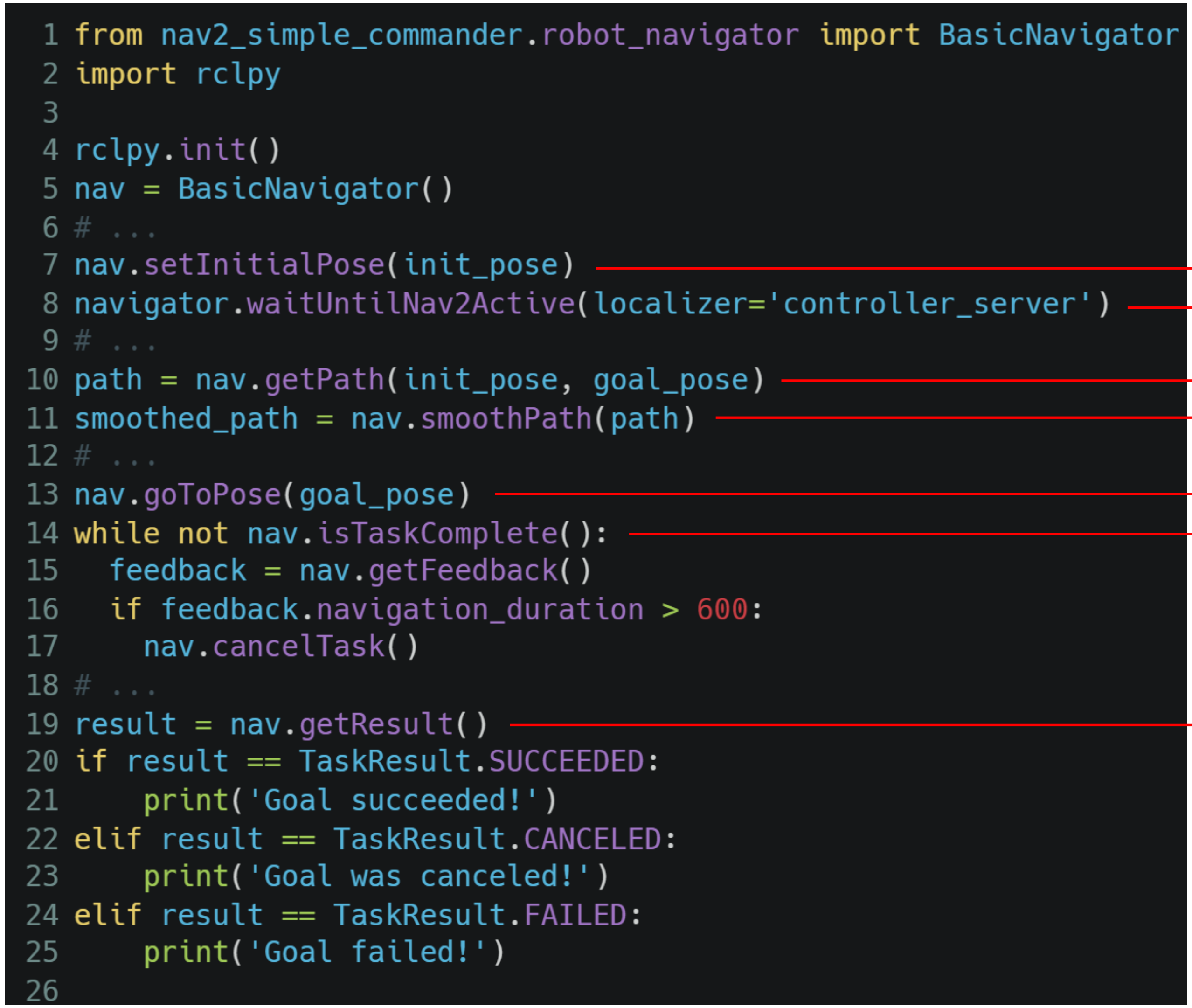
Turtlebot4 LLM

운영체제의 실제

안인규 (Inkyu An)

Nav2 – Commander API



```python
from nav2_simple_commander.robot_navigator import BasicNavigator
import rclpy

rclpy.init()
nav = BasicNavigator()
# ...
nav.setInitialPose(init_pose)
navigator.waitUntilNav2Active(localizer='controller_server')
# ...
path = nav.getPath(init_pose, goal_pose)
smoothed_path = nav.smoothPath(path)
# ...
nav.goToPose(goal_pose)
while not nav.isTaskComplete():
  feedback = nav.getFeedback()
  if feedback.navigation_duration > 600:
    nav.cancelTask()
# ...
result = nav.getResult()
if result == TaskResult.SUCCEEDED:
    print('Goal succeeded!')
elif result == TaskResult.CANCELED:
    print('Goal was canceled!')
elif result == TaskResult.FAILED:
    print('Goal failed!')

