Learn Python the Fun Way

Liana Bakradze

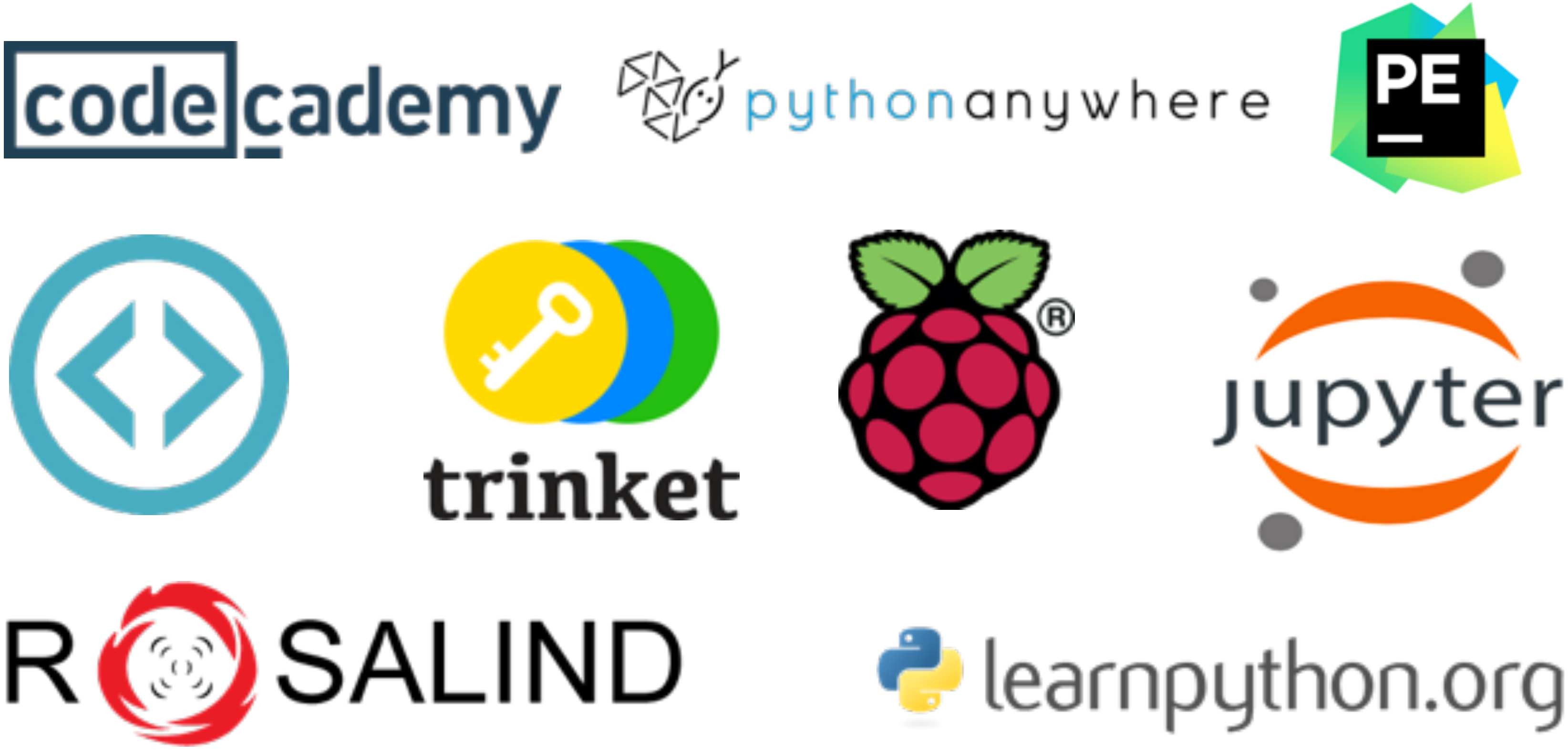
liana.bakradze@jetbrains.com
@med_vector

About Me

Saint Petersburg

Photo made by Andrey Korzun is licensed by CC BY-SA 3.0

Learn Python Resources



Why Games?

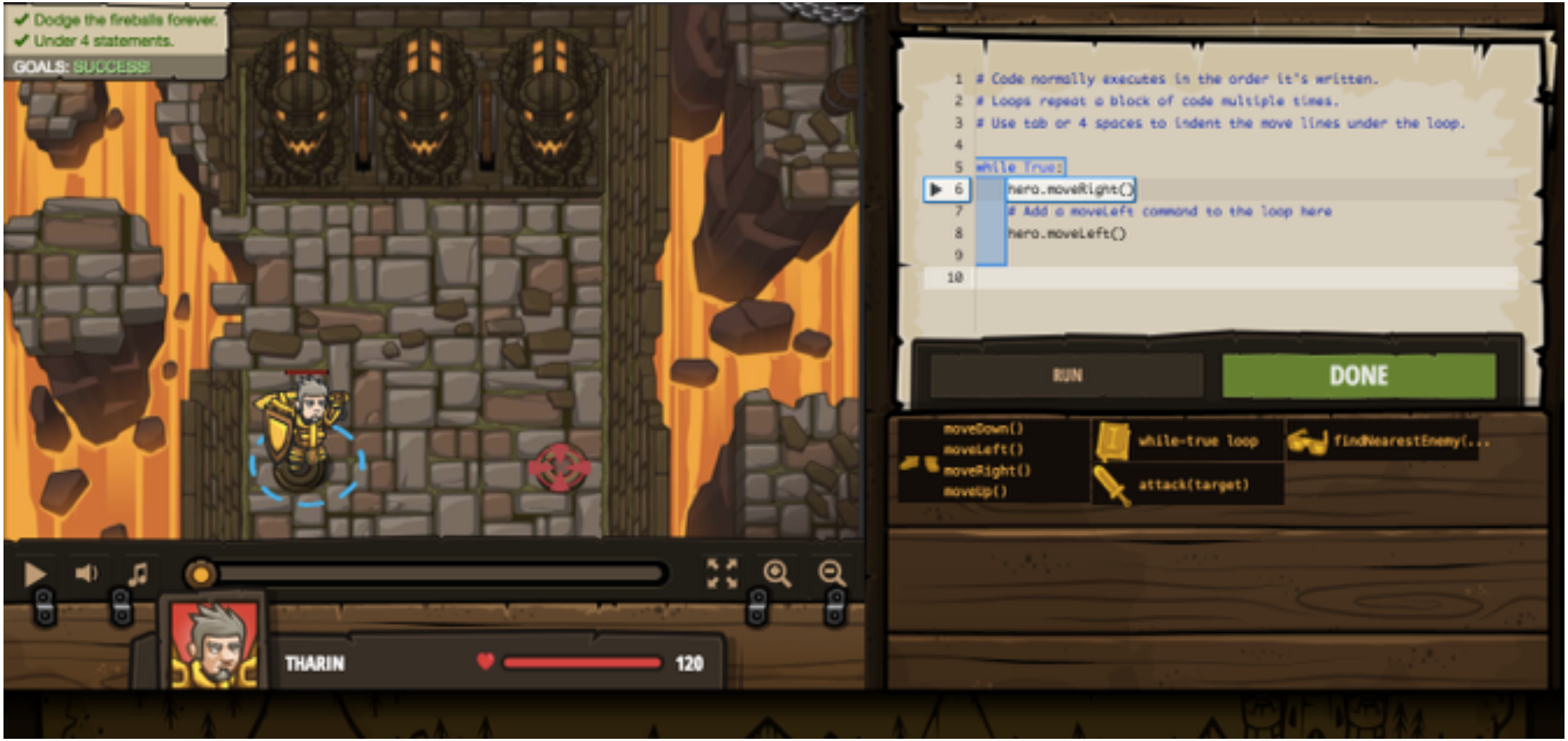
No fear of failure

Reward

Competition

Visualisation

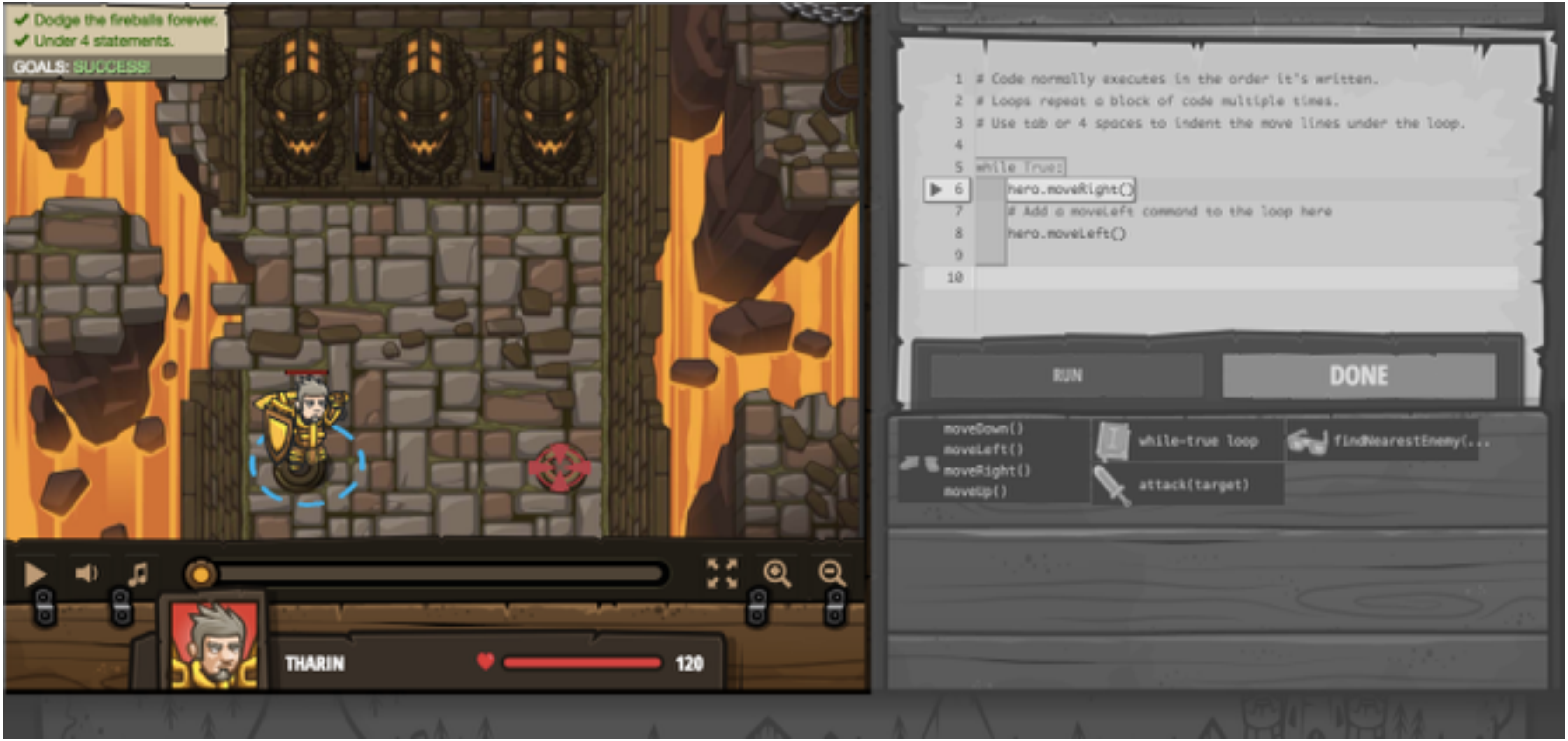
CODE COMBAT



✓ Dodge the fireballs forever.
✓ Under 4 statements.
GOALS: SUCCESS!

```
# Code normally executes in the order it's written.
# Loops repeat a block of code multiple times.
# Use tab or 4 spaces to indent the move lines under the loop.

while True:
    hero.moveRight()
    # Add a moveLeft command to the loop here
    hero.moveLeft()

