Step by step in Python

Eight branches, zero own variables

STEP 1 · THE PLAN

Same shape as P7, built-in counters

Branch	Condition	Then
1 IronLeft	isIron(left) and LoadedIron < 2	collect()
2 IronRight	isIron(right) and LoadedIron < 2	collect()
3 SilverLeft	isSilver(left) and LoadedSilver < 2	collect()
4 SilverRight	isSilver(right) and LoadedSilver < 2	collect()
5 GoldLeft	isGold(left) and LoadedGold < 2	collect()
6 GoldRight	isGold(right) and LoadedGold < 2	collect()
7 Ahead	US_Front <= 20	−2 / 2, 700 ms
8 Forward	else	4.5 / 4.5

STEP 2 · LINES 3–16

Helpers and collect(), as in P7

```
 3  def isIron(R, G, B):
 4      return R <= 60 and G <= 60 and B <= 60
 5
 6  def isSilver(R, G, B):
 7      return 90 <= R <= 135 and 92 <= G <= 137 and 98 <= B <= 142
 8
 9  def isGold(R, G, B):
10      return R >= 195 and 135 <= G <= 200 and B <= 80
11
12  def collect():
13      WheelLeft = 0; WheelRight = 0
14      LED_1 = 2                               # flashing
15      LED_R = 255; LED_G = 0; LED_B = 0       # red
16      Duration = 3200                         # hold 3.2 seconds
```

No counters at the top this time.

isIron, isSilver, isGold: the same colour ranges.

collect(): wheels 0, red LED, 3.2 s.

STEP 3 · LINES 18–31

Six collect branches, built-in counters

```
18  def game1():
19      LED_1 = 0
20      if isIron(CSLeft_R, CSLeft_G, CSLeft_B) and LoadedIron < 2:           # IronLeft
21          collect()
22      elif isIron(CSRight_R, CSRight_G, CSRight_B) and LoadedIron < 2:      # IronRight
23          collect()
24      elif isSilver(CSLeft_R, CSLeft_G, CSLeft_B) and LoadedSilver < 2:     # SilverLeft
25          collect()
26      elif isSilver(CSRight_R, CSRight_G, CSRight_B) and LoadedSilver < 2:  # SilverRight
27          collect()
28      elif isGold(CSLeft_R, CSLeft_G, CSLeft_B) and LoadedGold < 2:         # GoldLeft
29          collect()
30      elif isGold(CSRight_R, CSRight_G, CSRight_B) and LoadedGold < 2:      # GoldRight
31          collect()
```

LoadedIron < 2: type it exactly, capital L and I.

Nothing after collect(): the simulator counts.

STEP 4 · LINES 32–36

Ahead and Forward, unchanged

```
32      elif US_Front <= 20:                                    # Ahead
33          WheelLeft = -2; WheelRight = 2
34          Duration = 700
35      else:                                                   # Forward
36          WheelLeft = 4.5; WheelRight = 4.5
```