```

Localization의 초기 위치 지정

Nav2가 online이 되길 기다림

경로 반환

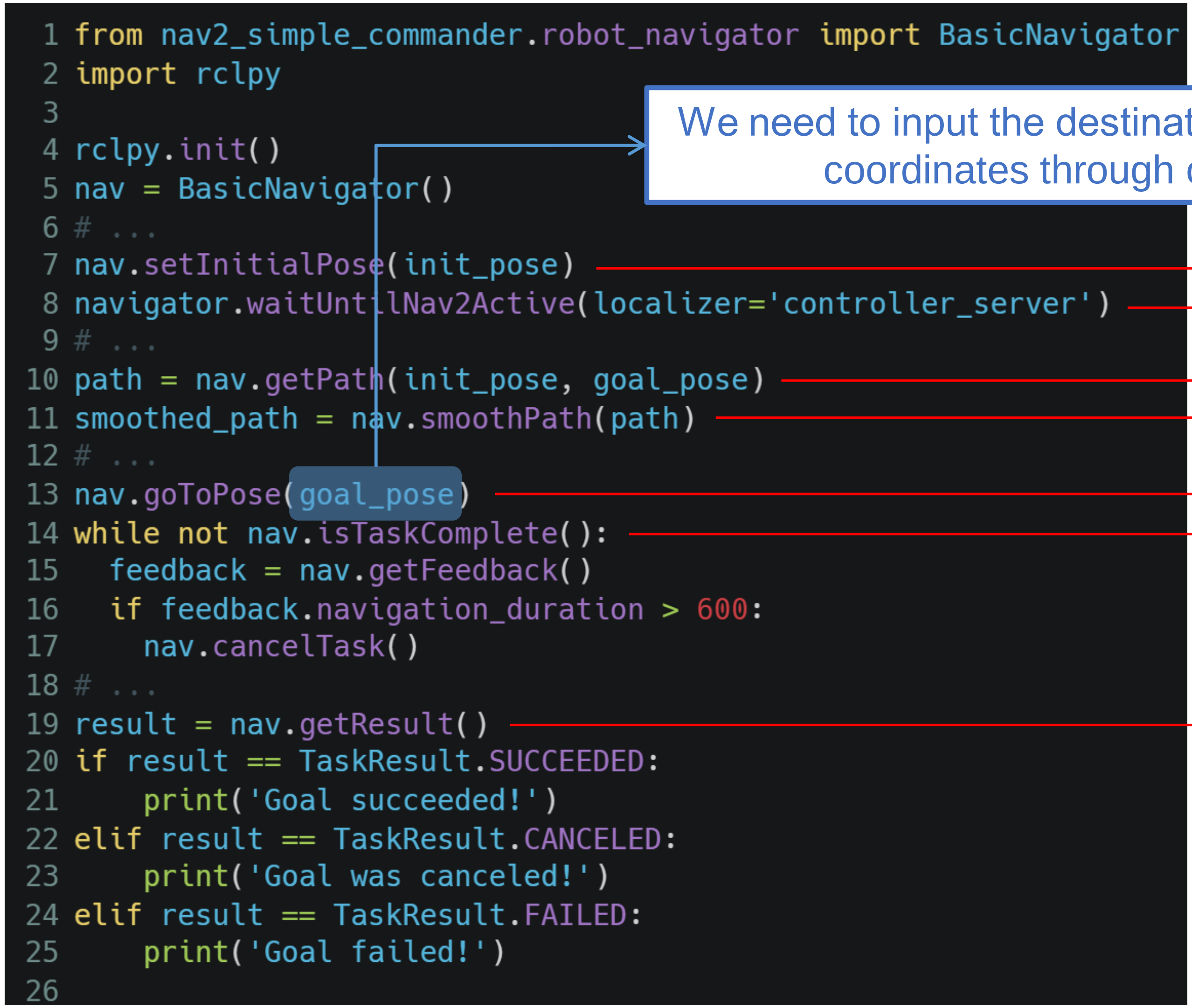
부드러운 경로 생성

목적지까지 이동 명령

Navigation task가 완료될 때 까지 기다림

결과 확인

Nav2 – Commander API



```python
from nav2_simple_commander.robot_navigator import BasicNavigator
import rclpy

rclpy.init()
nav = BasicNavigator()
# ...
nav.setInitialPose(init_pose)
navigator.waitUntilNav2Active(localizer='controller_server')
# ...
path = nav.getPath(init_pose, goal_pose)
smoothed_path = nav.smoothPath(path)
# ...
nav.goToPose(goal_pose)
while not nav.isTaskComplete():
  feedback = nav.getFeedback()
  if feedback.navigation_duration > 600:
    nav.cancelTask()
# ...
result = nav.getResult()
if result == TaskResult.SUCCEEDED:
    print('Goal succeeded!')
elif result == TaskResult.CANCELED:
    print('Goal was canceled!')
elif result == TaskResult.FAILED:
    print('Goal failed!')

```

We need to input the destination’s (x, y, z) coordinates through code

Localization의 초기 위치 지정

Nav2가 online이 되길 기다림

경로 반환

부드러운 경로 생성

목적지까지 이동 명령

Navigation task가 완료될 때 까지 기다림

결과 확인

Command using Natural Language

- Isn’t it possible to give commands using natural language? (e.g., Go to the toilet!)
  - In traditional navigation methods, you have to input the exact location of the toilet.

Powered by LLM

- LLMs are very powerful tools that can understand natural language.
- LLMs are also very powerful tools that can understand computer languages (e.g., Python).