```

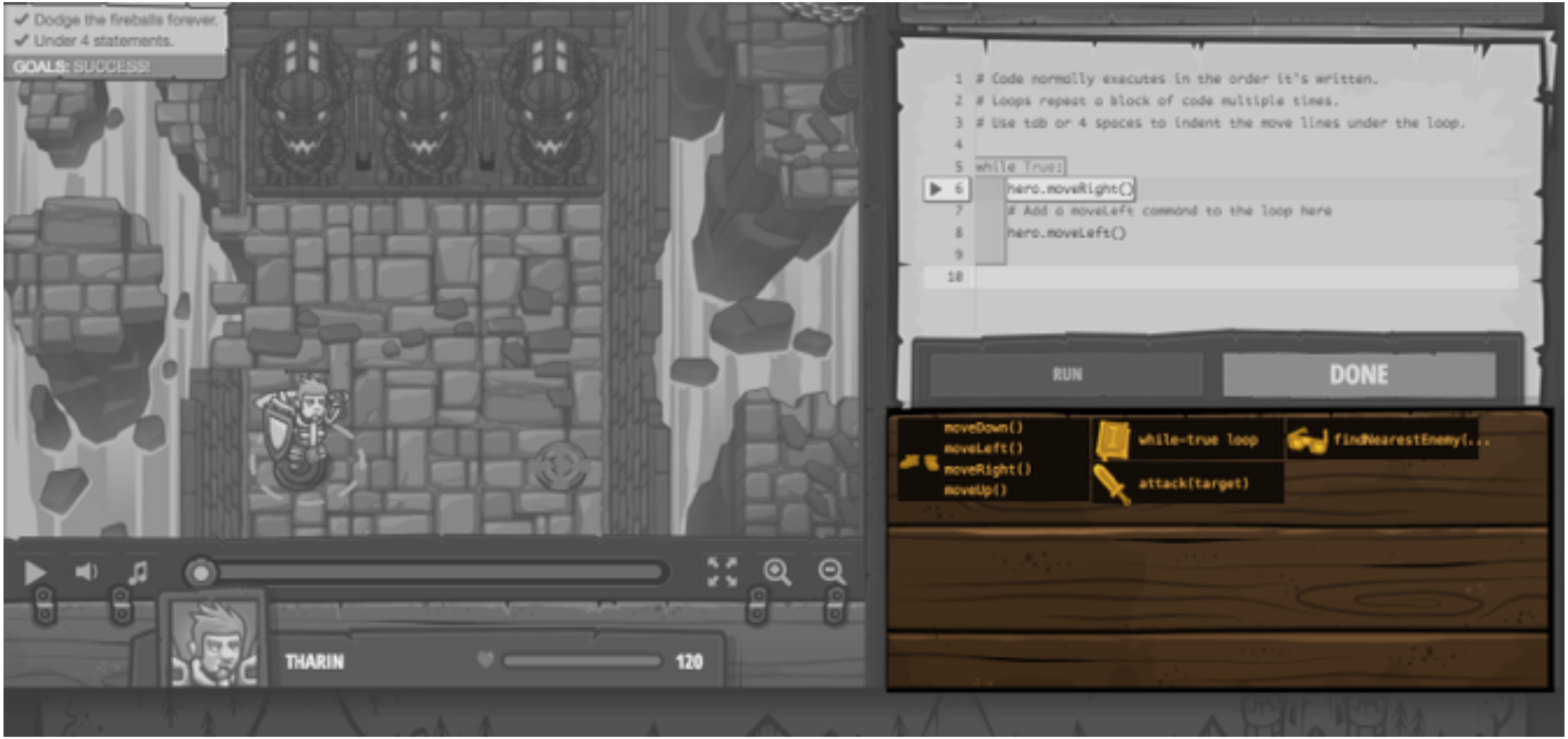
CODE COMBAT



✓ Dodge the fireballs forever.
✓ Under 4 statements.
GOALS: SUCCESS!

```
# Code normally executes in the order it's written.
# Loops repeat a block of code multiple times.
# Use tab or 4 spaces to indent the move lines under the loop.

while True:
    hero.moveRight()
    # Add a moveLeft command to the loop here
    hero.moveLeft()
```

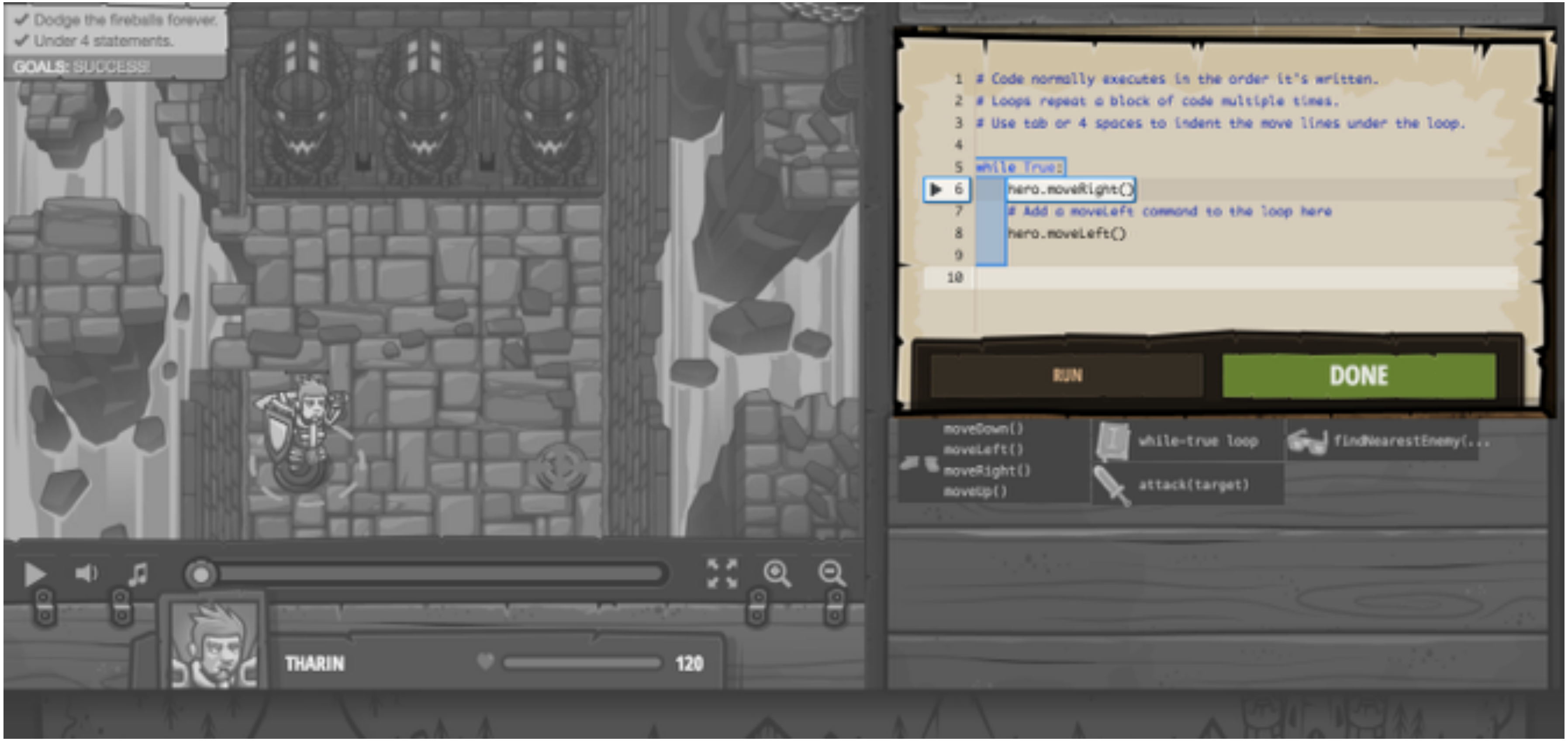
CODE COMBAT



✓ Dodge the fireballs forever.
✓ Under 4 statements.
GOALS: SUCCESS!

```
# Code normally executes in the order it's written.
# Loops repeat a block of code multiple times.
# Use tab or 4 spaces to indent the move lines under the loop.

while True:
    hero.moveRight()
    # Add a moveLeft command to the loop here
    hero.moveLeft()

```

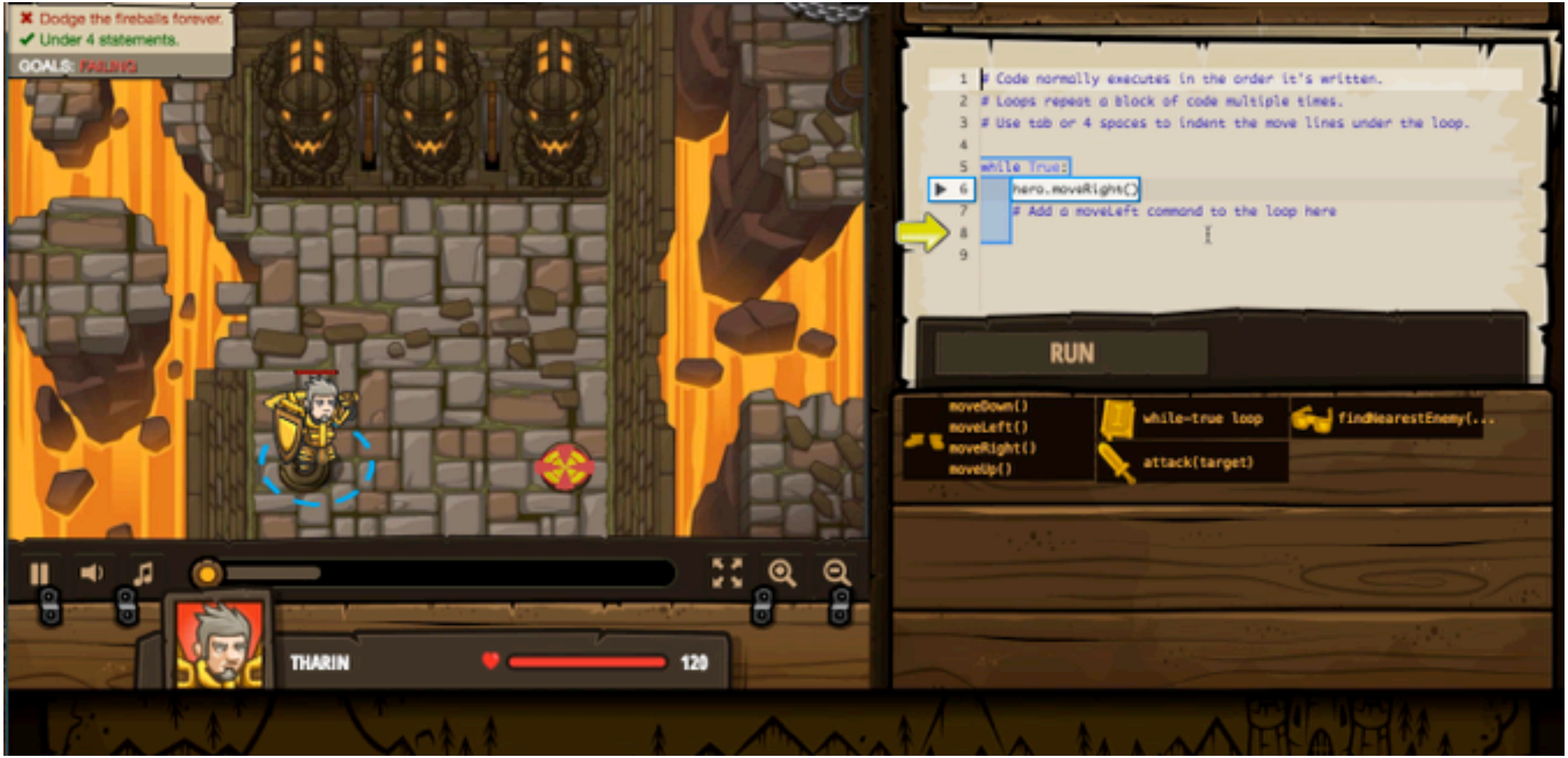
CODE COMBAT



✓ Dodge the fireballs forever.
✓ Under 4 statements.
GOALS: SUCCESS!

```
# Code normally executes in the order it's written.
# Loops repeat a block of code multiple times.
# Use tab or 4 spaces to indent the move lines under the loop.

while True:
    hero.moveRight()
    # Add a moveLeft command to the loop here
    hero.moveLeft()