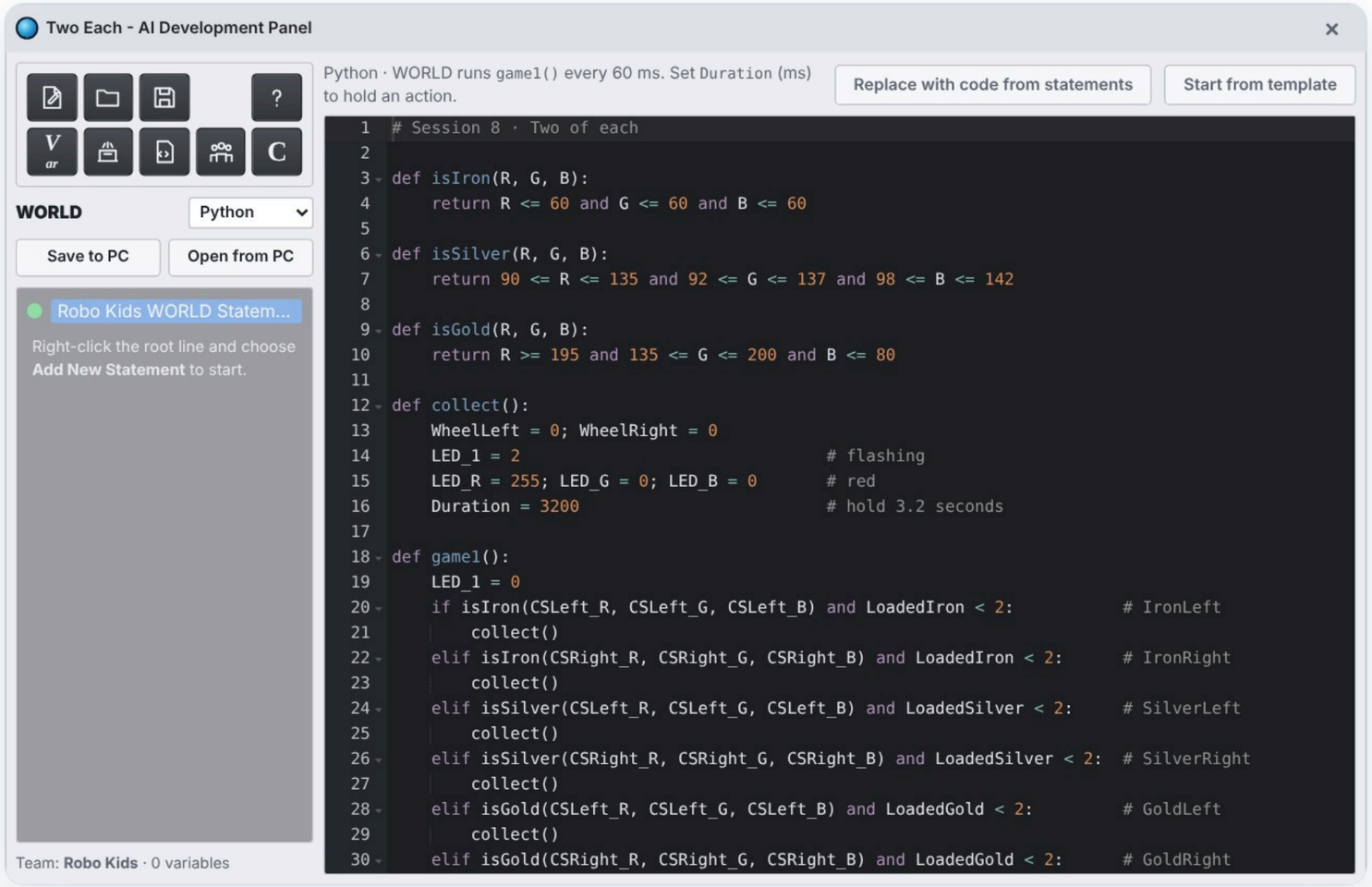
Collect branches stay on top.

A metal that is already full matches no branch above, so **Forward drives over it**.

STEP 5 · BUILD AND LOAD

The whole program, built

1. WORLD menu: **Python**.
2. Type the program, name it **Two Each**.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.





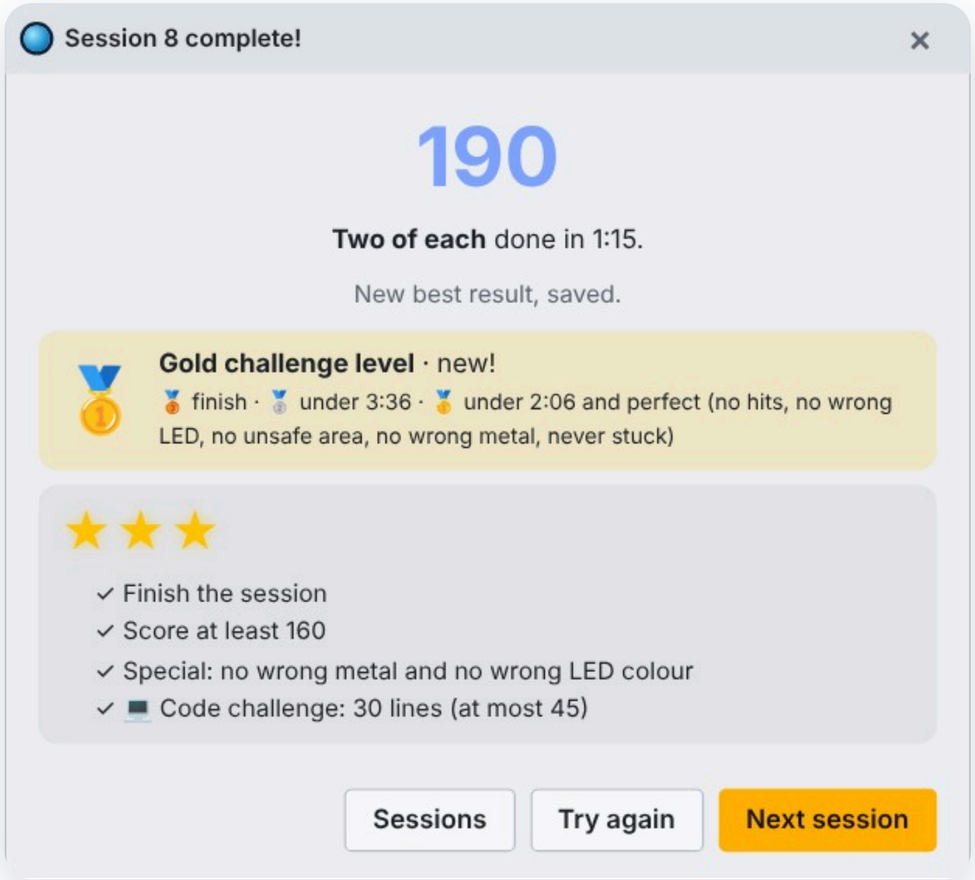
STEP 6 · RUN

The 2nd silver at 31.8 s

CSLeft reads **108, 110, 119**: isSilver is true.

