PRESENTATION 9 · LEARNING WITH C

Collect and deposit

Fill the robot with six objects, then empty it in a green deposit area for double points.

TheBotCup 2026 · AiBotCup Simulator



RECAP · PRESENTATION 8

The simulator counts what is on board

LoadedObjects

How many objects the robot carries: 0 to 6. **Today's star.**

LoadedIron · Silver · Gold

How many of each metal. Used in Presentation 8.

They go up only when an object is really on board, and **down to 0** when the robot deposits.

PART 1

The mission: Session 9

Collect six, then deposit six

SESSION 9 · COLLECT AND DEPOSIT

Any metal, six of them, then unload

What happens	Points
Collect an object	its value
Deposit it	its value again
6 deposited	+150

Time limit **7:00**. Bonus star: deposit everything in **one trip**.





THE FIELD

Two green deposit areas

One in the **top-left** corner, one in the **top-right**. Either one works.

6 iron, 6 silver and 6 gold: **any six** will do.

Block in the middle, START at the bottom, as before.

HOW TO DEPOSIT

Collect’s twin: a deposit() function

Both sensors

Left **and** right colour sensors must be on the green.

Deposit

Wheels stop, LED **steady green** (not flashing) for 3 s.

Unloaded

Every object scores again. LoadedObjects goes back to **0**.

Collect: **red, flashing** · Deposit: **green, steady**. Our deposit() sets this.

COLOUR SENSORS · GREEN

What does green look like in numbers?

	R	G	B
Deposit area	30	160	70
Our test, left sensor	31	159	67
Range we use	0 – 75	130 – 200	40 – 110

Low red, high green, low blue. No metal and not the floor fits all three ranges. The range comes from the simulator’s own tip.

PART 2

The plan

Two jobs, one switch

THE PLAN

First fill up, then empty out

JOB 1 · while LoadedObjects < 6

Collect any metal

Bounce round the field like Presentation 6.

JOB 2 · when LoadedObjects == 6

Deposit on green

Keep bouncing until both sensors see green.

One built-in number switches the robot from job 1 to job 2. No variables to make.

COMPARING NUMBERS

< 6 and == 6

`LoadedObjects < 6`

“less than 6”: true for 0, 1, 2, 3, 4, 5. On every **collect** branch.

`LoadedObjects == 6`

“is equal to 6”: true only when full. On the **deposit** branch.

== (two signs) **tests** if equal. **=** (one sign) **changes** a variable, like Iron = Iron + 1.

PART 3

Step by step in C

Nine branches

STEP 1 · THE PLAN

Deposit on top, then collect, then move

Branch	Condition	Then
1 Deposit	bothGreen() && LoadedObjects == 6	deposit()
2 IronLeft	isIron(left) && LoadedObjects < 6	collect()
3 IronRight	isIron(right) && LoadedObjects < 6	collect()
4 SilverLeft	isSilver(left) && LoadedObjects < 6	collect()
5 SilverRight	isSilver(right) && LoadedObjects < 6	collect()
6 GoldLeft	isGold(left) && LoadedObjects < 6	collect()
7 GoldRight	isGold(right) && LoadedObjects < 6	collect()
8 Ahead	US_Front <= 20	−2 / 2, 700 ms
9 Forward	else	4.5 / 4.5

STEP 2 · LINES 15–35

Green test and deposit()

```
15  int isGreen(int R, int G, int B) {
16      return R <= 75 && G >= 130 && G <= 200 && B >= 40 && B <= 110;
17  }
18
19  int bothGreen() {
20      return isGreen(CSLeft_R, CSLeft_G, CSLeft_B) && isGreen(CSRight_R, CSRight_G, CSRight_B);
21  }
22
23  void collect() {
24      WheelLeft = 0; WheelRight = 0;
25      LED_1 = 2;                              // flashing
26      LED_R = 255; LED_G = 0; LED_B = 0;      // red
27      Duration = 3200;                        // hold 3.2 seconds
28  }
29
30  void deposit() {
31      WheelLeft = 0; WheelRight = 0;
32      LED_1 = 1;                              // steady
33      LED_R = 0; LED_G = 255; LED_B = 0;      // green
34      Duration = 3200;
35  }
```

isGreen: R 0–75, G 130–200, B 40–110.

bothGreen(): left && right on green.

deposit(): wheels 0, LED steady green, 3.2 s.