```

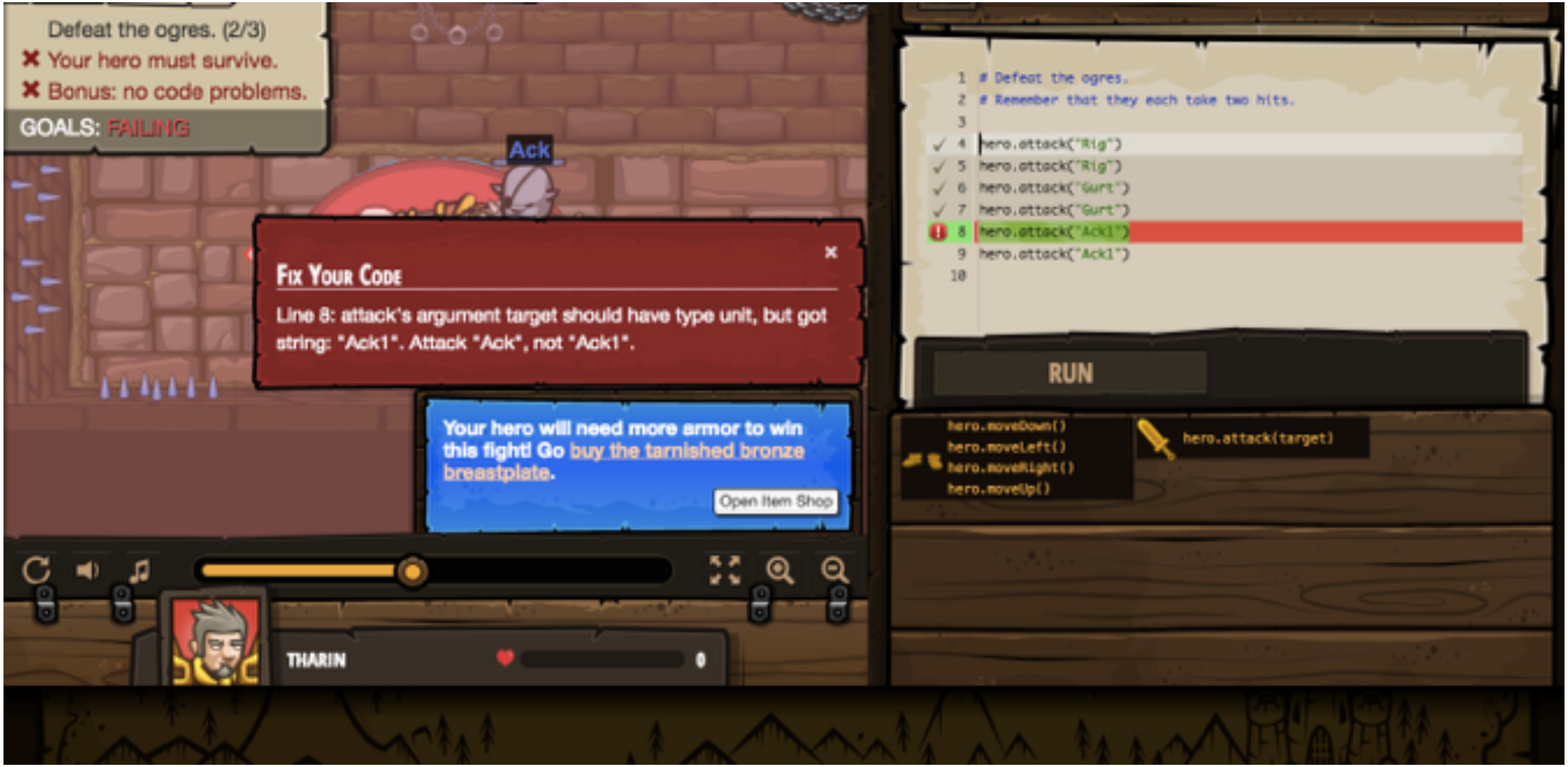
CODE COMBAT



✗ Dodge the fireballs forever.
✓ Under 4 statements.
GOALS: FAILING

```
# Code normally executes in the order it's written.
# Loops repeat a block of code multiple times.
# Use tab or 4 spaces to indent the move lines under the loop.

while True:
    hero.moveRight()
    # Add a moveLeft command to the loop here

```

CODE COMBAT



Defeat the ogres. (2/3)

✗ Your hero must survive.

✗ Bonus: no code problems.

GOALS: FAILING

```
1  # Defeat the ogres.
2  # Remember that they each take two hits.
3
4  hero.attack("Rig")
5  hero.attack("Rig")
6  hero.attack("Gurt")
7  hero.attack("Gurt")
8  hero.attack("Ack1")
9  hero.attack("Ack1")
10
```

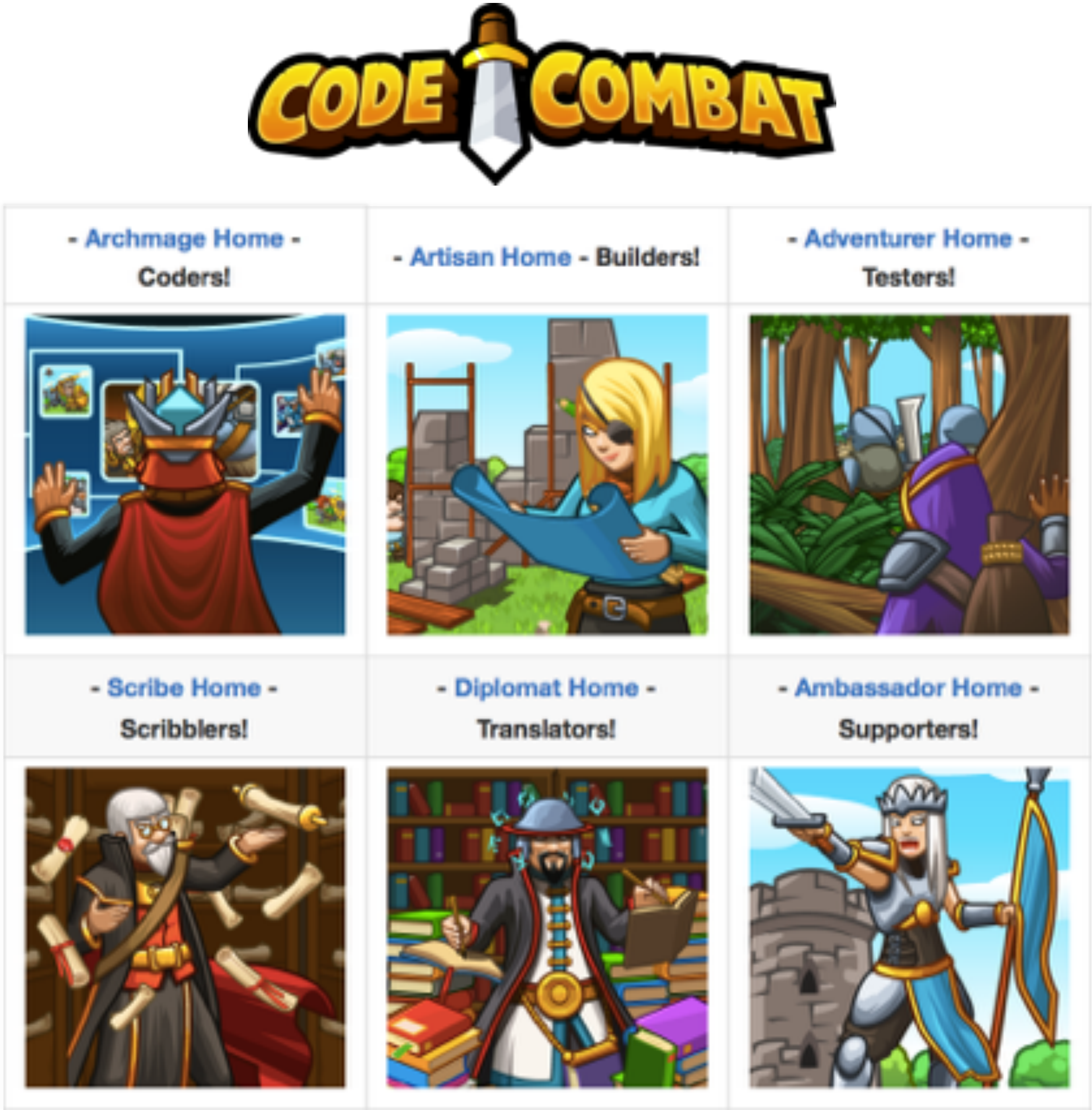
Fix Your Code

Line 8: attack's argument target should have type unit, but got string: "Ack1". Attack "Ack", not "Ack1".

Your hero will need more armor to win this fight! Go buy the tarnished bronze breastplate.

