Question: Are there either any chairs or televisions that are black?

Baseline: **no** ✗

PropTest: **yes** ✓

Generated Test

```
def execute_test(image):
    result = execute_command(image, my_fig,
                             time_wait_between_lines,
                             syntax)
    # Test case 1:
    assert result in [ “yes”, “no”], "Expected output to be yes or no"
    return result
```

Generated Code

```
def execute_command(image, my_fig, time_wait_between_lines, syntax):
    image_patch = ImagePatch(image)
    chairs_patches = image_patch.find("chairs”)
    televisions_patches = image_patch.find("televisions”)
    black_chairs = []
    black_televisions = []
    for chair_patch in chairs_patches:
        if chair_patch.verify_property("chair”, "black”):
            black_chairs.append(chair_patch)
    for television_patch in televisions_patches:
        if television_patch.verify_property("television”,"black”):
            black_televisions.append(television_patch)
    return bool_to_yesno(len(black_chairs) > 0 or len(black_televisions) > 0)
```

Question: What is the largest city in the nation where this plane is headquartered?

Baseline: **new york** ✗

PropTest: **tokyo** ✓

Generated Test

```
def execute_test(image):
    result = execute_command(image, my_fig,
                             time_wait_between_lines,
                             syntax)
    # Test case 1:
    assert isinstance(result, str), "Expected output to be string."
    # Test case 2:
    assert len(result.split()) in [1, 2],
           "Expected output to be one or two words"
    # Test case 3:
    assert llm_query(f"Is {result} a city?”, long_answer = False) == 'yes',
                     "Expected output to be a city"
    return result
```

Generated Code

```
def execute_command(image, my_fig, time_wait_between_lines, syntax):
    image_patch = ImagePatch(image)
    guesses = []
    plane_location= image_patch.simple_query(" Where is this plane headquartered?")
    external_knowledge_query = "What is the largest city in the nation where {} is
                headquartered?".format(plane_location)
    step_by_step_guess = llm_query(external_knowledge_query)
    guesses.append("where is this plane headquartered is{}".format(plane_location) +
                ", so " + step_by_step_guess)

    direct_guess = image.simple_query("What is the largest city in the nation where
                this plane is headquartered?")
    guess.append(direct_guess)
    return process_guesses ("What is the largest city in the nation where this plane is
                headquartered?", guesses)
```

Referring expression: guy on the yellow bike

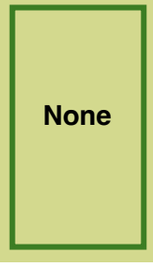
Generated Test

```
def execute_test(image):
    result = execute_command(image, my_fig, time_wait_between_lines,
                             syntax)
    # Test case 1:
    assert ”yes” in simple_query(“Is there a guy?”).lower(),
           "Expected output to be a guy"
    # Test case 2:
    assert bool_to_yesno(result.verify_property (“guy”, “yellow bike”)),
                         "Expected output to be a guy on a yellow bike"
    return result
```

Generated Code

```
def execute_command(image, my_fig, time_wait_between_lines, syntax):
    image_patch = ImagePatch(image)
    person_patches = image_patch.find("person")
    person_patch = best_image_match(person_patches, [“yellow bike”])
    return person_patch
```

Baseline:



✗

PropTest:

✓