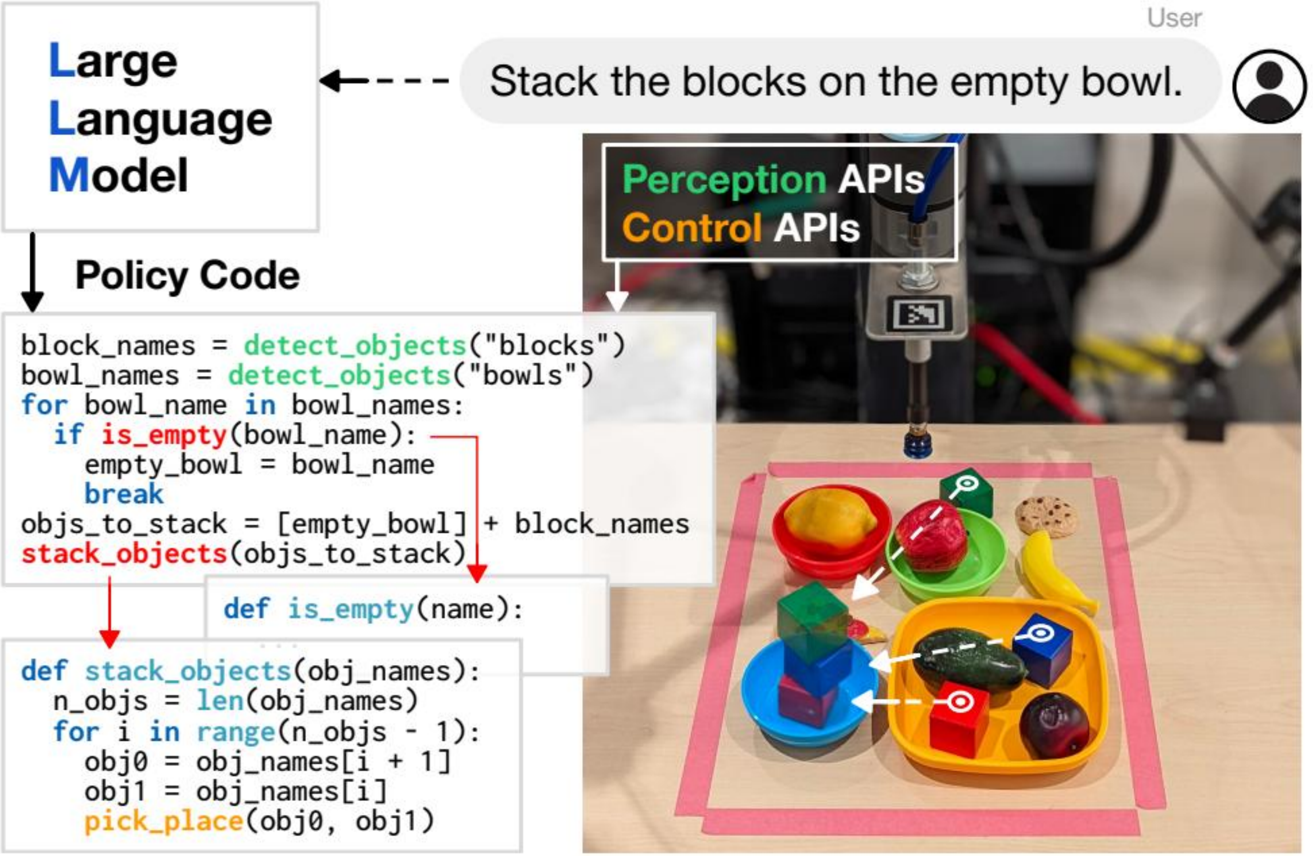
Powered by LLM

- LLMs are very powerful tools that can understand natural language.
- LLMs are also very powerful tools that can understand computer languages (e.g., Python).

- What if an LLM received natural language input and generated Python code to control a robot?
  - Code as Policies: Language Model Programs for Embodied Control, Robotics at Google, arXiv 2022

Code as Policies, Google

- Language model generated programs (LMPs)
  - Robots can use code-writing LLMs to translate natural language commands into robot policy code which process perception outputs, parameterize control primitives, recursively generate code for undefined function



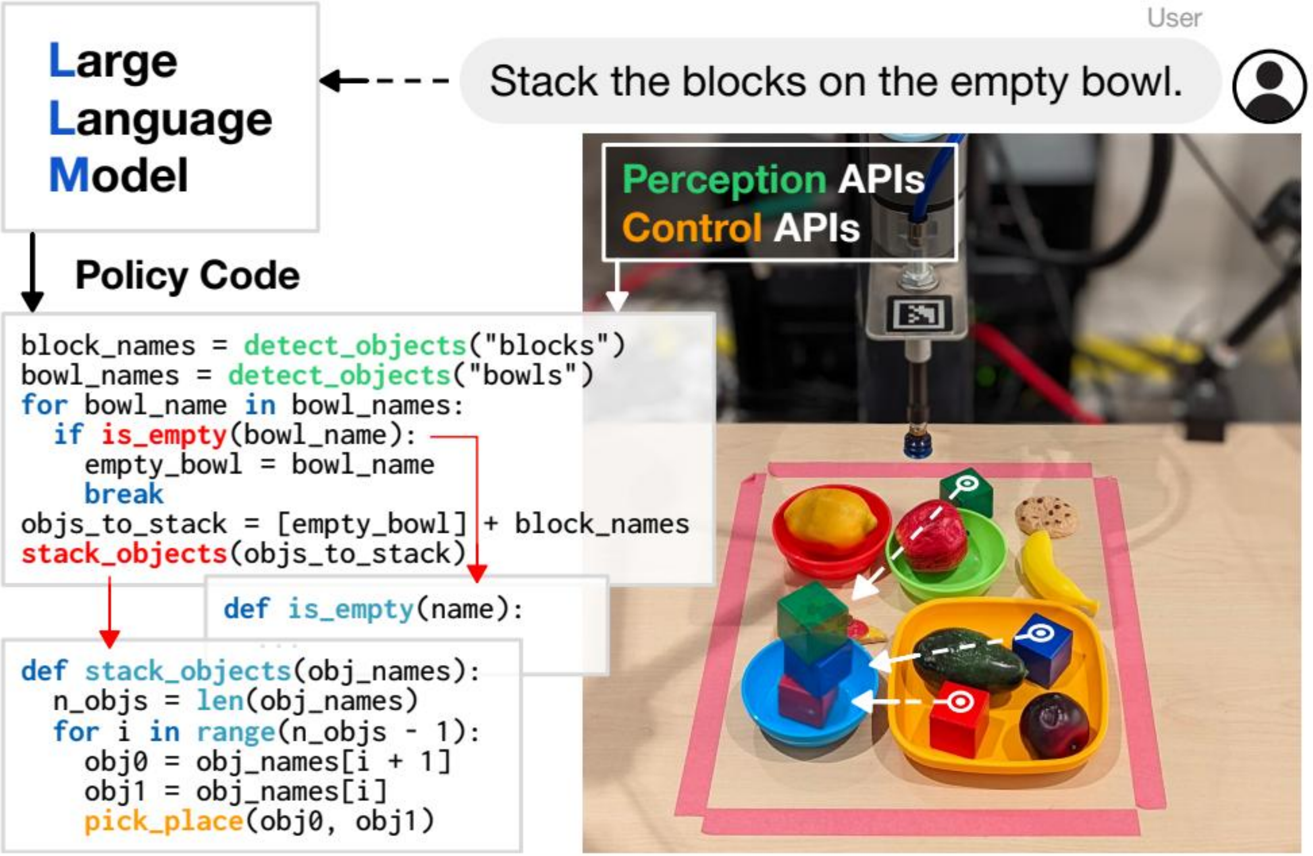
```
block_names = detect_objects("blocks")
bowl_names = detect_objects("bowls")
for bowl_name in bowl_names:
  if is_empty(bowl_name):
    empty_bowl = bowl_name
    break
objs_to_stack = [empty_bowl] + block_names
stack_objects(objs_to_stack)
```

```
def is_empty(name):
```

```
def stack_objects(obj_names):
  n_objs = len(obj_names)
  for i in range(n_objs - 1):
    obj0 = obj_names[i + 1]
    obj1 = obj_names[i]
    pick_place(obj0, obj1)
```