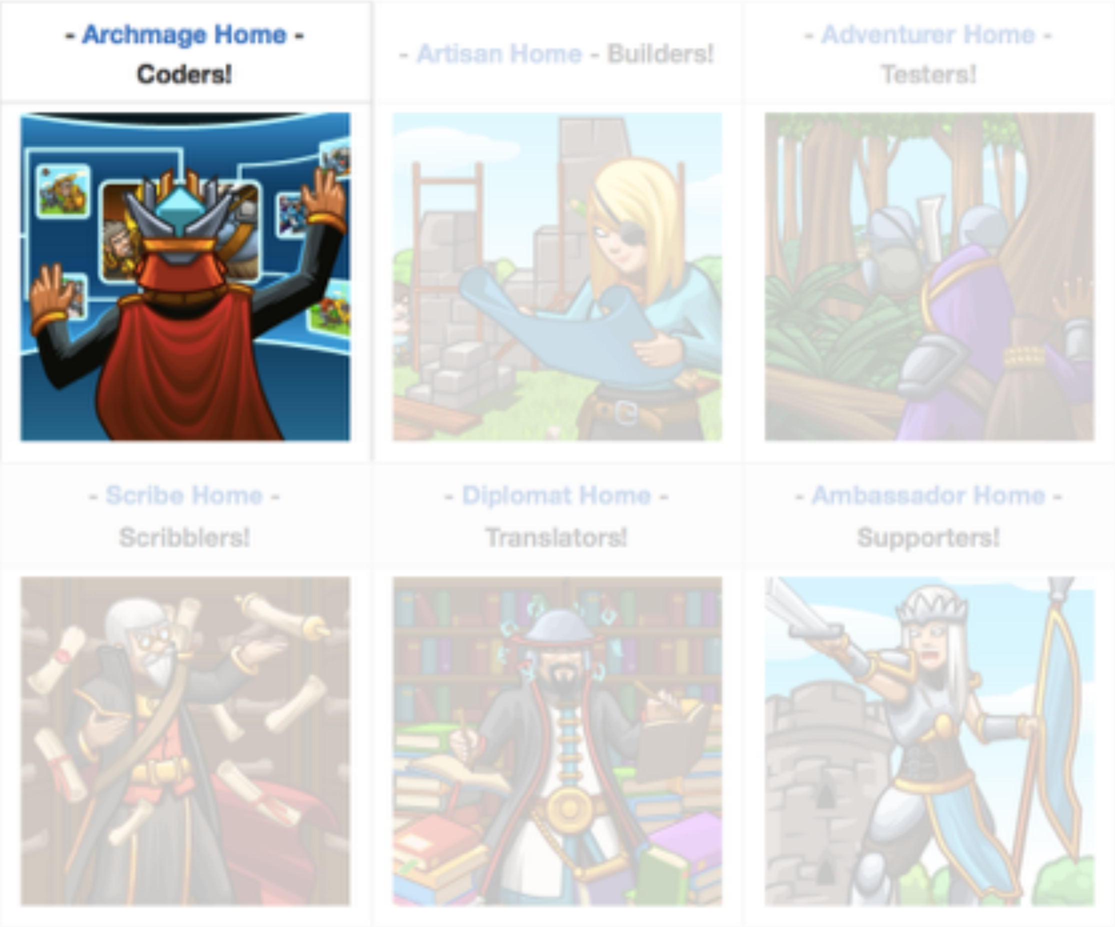
```
hero.moveDown()
hero.moveLeft()
hero.moveRight()
hero.moveUp()
```

```
hero.attack(target)
```



- Archmage Home - Coders!	- Artisan Home - Builders!	- Adventurer Home - Testers!
- Scribe Home - Scribblers!	- Diplomat Home - Translators!	- Ambassador Home - Supporters!

CODE COMBAT



- Archmage Home - Coders!

- Artisan Home - Builders!

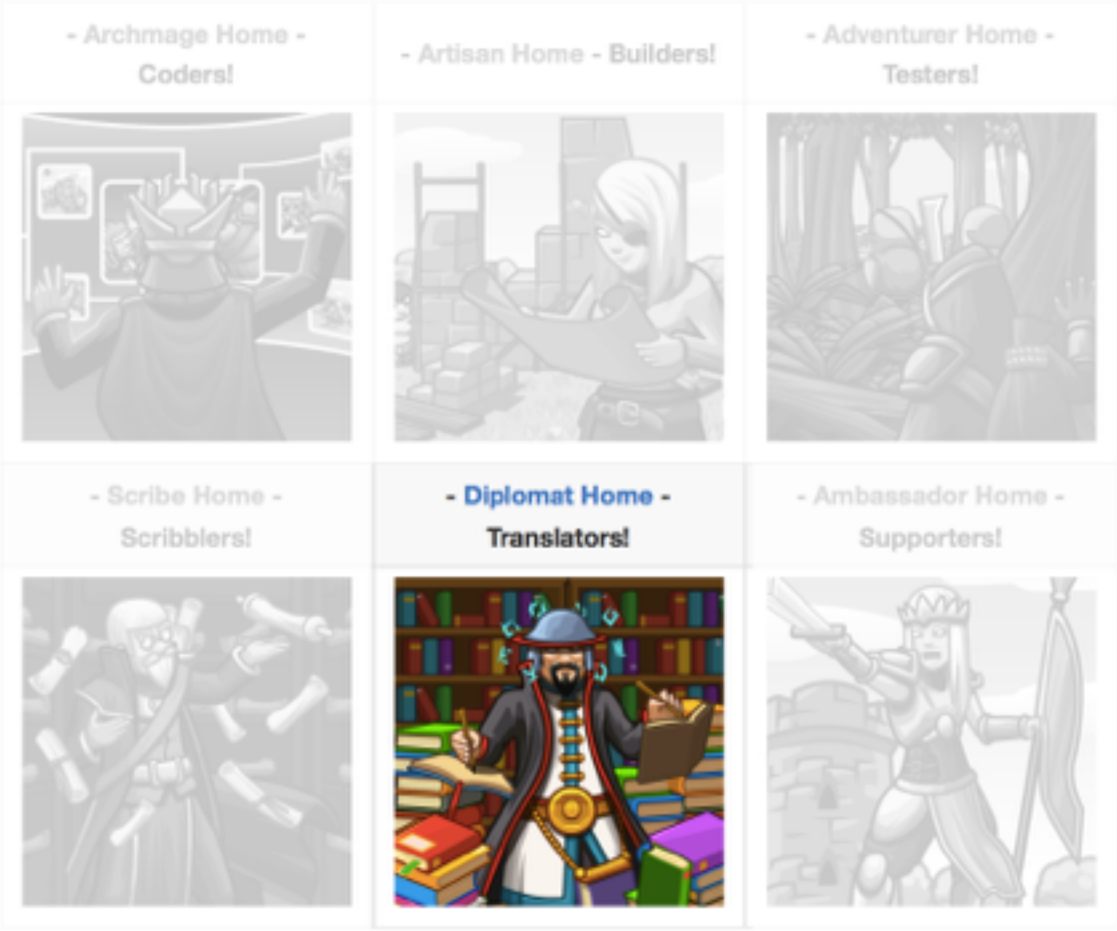
- Adventurer Home - Testers!

- Scribe Home - Scribblers!

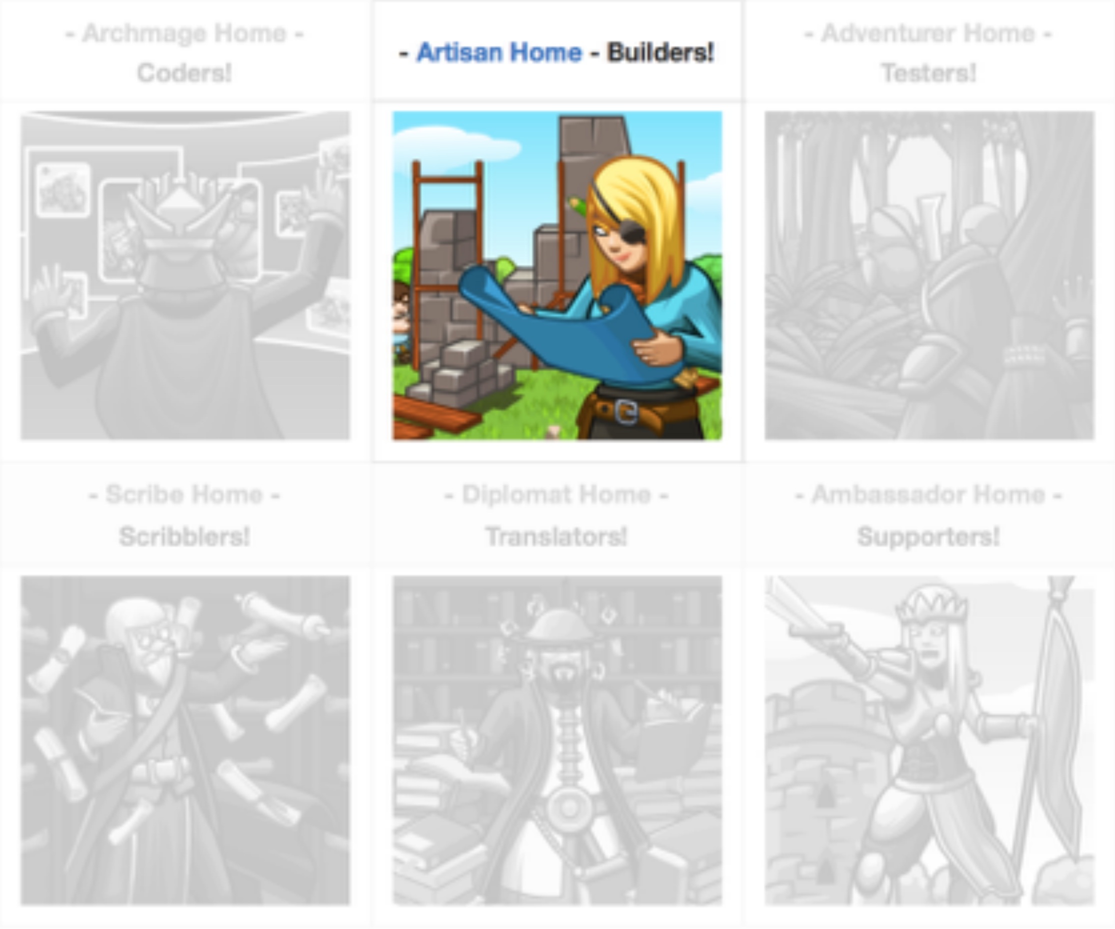
- Diplomat Home - Translators!

- Ambassador Home - Supporters!

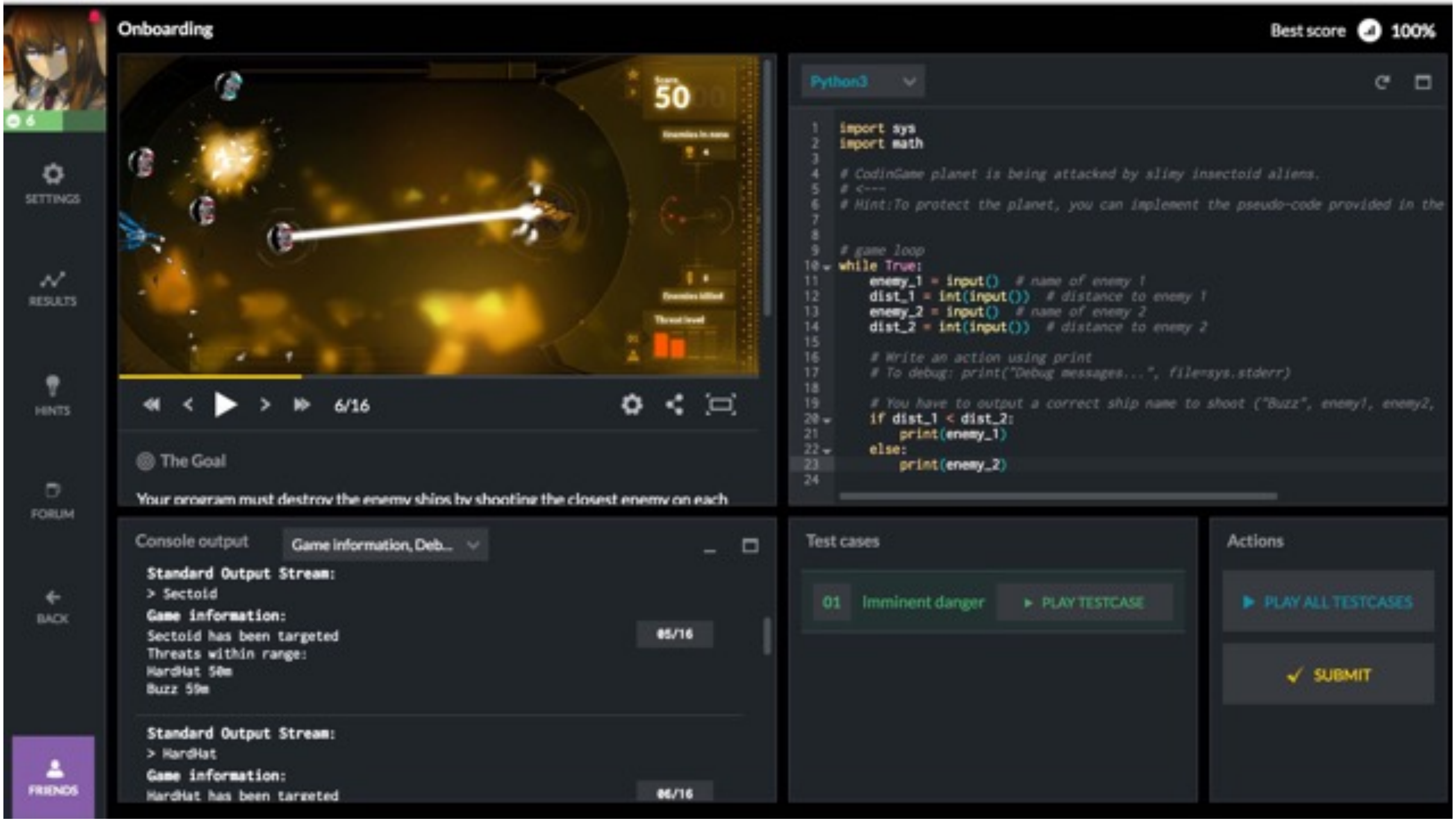
CODE COMBAT


- Archmage Home - Coders!
- Artisan Home - Builders!
- Adventurer Home - Testers!
- Scribe Home - Scribblers!
- Diplomat Home - Translators!
- Ambassador Home - Supporters!