STEP 3 · LINES 37–52

Deposit first, then six collects

```
37  void Game1() {
38      LED_1 = 0;
39      if (bothGreen() && LoadedObjects == 6) {                                  // Deposit
40          deposit();
41      } else if (isIron(CSLeft_R, CSLeft_G, CSLeft_B) && LoadedObjects < 6) {   // IronLeft
42          collect();
43      } else if (isIron(CSRight_R, CSRight_G, CSRight_B) && LoadedObjects < 6) { // IronRight
44          collect();
45      } else if (isSilver(CSLeft_R, CSLeft_G, CSLeft_B) && LoadedObjects < 6) { // SilverLeft
46          collect();
47      } else if (isSilver(CSRight_R, CSRight_G, CSRight_B) && LoadedObjects < 6) { // SilverRight
48          collect();
49      } else if (isGold(CSLeft_R, CSLeft_G, CSLeft_B) && LoadedObjects < 6) {   // GoldLeft
50          collect();
51      } else if (isGold(CSRight_R, CSRight_G, CSRight_B) && LoadedObjects < 6) { // GoldRight
52          collect();
```

== 6: deposit only when full.

< 6: collect any metal while there is room.

STEP 4 · LINES 53–59

Ahead and Forward keep it moving

```
53      } else if (US_Front <= 20) {                                    // Ahead
54          WheelLeft = -2; WheelRight = 2;
55          Duration = 700;
56      } else {                                                        // Forward
57          WheelLeft = 4.5; WheelRight = 4.5;
58      }
59  }
```

Full or not, the robot keeps bouncing.

When full, it is simply looking for **green** instead of metal.

WHY THIS ORDER?

Each job only wakes up at the right time

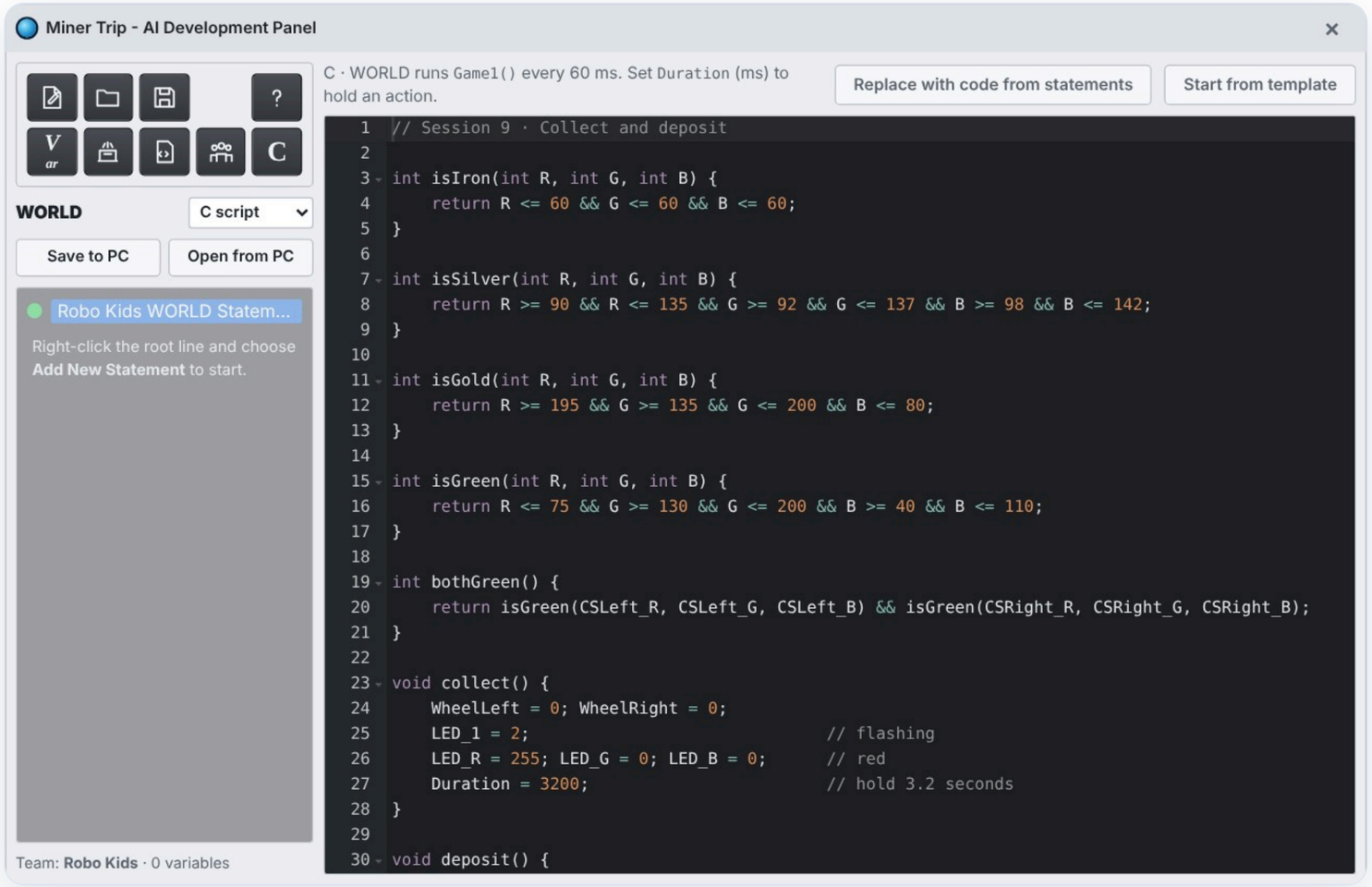
Robot carries	On green	On metal
0 – 5 objects	drives over it	**Collect**
6 objects	**Deposit**	drives over it