Code as Policies, Google

- Language model generated programs (LMPs)
    - Robots can use code-writing LLMs to translate natural language commands into robot policy code which process perception outputs, parameterize control primitives, recursively generate code for undefined function



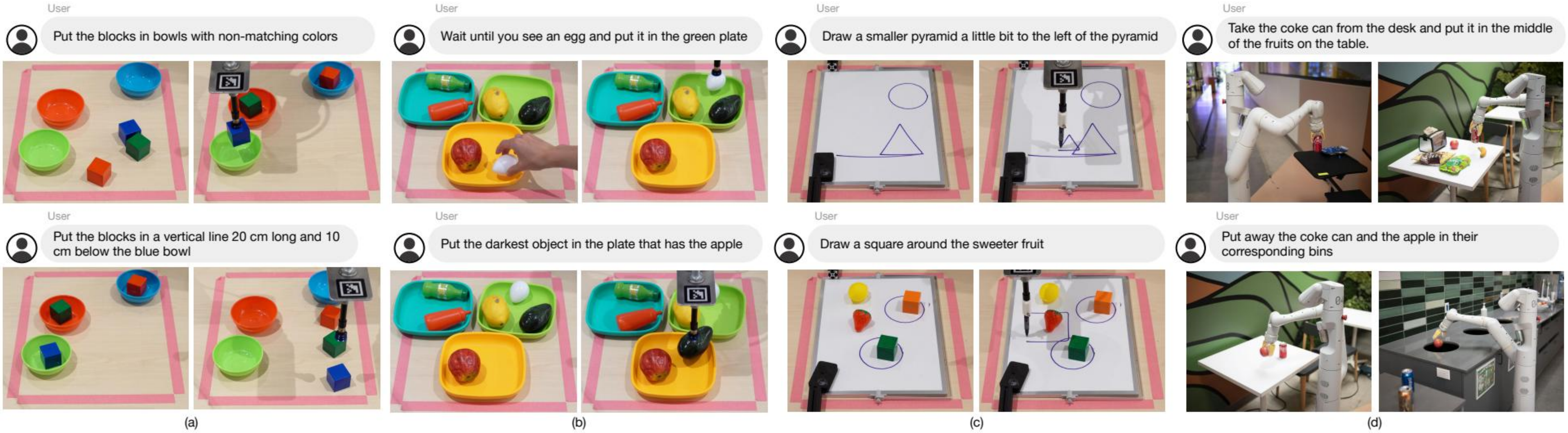
```
block_names = detect_objects("blocks")
bowl_names = detect_objects("bowls")
for bowl_name in bowl_names:
  if is_empty(bowl_name):
    empty_bowl = bowl_name
    break
objs_to_stack = [empty_bowl] + block_names
stack_objects(objs_to_stack)
```

```
def is_empty(name):
```

```
def stack_objects(obj_names):
  n_objs = len(obj_names)
  for i in range(n_objs - 1):
    obj0 = obj_names[i + 1]
    obj1 = obj_names[i]
    pick_place(obj0, obj1)
```

→ **Only can understand “Text” (=Prompt)!**

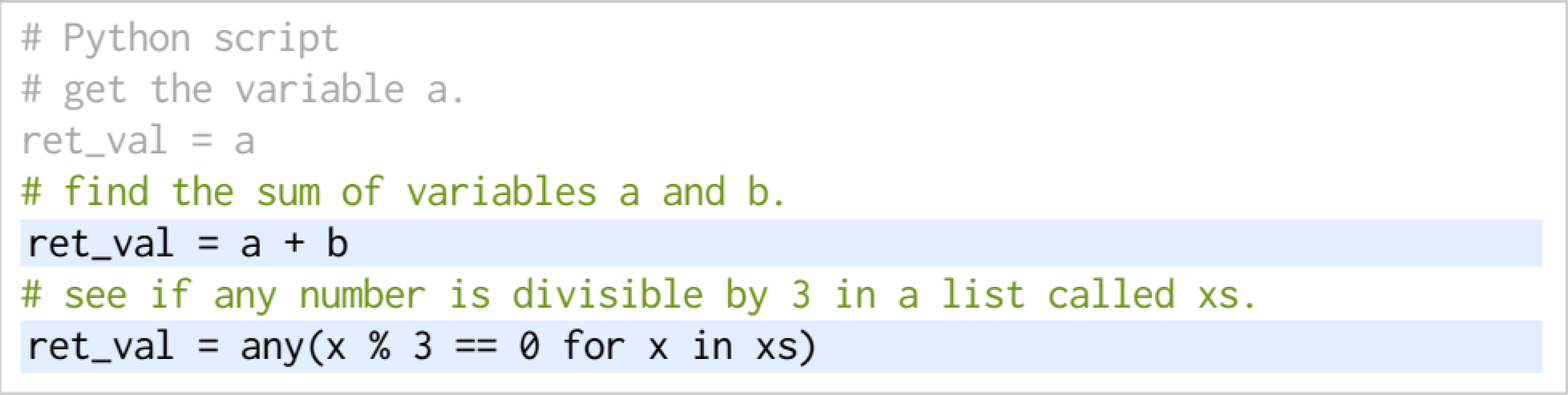
Code as Policies, Google


User
Put the blocks in bowls with non-matching colors
User
Wait until you see an egg and put it in the green plate
User
Draw a smaller pyramid a little bit to the left of the pyramid
User
Take the coke can from the desk and put it in the middle of the fruits on the table.
User
Put the blocks in a vertical line 20 cm long and 10 cm below the blue bowl
User
Put the darkest object in the plate that has the apple
User
Draw a square around the sweeter fruit
User
Put away the coke can and the apple in their corresponding bins
(a)
(b)
(c)
(d)