CODE COMBAT


- Archmage Home -
Coders!
- Artisan Home - Builders!
- Adventurer Home -
Testers!
- Scribe Home -
Scribblers!
- Diplomat Home -
Translators!
- Ambassador Home -
Supporters!

CodinGame



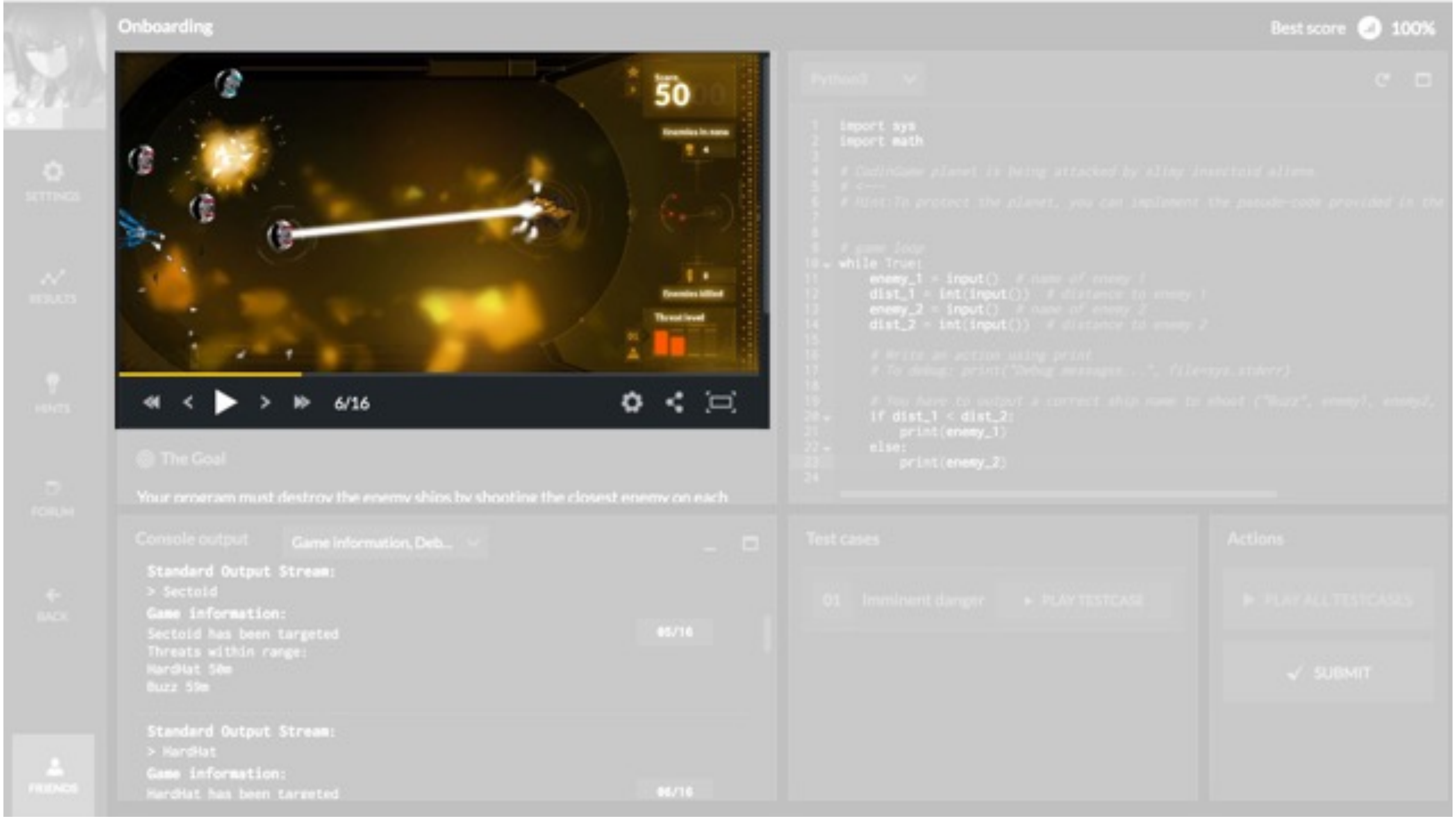
The Goal

Your program must destroy the enemy ships by shooting the closest enemy on each

```python
import sys
import math

# CodinGame planet is being attacked by slimy insectoid aliens.
# <---
# Hint:To protect the planet, you can implement the pseudo-code provided in the


# game loop
while True:
    enemy_1 = input()  # name of enemy 1
    dist_1 = int(input())  # distance to enemy 1
    enemy_2 = input()  # name of enemy 2
    dist_2 = int(input())  # distance to enemy 2

    # Write an action using print
    # To debug: print("Debug messages...", file=sys.stderr)

    # You have to output a correct ship name to shoot ("Buzz", enemy1, enemy2,
    if dist_1 < dist_2:
        print(enemy_1)
    else:
        print(enemy_2)

```

```
Standard Output Stream:
> Sectoid
Game information:
Sectoid has been targeted
Threats within range:
HardHat 50m
Buzz 59m

Standard Output Stream:
> HardHat
Game information:
HardHat has been targeted
```

CodinGame



The Goal

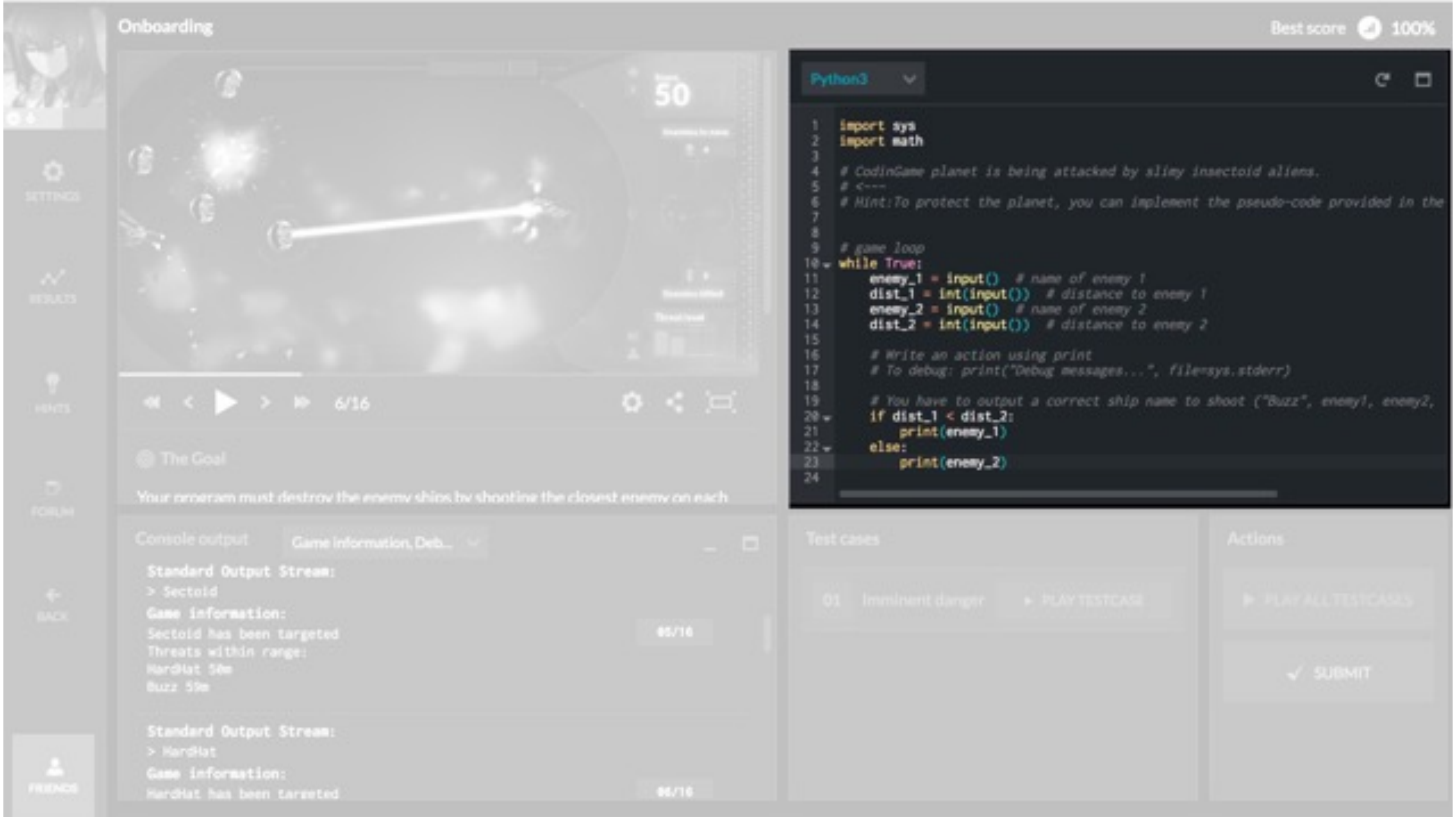
Your program must destroy the enemy ships by shooting the closest enemy on each

```
import sys
import math

# CodinGame planet is being attacked by slimy insectoid aliens.
# <---
# Hint:To protect the planet, you can implement the pseudo-code provided in the


# game loop
while True:
    enemy_1 = input()  # name of enemy 1
    dist_1 = int(input())  # distance to enemy 1
    enemy_2 = input()  # name of enemy 2
    dist_2 = int(input())  # distance to enemy 2

    # Write an action using print
    # To debug: print("Debug messages...", file=sys.stderr)

    # You have to output a correct ship name to shoot ("Buzz", enemy1, enemy2,
    if dist_1 < dist_2:
        print(enemy_1)
    else:
        print(enemy_2)

```

```
Standard Output Stream:
> Sectoid
Game information:
Sectoid has been targeted                                05/16
Threats within range:
HardHat 50m
Buzz 59m

Standard Output Stream:
> HardHat
Game information:
HardHat has been targeted                                06/16
```