LoadedSilver is 1, so **LoadedSilver < 2** is true: SilverLeft runs.

After the 3.2 s collect the simulator makes it **2**.



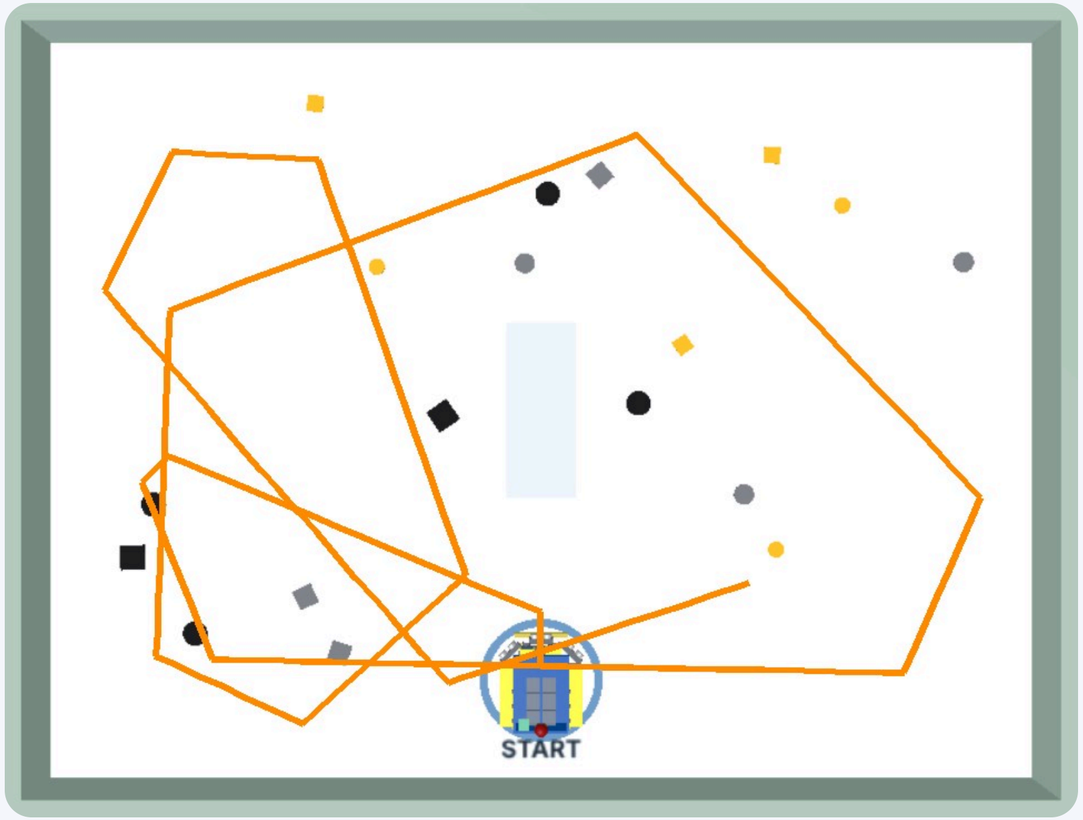
RESULT · TESTED

190 points in 1:15 · Gold

Score · 2×10 + 2×15 + 2×20 = 90, plus 100 = 190. No penalties.

Medals · Silver: under 3:36 · Gold: under 2:06 and perfect.

Code challenge · 30 lines, at most 45.



THE PATH · RECORDED

Same bounces, smarter choices

Order in our run: iron, silver, silver, iron, gold, gold.

Compared with no counting: **+20 points** and 6 s faster.

Built-in counters: **less code, always right**.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Build error with a line number	Colon : after def / if / elif / else, and indentation
Error about a name	Spelling: **LoadedIron**, **LoadedSilver**, **LoadedGold**
A third of one metal collected	That branch's **and Loaded... < 2** is missing or names the wrong metal
Never collects silver or gold	Check the R, G, B ranges in isSilver and isGold
Old program still on the robot	Build again, then Load on BLUE

NEXT TIME

Presentation 9 · Collect and deposit

Fill the robot with **LoadedObjects**, then empty it in the green deposit area.