- Language model generated programs (LMPs)
  - **Input**: 자연어 명령 (e.g., Put the red block on the green bowl)
  - **Output**: Python code
  - **Prompt**: Hints + Examples

Code as Policies, Google

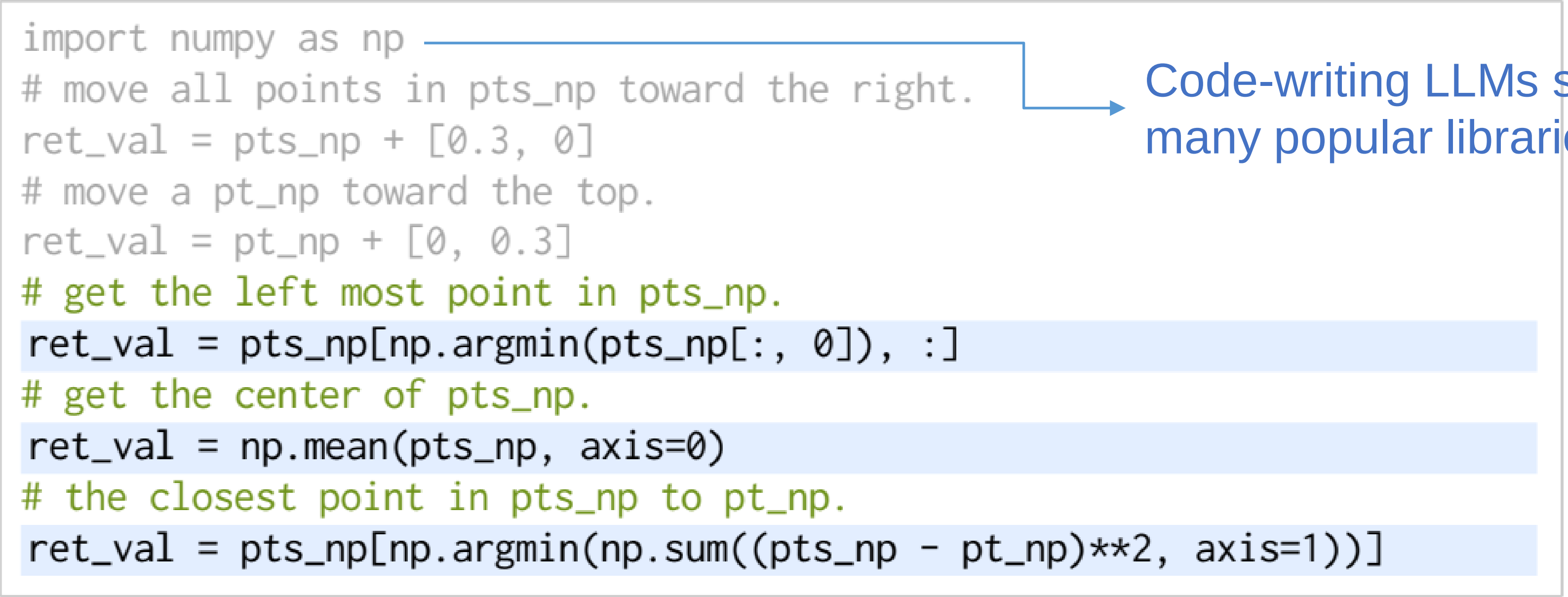
- Prompt:
  - Hints: import statements that inform the LLM which APIs are available and type hints on how to use those APIs.
  - Example:
    - Input instructions are green / Generated Outputs are highlighted



```
# Python script
# get the variable a.
ret_val = a
# find the sum of variables a and b.
ret_val = a + b
# see if any number is divisible by 3 in a list called xs.
ret_val = any(x % 3 == 0 for x in xs)
```

Code as Policies, Google

- Prompt:
  - Hints: import statements that inform the LLM which APIs are available and type hints on how to use those APIs.
  - Example:
    - Input instructions are green / Generated Outputs are highlighted



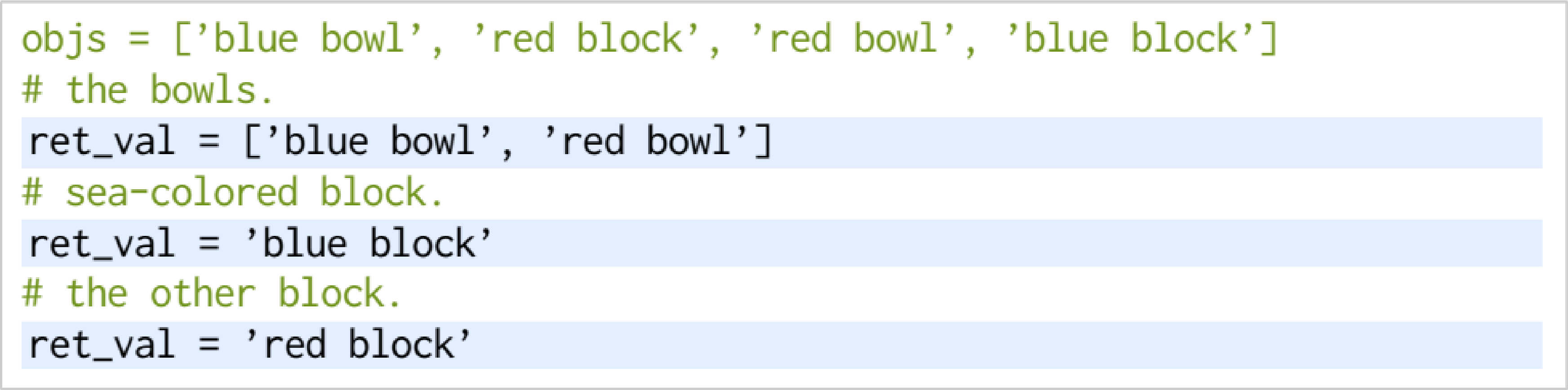
```
import numpy as np
# move all points in pts_np toward the right.
ret_val = pts_np + [0.3, 0]
# move a pt_np toward the top.
ret_val = pt_np + [0, 0.3]
# get the left most point in pts_np.
ret_val = pts_np[np.argmin(pts_np[:, 0]), :]
# get the center of pts_np.
ret_val = np.mean(pts_np, axis=0)
# the closest point in pts_np to pt_np.
ret_val = pts_np[np.argmin(np.sum((pts_np - pt_np)**2, axis=1))]
```