01 Imminent danger ▶ PLAY TESTCASE

CodinGame



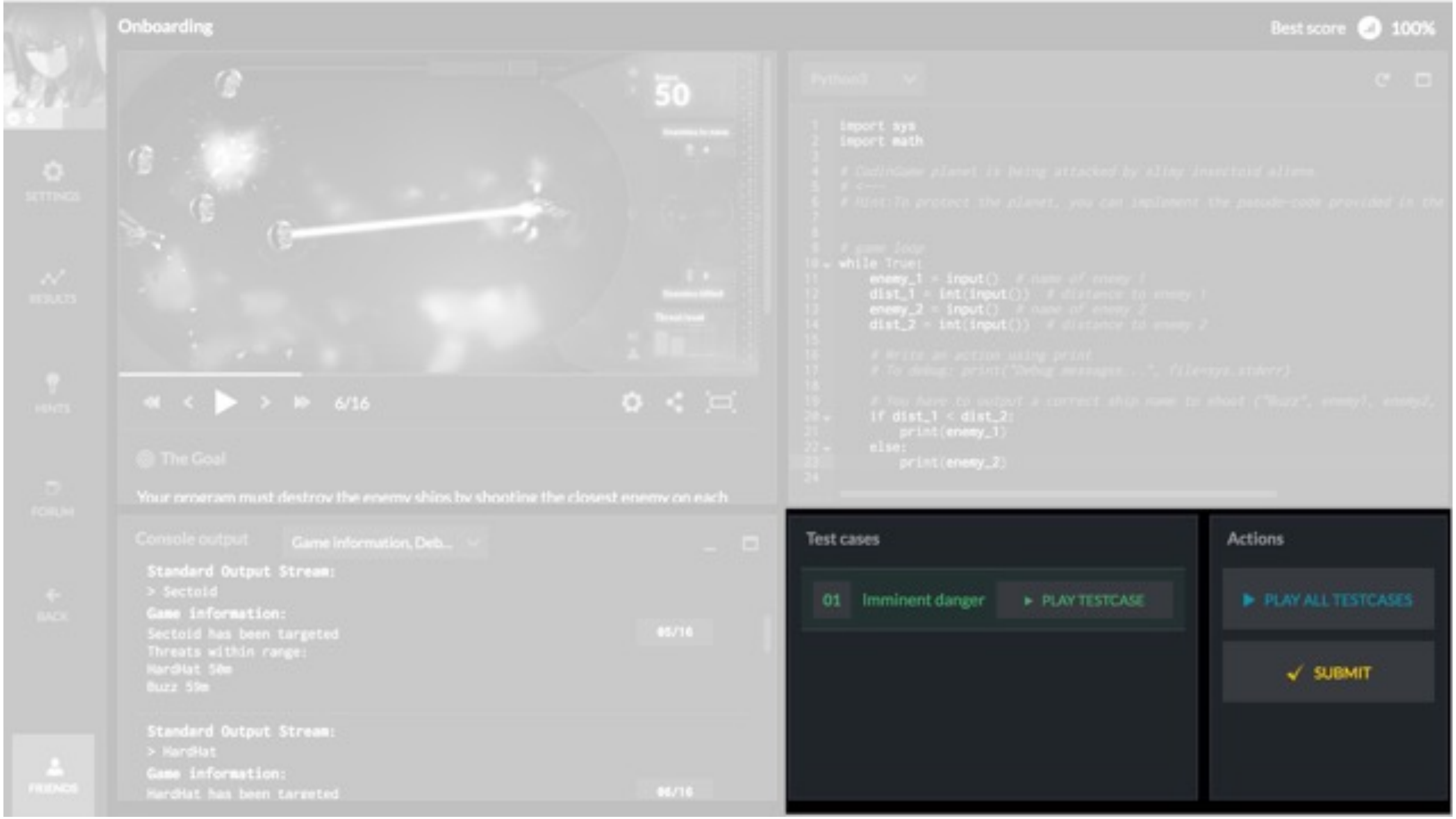
```python
import sys
import math

# CodinGame planet is being attacked by slimy insectoid aliens.
# <---
# Hint:To protect the planet, you can implement the pseudo-code provided in the


# game loop
while True:
    enemy_1 = input()  # name of enemy 1
    dist_1 = int(input())  # distance to enemy 1
    enemy_2 = input()  # name of enemy 2
    dist_2 = int(input())  # distance to enemy 2

    # Write an action using print
    # To debug: print("Debug messages...", file=sys.stderr)

    # You have to output a correct ship name to shoot ("Buzz", enemy1, enemy2,
    if dist_1 < dist_2:
        print(enemy_1)
    else:
        print(enemy_2)

```

The Goal

Your program must destroy the enemy ships by shooting the closest enemy on each

```
Standard Output Stream:
> Sectoid
Game information:
Sectoid has been targeted                    05/16
Threats within range:
HardHat 50m
Buzz 59m

Standard Output Stream:
> HardHat
Game information:
HardHat has been targeted                    06/16
```

CodinGame

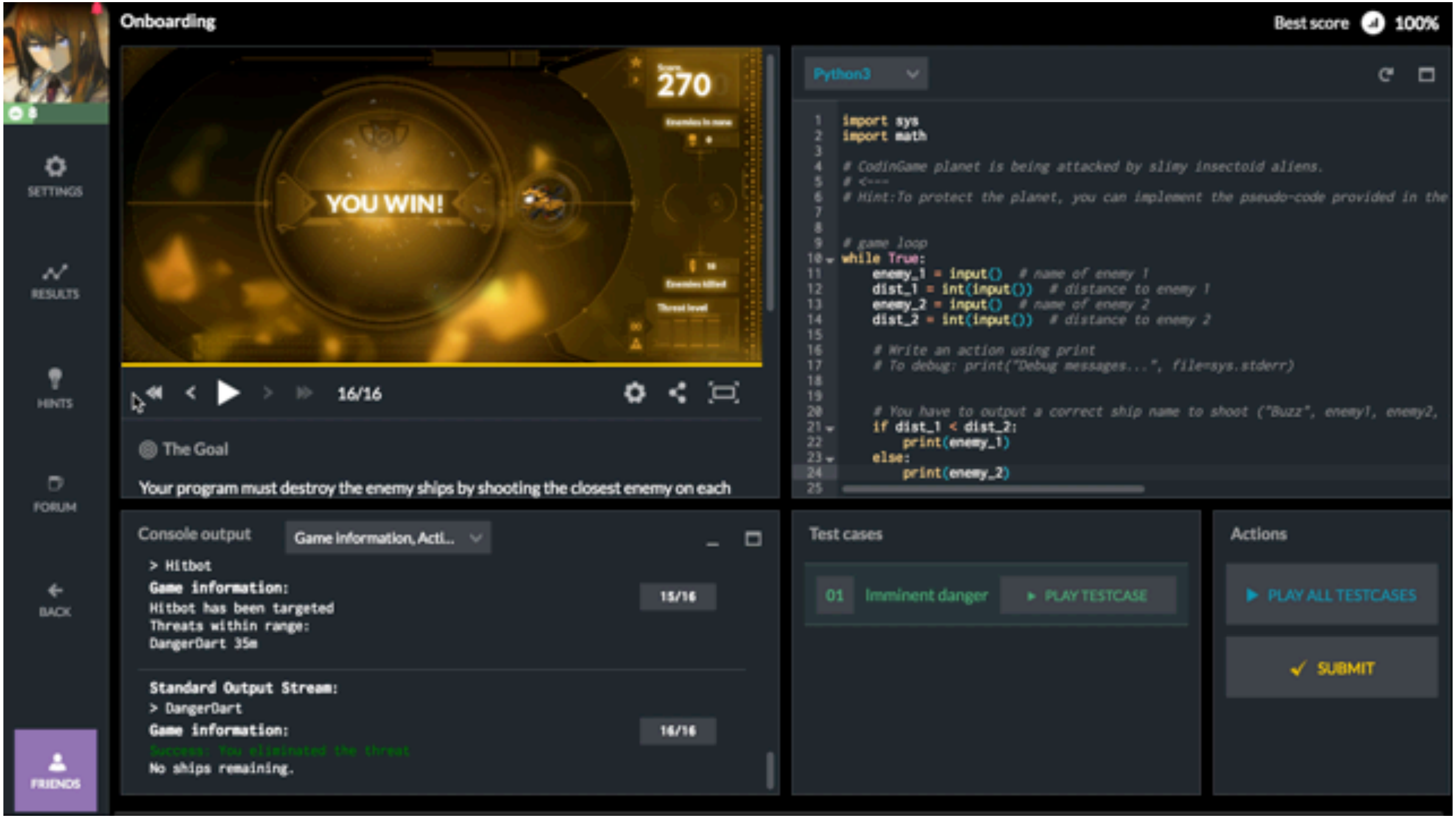


Your program must destroy the enemy ships by shooting the closest enemy on each

```python
import sys
import math

# CodinGame planet is being attacked by slimy insectoid aliens
# <---
# Hint:To protect the planet, you can implement the pseudo-code provided in the


# game loop
while True:
    enemy_1 = input()  # name of enemy 1
    dist_1 = int(input())  # distance to enemy 1
    enemy_2 = input()  # name of enemy 2
    dist_2 = int(input())  # distance to enemy 2

    # Write an action using print
    # To debug: print("Debug messages...", file=sys.stderr)

    # You have to output a correct ship name to shoot ("Buzz", enemy1, enemy2,
    if dist_1 < dist_2:
        print(enemy_1)
    else:
        print(enemy_2)
```

```
Standard Output Stream:
> Sectoid
Game information:
Sectoid has been targeted
Threats within range:
HardHat 50m
Buzz 59m

Standard Output Stream:
> HardHat
Game information:
HardHat has been targeted
```