Deposit and collect can never match at the same time, so their order does not fight. Forward is still last.

STEP 5 · BUILD AND LOAD

The whole program, built

1. WORLD menu: **C script**.
2. Type the program, name it **Miner Trip**.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.





STEP 6 · RUN · JOB 1

The sixth object at 57.1 s

CSRight reads **25, 24, 24**: iron.

LoadedObjects is 5, so **< 6** is true: IronRight runs.

When the collect ends it is **6**: job 2 starts.



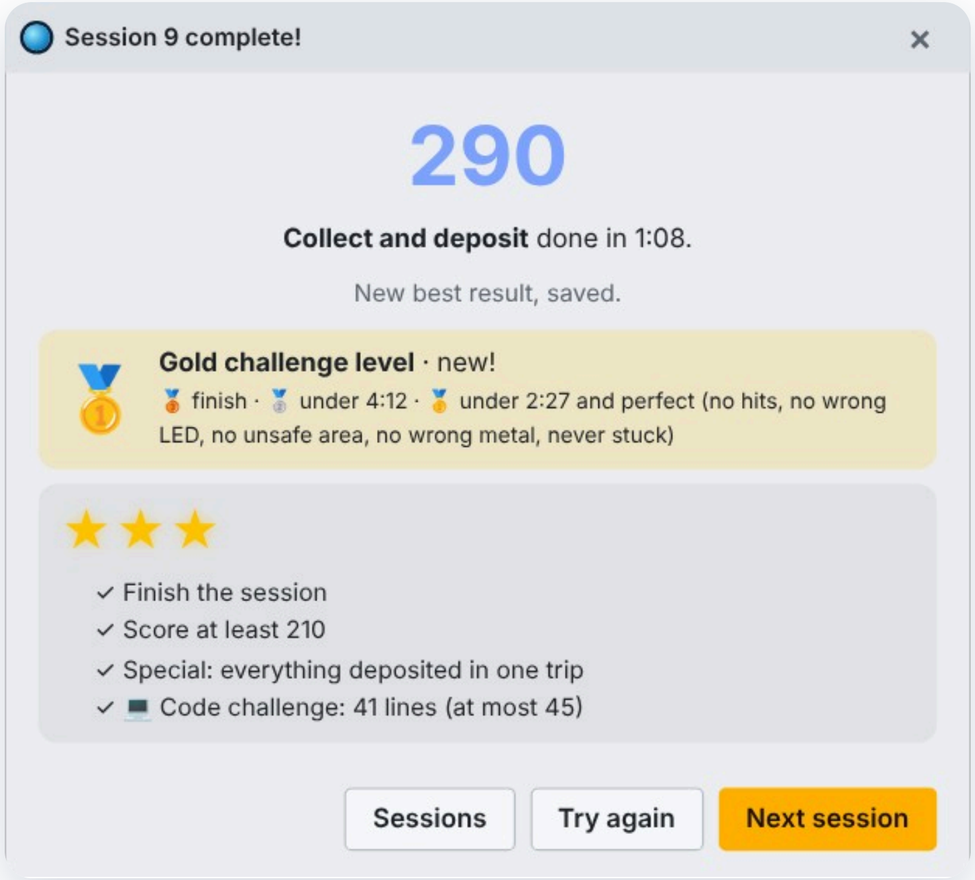
STEP 7 · RUN · JOB 2

Both on green at 64.6 s

CSLeft **30, 160, 67** · CSRight **28, 159, 69**: both green.