Code-writing LLMs store knowledge of many popular libraries

Third-party libraries

Code as Policies, Google

- Prompt:
  - Hints: import statements that inform the LLM which APIs are available and type hints on how to use those APIs.
  - Example:
    - Input instructions are green / Generated Outputs are highlighted

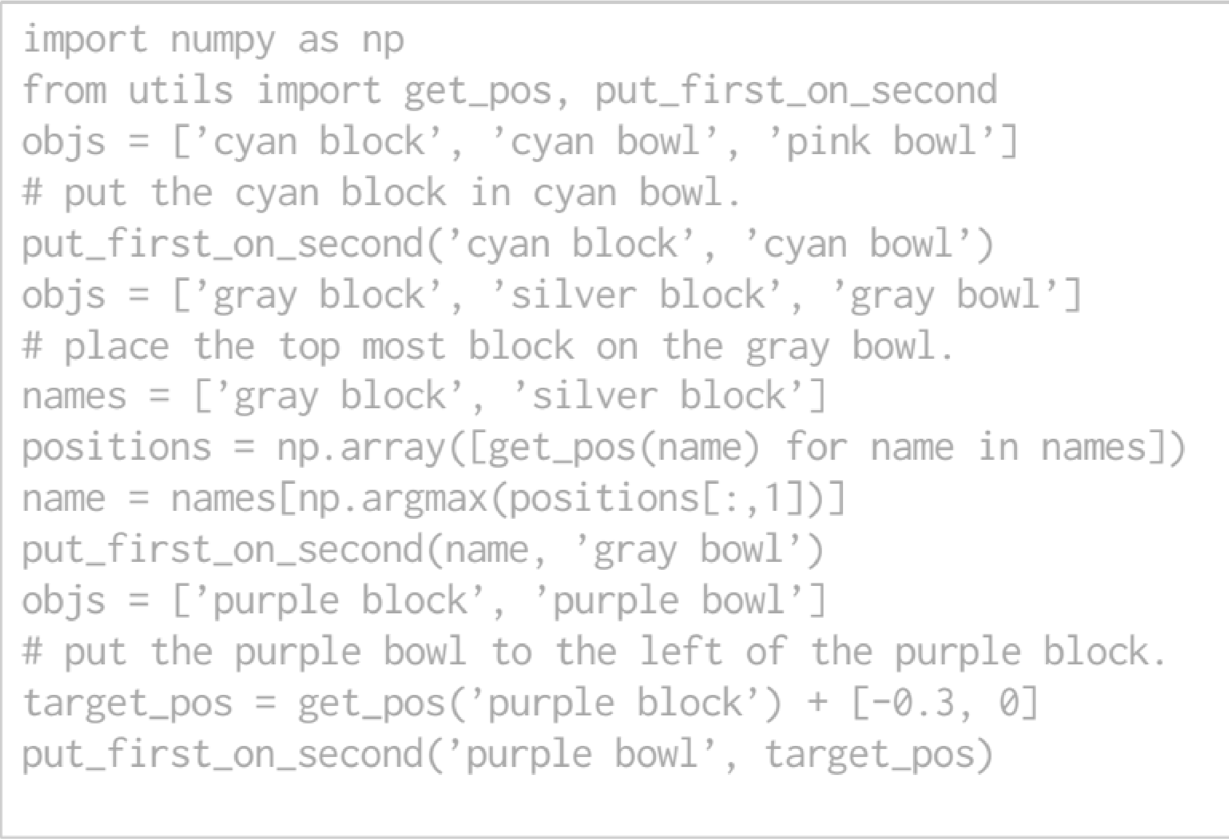


```
objs = [’blue bowl’, ’red block’, ’red bowl’, ’blue block’]
# the bowls.
ret_val = [’blue bowl’, ’red bowl’]
# sea-colored block.
ret_val = ’blue block’
# the other block.
ret_val = ’red block’
```

Language reasoning

Code as Policies, Google

- Prompt:
  - Hints: import statements that inform the LLM which APIs are available and type hints on how to use those APIs.
  - Example:
    - Input instructions are green / Generated Outputs are highlighted