CodinGame



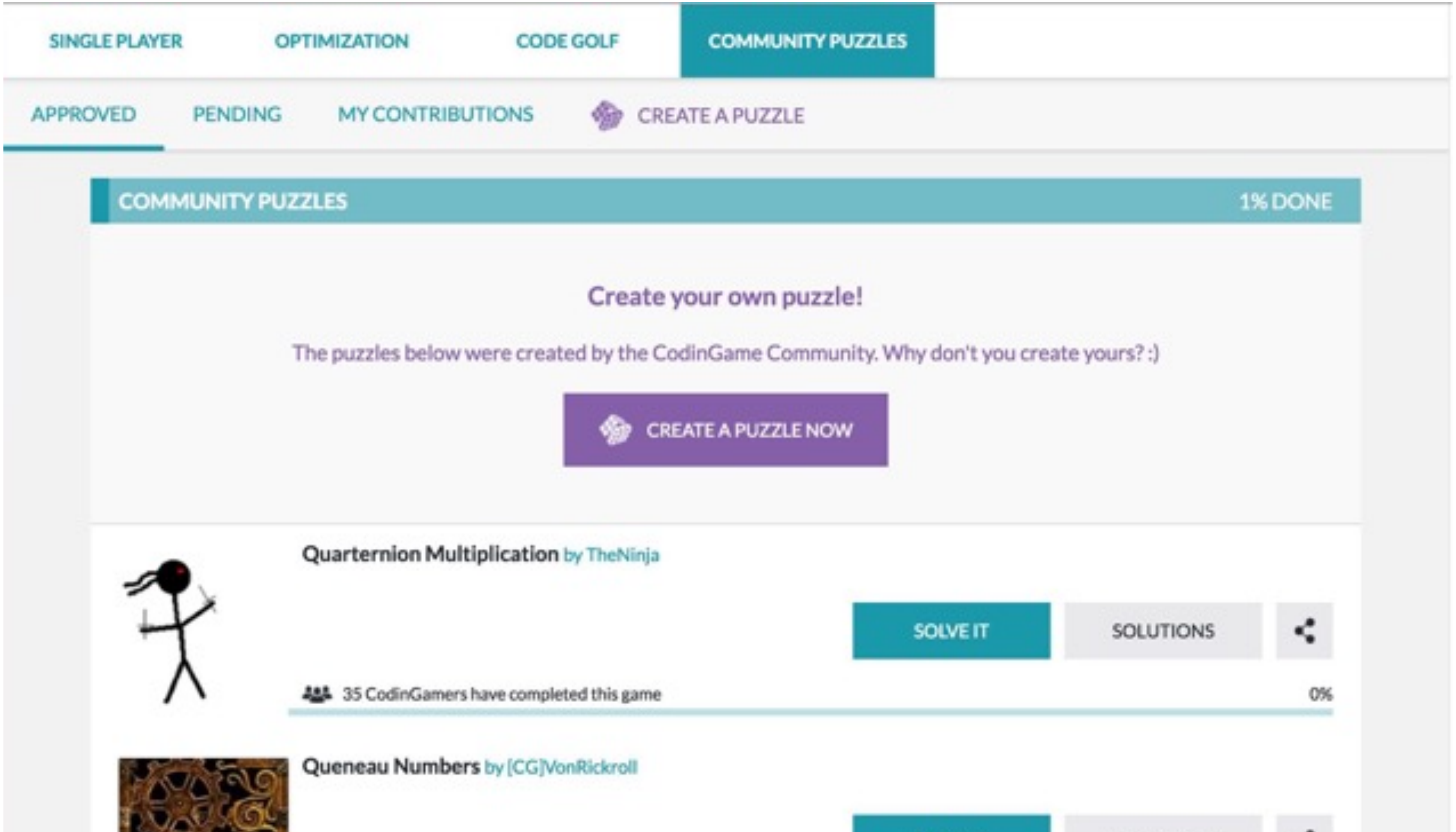
The Goal

Your program must destroy the enemy ships by shooting the closest enemy on each

```python
import sys
import math

# CodinGame planet is being attacked by slimy insectoid aliens.
# <---
# Hint:To protect the planet, you can implement the pseudo-code provided in the


# game loop
while True:
    enemy_1 = input()  # name of enemy 1
    dist_1 = int(input())  # distance to enemy 1
    enemy_2 = input()  # name of enemy 2
    dist_2 = int(input())  # distance to enemy 2

    # Write an action using print
    # To debug: print("Debug messages...", file=sys.stderr)


    # You have to output a correct ship name to shoot ("Buzz", enemy1, enemy2,
    if dist_1 < dist_2:
        print(enemy_1)
    else:
        print(enemy_2)

```

```
> Hitbot
Game information:
Hitbot has been targeted
Threats within range:
DangerDart 35m

Standard Output Stream:
> DangerDart
Game information:
Success: You eliminated the threat
No ships remaining.
```

CodinGame



Create your own puzzle!

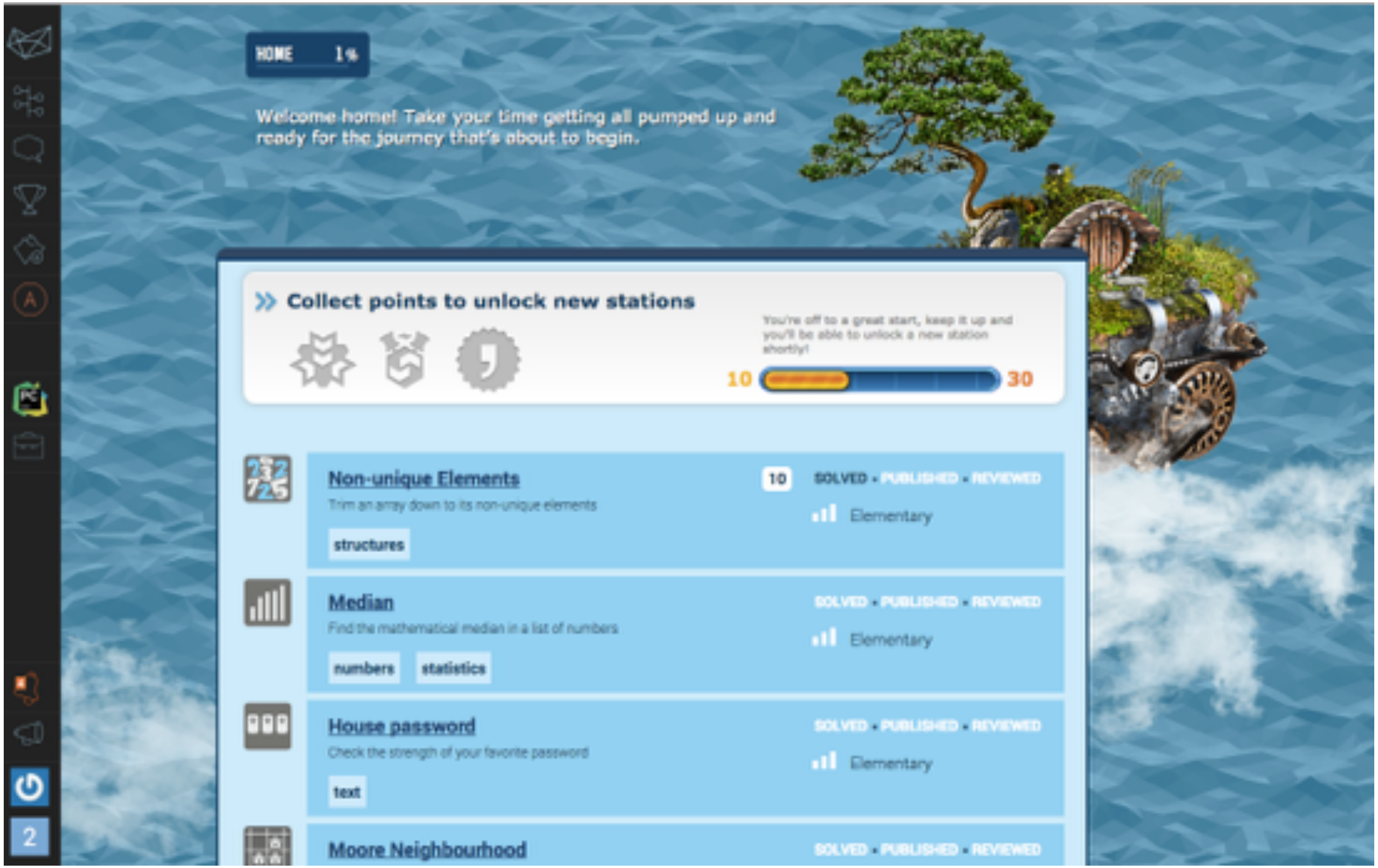
The puzzles below were created by the CodinGame Community. Why don't you create yours? :)

Quarternion Multiplication by TheNinja

35 CodinGamers have completed this game

Queneau Numbers by [CG]VonRickroll

CheckiO



HOME 1%

Welcome home! Take your time getting all pumped up and ready for the journey that's about to begin.

Collect points to unlock new stations

You're off to a great start, keep it up and you'll be able to unlock a new station shortly!

10 30

Non-unique Elements 10 SOLVED • PUBLISHED • REVIEWED

Trim an array down to its non-unique elements

Elementary

structures

Median SOLVED • PUBLISHED • REVIEWED

Find the mathematical median in a list of numbers

Elementary

numbers statistics

House password SOLVED • PUBLISHED • REVIEWED

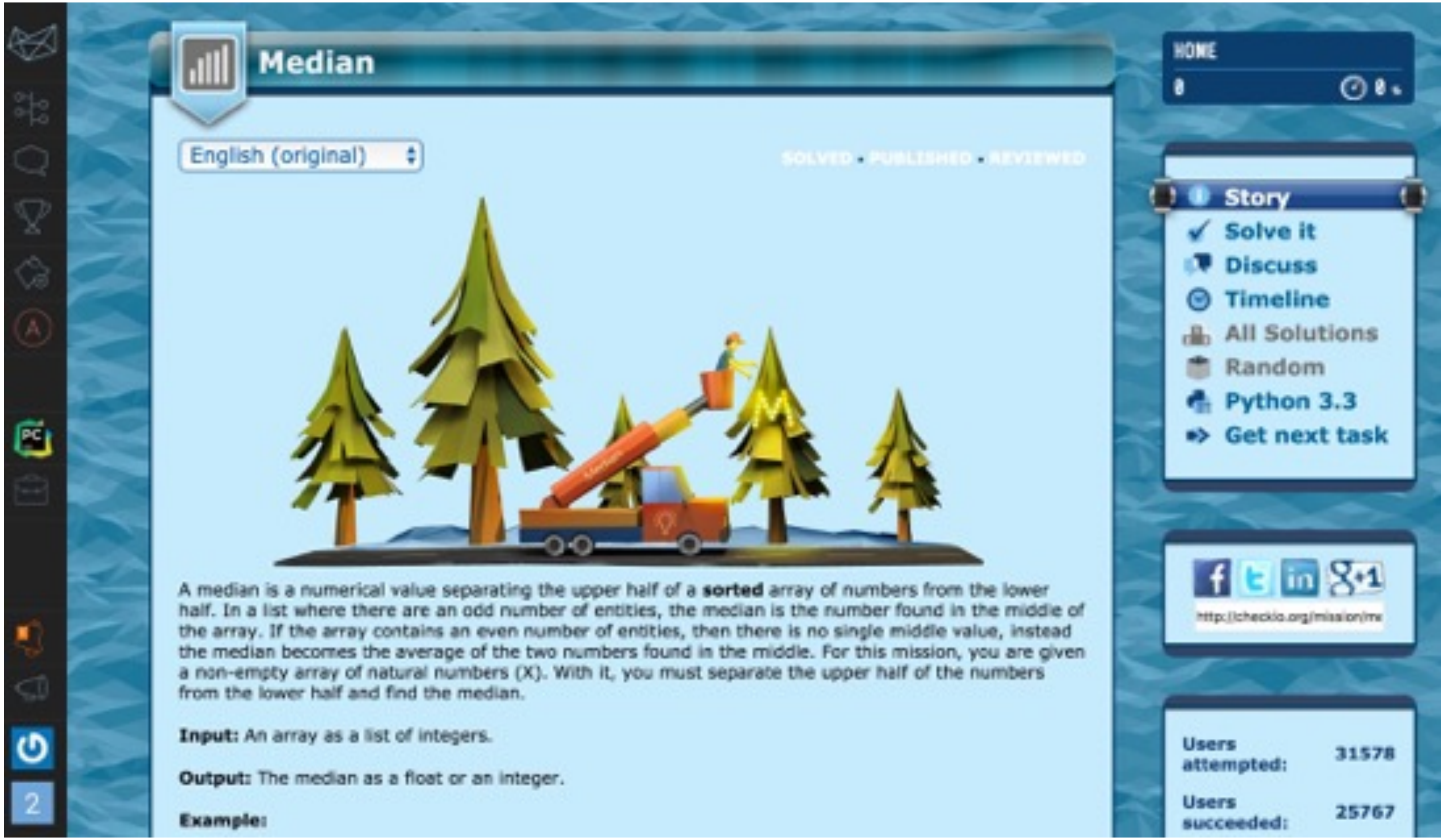
Check the strength of your favorite password

Elementary

text

Moore Neighbourhood SOLVED • PUBLISHED • REVIEWED

CheckiO



Median

A median is a numerical value separating the upper half of a **sorted** array of numbers from the lower half. In a list where there are an odd number of entities, the median is the number found in the middle of the array. If the array contains an even number of entities, then there is no single middle value, instead the median becomes the average of the two numbers found in the middle. For this mission, you are given a non-empty array of natural numbers (X). With it, you must separate the upper half of the numbers from the lower half and find the median.