LoadedObjects == 6 is true: **deposit()** runs.

LED steady green for 3 s, then all six are unloaded.



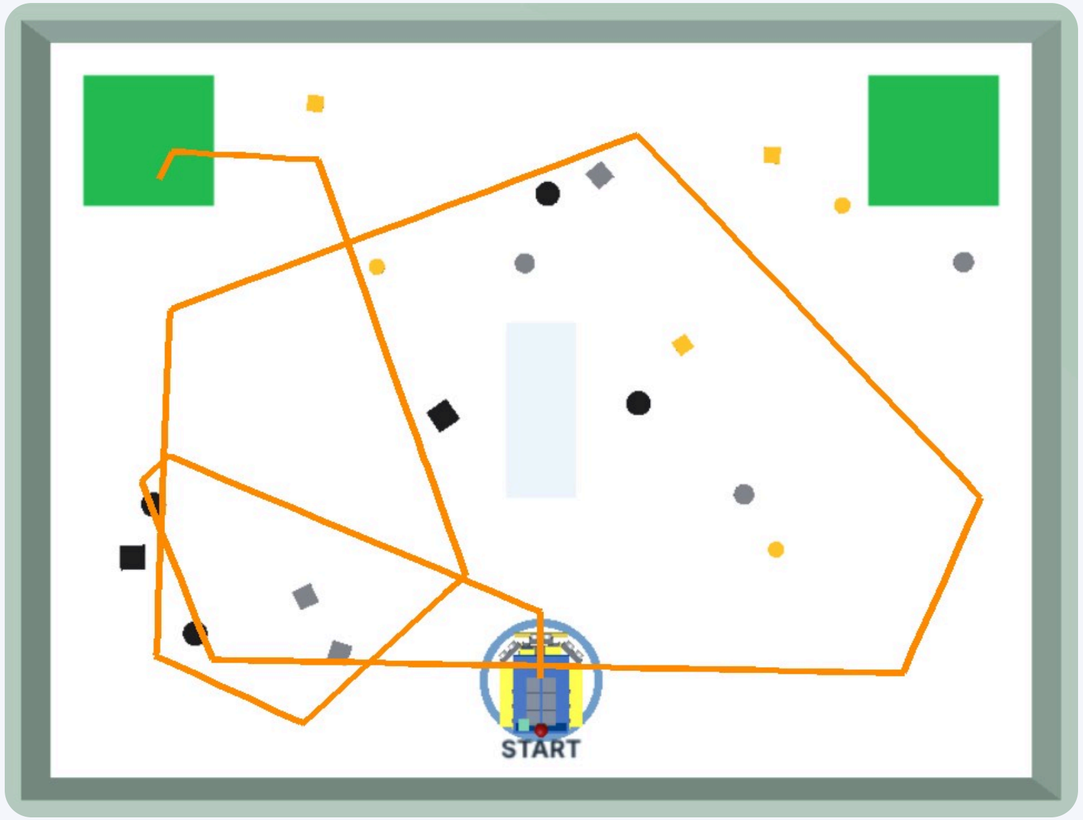
RESULT · TESTED

290 points in 1:08 · Gold

Score · collected 4×10 + 2×15 = 70, deposited 70 again, +150 = 290.

Medals · Silver: under 4:12 · Gold: under 2:27 and perfect.

Stars · deposited in one trip · 41 of at most 45 lines.



THE PATH · RECORDED

Bounce until full, bounce until green

Full at about **1:00**, depositing at **64.6 s**, done at **1:08**.

Finding green by bouncing took **luck** as well as code.

Challenge: steer straight to a corner when full.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Build error with a line number	; at the end of lines, { } matched
Drives over green when full	Ranges in isGreen; condition **LoadedObjects == 6**
Stops on green but nothing happens	Call **deposit()**, not collect(); LED green, not red
Deposits with fewer than 6	**==** 6, not < 6, on the Deposit branch
Tries to collect a 7th object	Collect branches need **&& LoadedObjects < 6**
Old program still on the robot	Build again, then Load on BLUE

NEXT TIME

Presentation 10 · Avoid the traps

Red traps with a light-red **warning band**: see it, back off, go round.