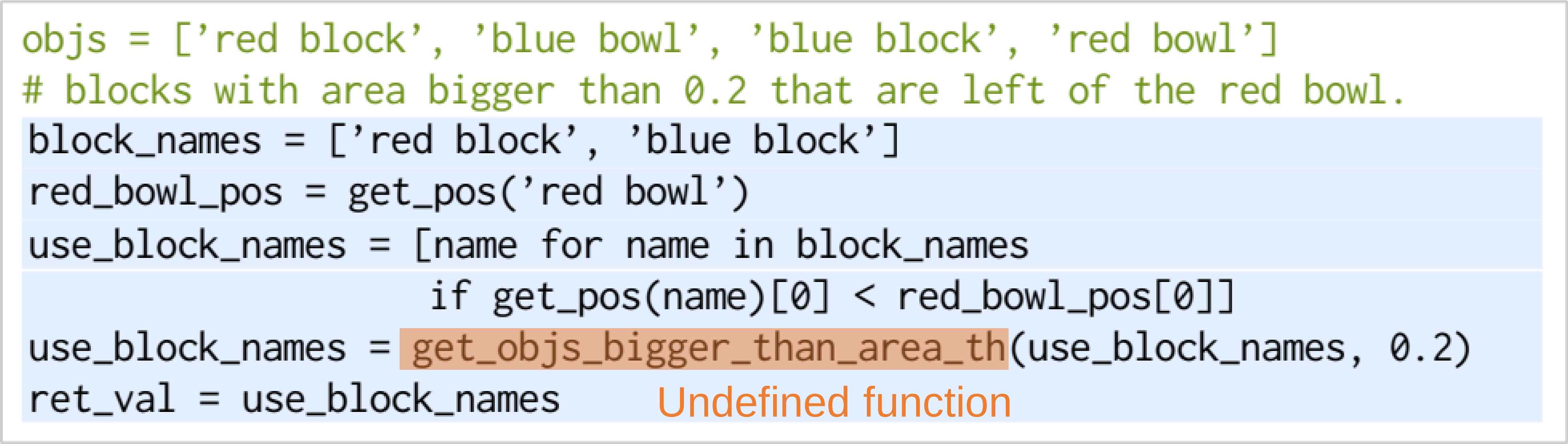


```
import numpy as np
from utils import get_pos, put_first_on_second
objs = ['cyan block', 'cyan bowl', 'pink bowl']
# put the cyan block in cyan bowl.
put_first_on_second('cyan block', 'cyan bowl')
objs = ['gray block', 'silver block', 'gray bowl']
# place the top most block on the gray bowl.
names = ['gray block', 'silver block']
positions = np.array([get_pos(name) for name in names])
name = names[np.argmax(positions[:,1])]
put_first_on_second(name, 'gray bowl')
objs = ['purple block', 'purple bowl']
# put the purple bowl to the left of the purple block.
target_pos = get_pos('purple block') + [-0.3, 0]
put_first_on_second('purple bowl', target_pos)
```

Full prompt

Code as Policies, Google

- Prompt:
  - Hints: import statements that inform the LLM which APIs are available and type hints on how to use those APIs.
  - Example:
    - Input instructions are green / Generated Outputs are highlighted

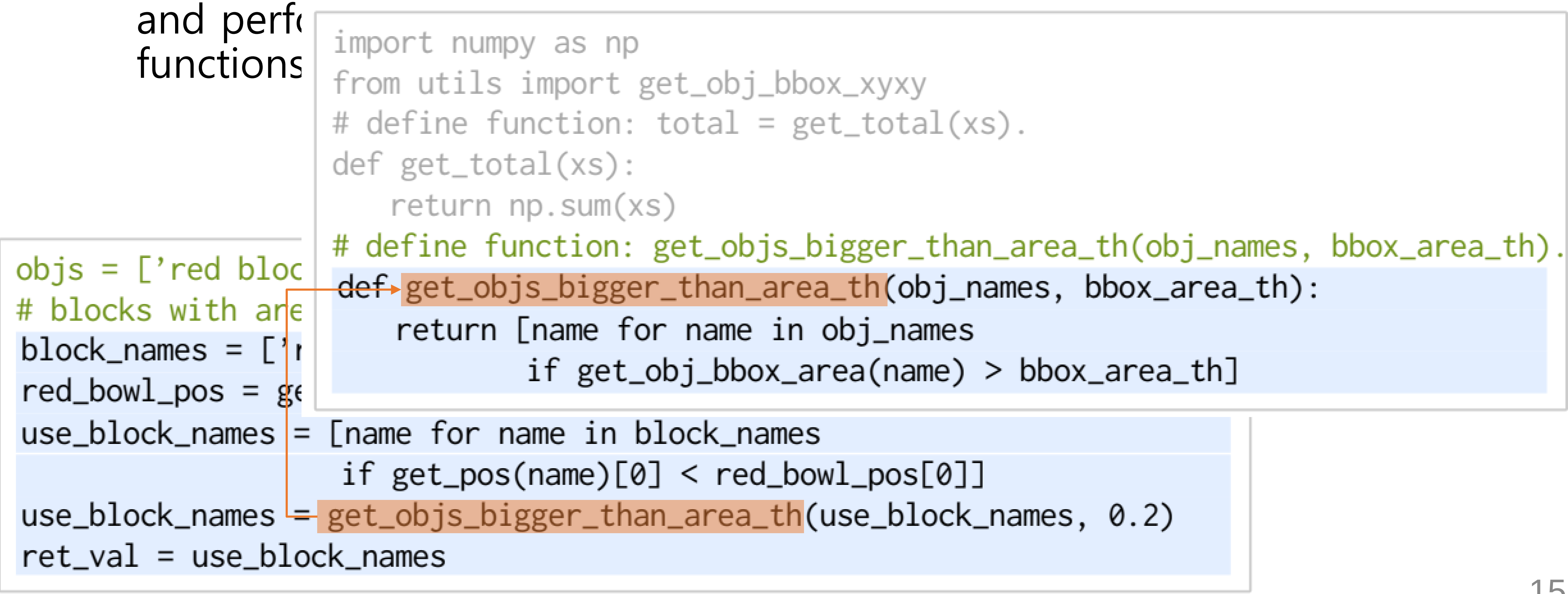


```
objs = [’red block’, ’blue bowl’, ’blue block’, ’red bowl’]
# blocks with area bigger than 0.2 that are left of the red bowl.
block_names = [’red block’, ’blue block’]
red_bowl_pos = get_pos(’red bowl’)
use_block_names = [name for name in block_names
                   if get_pos(name)[0] < red_bowl_pos[0]]
use_block_names = get_objs_bigger_than_area_th(use_block_names, 0.2)
ret_val = use_block_names
```

Undefined function

Code as Policies, Google

- Prompt:
  - Identify any undefined functions in the output generated by the LMP, and perf
    functions

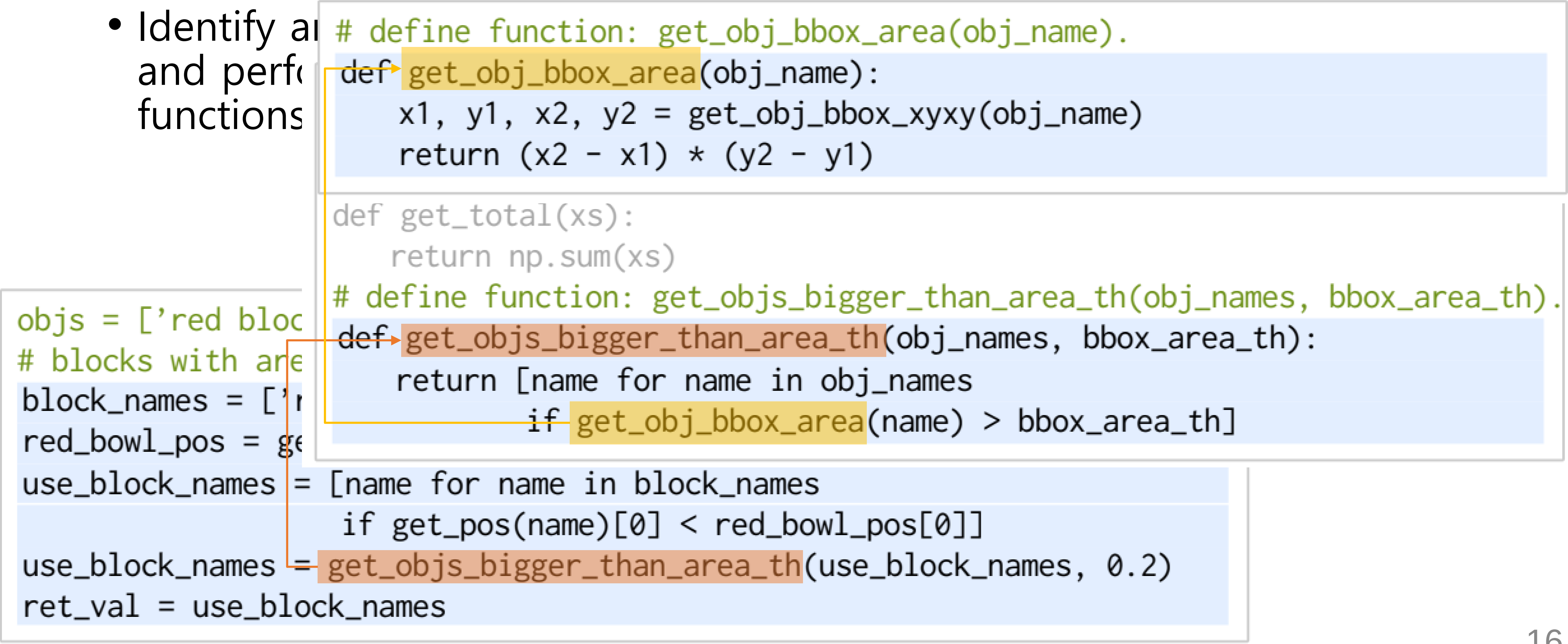


```
import numpy as np
from utils import get_obj_bbox_xyxy
# define function: total = get_total(xs).
def get_total(xs):
    return np.sum(xs)
# define function: get_objs_bigger_than_area_th(obj_names, bbox_area_th).
def get_objs_bigger_than_area_th(obj_names, bbox_area_th):
    return [name for name in obj_names
            if get_obj_bbox_area(name) > bbox_area_th]
```

```
objs = ['red bloc
# blocks with are
block_names = ['r
red_bowl_pos = ge
use_block_names = [name for name in block_names
                   if get_pos(name)[0] < red_bowl_pos[0]]
use_block_names = get_objs_bigger_than_area_th(use_block_names, 0.2)
ret_val = use_block_names
```

Code as Policies, Google

- Prompt:
  - Identify a
    and perf
    functions



```
# define function: get_obj_bbox_area(obj_name).
def get_obj_bbox_area(obj_name):
    x1, y1, x2, y2 = get_obj_bbox_xyxy(obj_name)
    return (x2 - x1) * (y2 - y1)
def get_total(xs):
    return np.sum(xs)
# define function: get_objs_bigger_than_area_th(obj_names, bbox_area_th).
def get_objs_bigger_than_area_th(obj_names, bbox_area_th):
    return [name for name in obj_names
                if get_obj_bbox_area(name) > bbox_area_th]
```

```
objs = ['red bloc
# blocks with are
block_names = ['r
red_bowl_pos = ge
use_block_names = [name for name in block_names
                    if get_pos(name)[0] < red_bowl_pos[0]]
use_block_names = get_objs_bigger_than_area_th(use_block_names, 0.2)
ret_val = use_block_names
```

Turtlebot w/LLM (OpenAI)

- Installation:
  - Create an OpenAI Account (https://auth.openai.com/create-account) and API key (https://platform.openai.com/account/api-keys)
  - Verify that your OpenAI account has some credit (https://platform.openai.com/usage)
  - Install the OpenAI Python library via *pip install openai* (It is recommended to utilize virtual environments…)
  - Clone the code into your workspace (https://github.com/turtlebot/turtlebot4_tutorials)
  - Build it!

Turtlebot w/LLM (OpenAI)

- To run the example, first start the Gazebo simulation, specifying the ‘depot’ world:

```
$ ros2 launch turtlebot4_gz_bringup turtlebot4_gz.launch.py nav2:=true slam:=false localization:=true
rviz:=true world:=depot map:=/opt/ros/jazzy/share/turtlebot4_navigation/maps/depot.yaml
```

- Open another terminal and run:

```
$ ros2 launch turtlebot4_openai_tutorials natural_language_nav_launch.py openai_api_key:=API_KEY
parking_brake:=false
```

- Once the robot is undocked, open a third terminal and run:

```
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Dock"
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Go to -1,0, face East"
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Go to 5,5"
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Move to the wooden object"
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Navigate to the item which can hold oil"
$ ros2 topic pub --once /user_input std_msgs/msg/String "data: Travel to the room containing a toilet"
```