Input: An array as a list of integers.

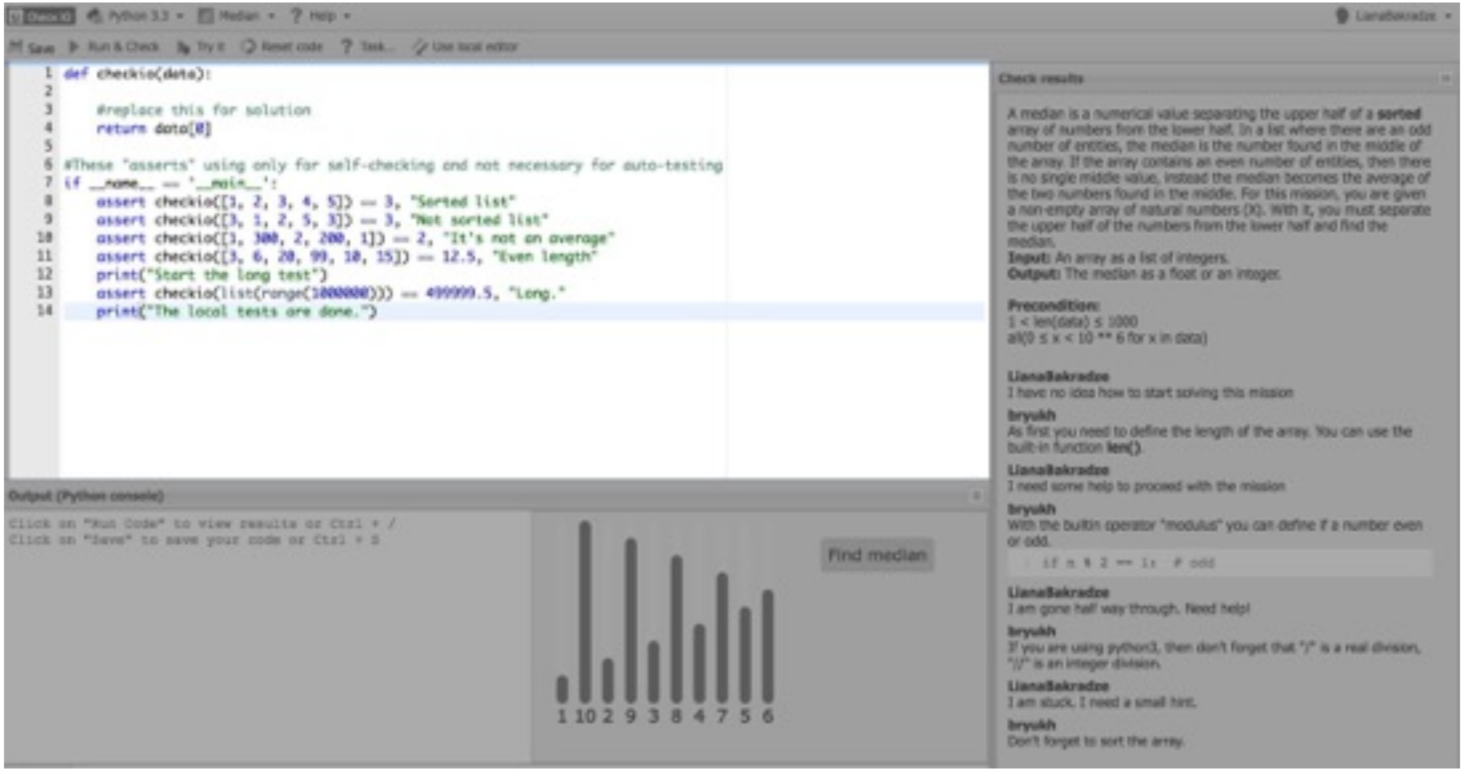
Output: The median as a float or an integer.

Example:

Users attempted: 31578

Users succeeded: 25767

CheckiO



```python
def checkio(data):

    #replace this for solution
    return data[0]

#These "asserts" using only for self-checking and not necessary for auto-testing
if __name__ == '__main__':
    assert checkio([1, 2, 3, 4, 5]) == 3, "Sorted list"
    assert checkio([3, 1, 2, 5, 3]) == 3, "Not sorted list"
    assert checkio([1, 300, 2, 200, 1]) == 2, "It's not an average"
    assert checkio([3, 6, 20, 99, 10, 15]) == 12.5, "Even length"
    print("Start the long test")
    assert checkio(list(range(1000000))) == 499999.5, "Long."
    print("The local tests are done.")
```

```
Click on "Run Code" to view results or Ctrl + /
Click on "Save" to save your code or Ctrl + S
```

A median is a numerical value separating the upper half of a **sorted** array of numbers from the lower half. In a list where there are an odd number of entities, the median is the number found in the middle of the array. If the array contains an even number of entities, then there is no single middle value, instead the median becomes the average of the two numbers found in the middle. For this mission, you are given a non-empty array of natural numbers (X). With it, you must seperate the upper half of the numbers from the lower half and find the median.
Input: An array as a list of integers.
Output: The median as a float or an integer.

Precondition:
1 < len(data) ≤ 1000
all(0 ≤ x < 10 ** 6 for x in data)

LianaBakradze
I have no idea how to start solving this mission

bryukh
As first you need to define the length of the array. You can use the built-in function **len()**.

LianaBakradze
I need some help to proceed with the mission

bryukh
With the builtin operator "modulus" you can define if a number even or odd.

```
if n % 2 == 1:  # odd
```

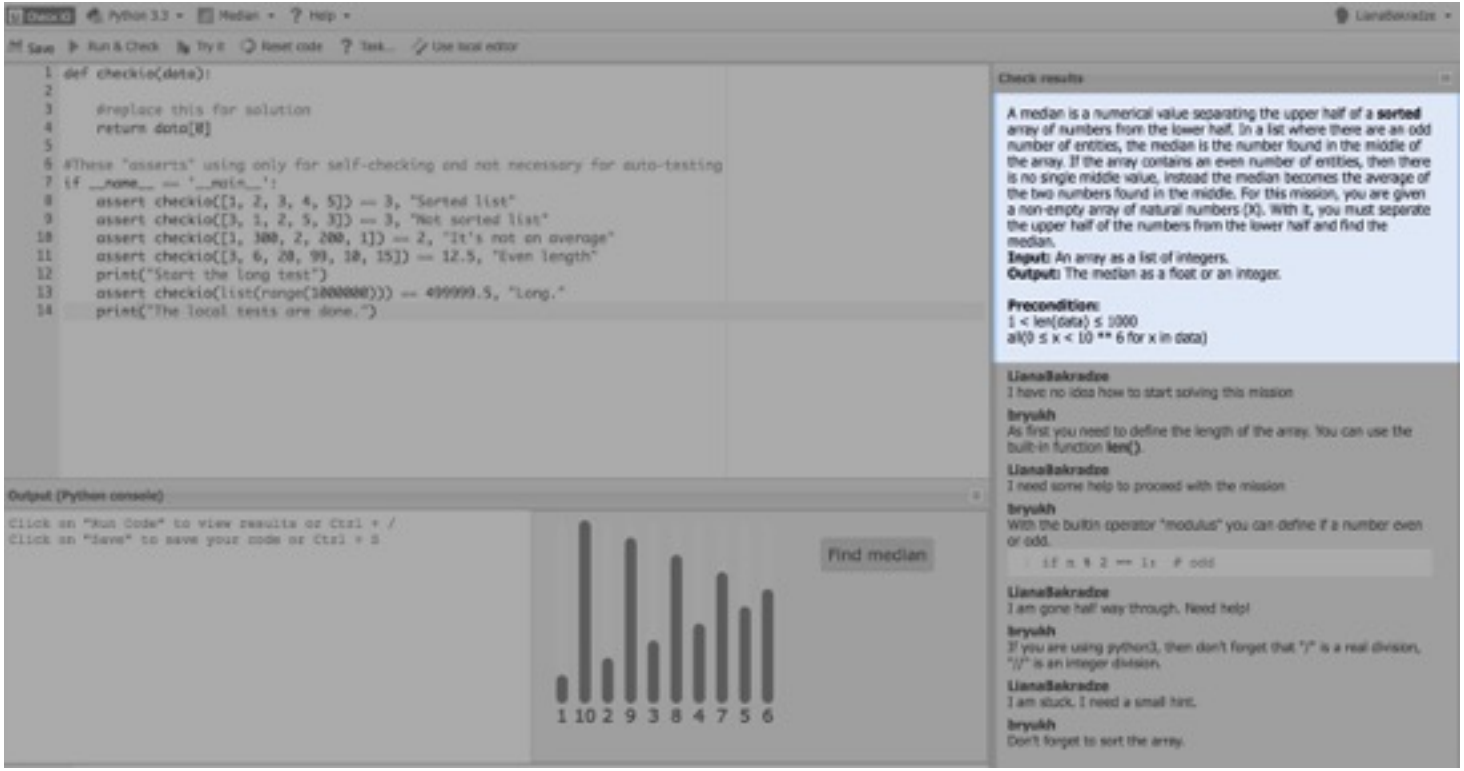
LianaBakradze
I am gone half way through. Need help!

bryukh
If you are using python3, then don't forget that "/" is a real division, "//" is an integer division.

LianaBakradze
I am stuck. I need a small hint.

bryukh
Don't forget to sort the array.

CheckiO



```
def checkio(data):

    #replace this for solution
    return data[0]

#These "asserts" using only for self-checking and not necessary for auto-testing
if __name__ == '__main__':
    assert checkio([1, 2, 3, 4, 5]) == 3, "Sorted list"
    assert checkio([3, 1, 2, 5, 3]) == 3, "Not sorted list"
    assert checkio([1, 300, 2, 200, 1]) == 2, "It's not an average"
    assert checkio([3, 6, 20, 99, 10, 15]) == 12.5, "Even length"
    print("Start the long test")
    assert checkio(list(range(1000000))) == 499999.5, "Long."
    print("The local tests are done.")
```

```
Click on "Run Code" to view results or Ctrl + /
Click on "Save" to save your code or Ctrl + S
```

A median is a numerical value separating the upper half of a **sorted** array of numbers from the lower half. In a list where there are an odd number of entities, the median is the number found in the middle of the array. If the array contains an even number of entities, then there is no single middle value, instead the median becomes the average of the two numbers found in the middle. For this mission, you are given a non-empty array of natural numbers (X). With it, you must seperate the upper half of the numbers from the lower half and find the median.

Input: An array as a list of integers.

Output: The median as a float or an integer.

Precondition:
1 < len(data) ≤ 1000
all(0 ≤ x < 10 ** 6 for x in data)

LianaBakradze
I have no idea how to start solving this mission

bryukh
As first you need to define the length of the array. You can use the built-in function **len()**.

LianaBakradze
I need some help to proceed with the mission

bryukh
With the builtin operator "modulus" you can define if a number even or odd.

```
if n % 2 == 1:  # odd
```

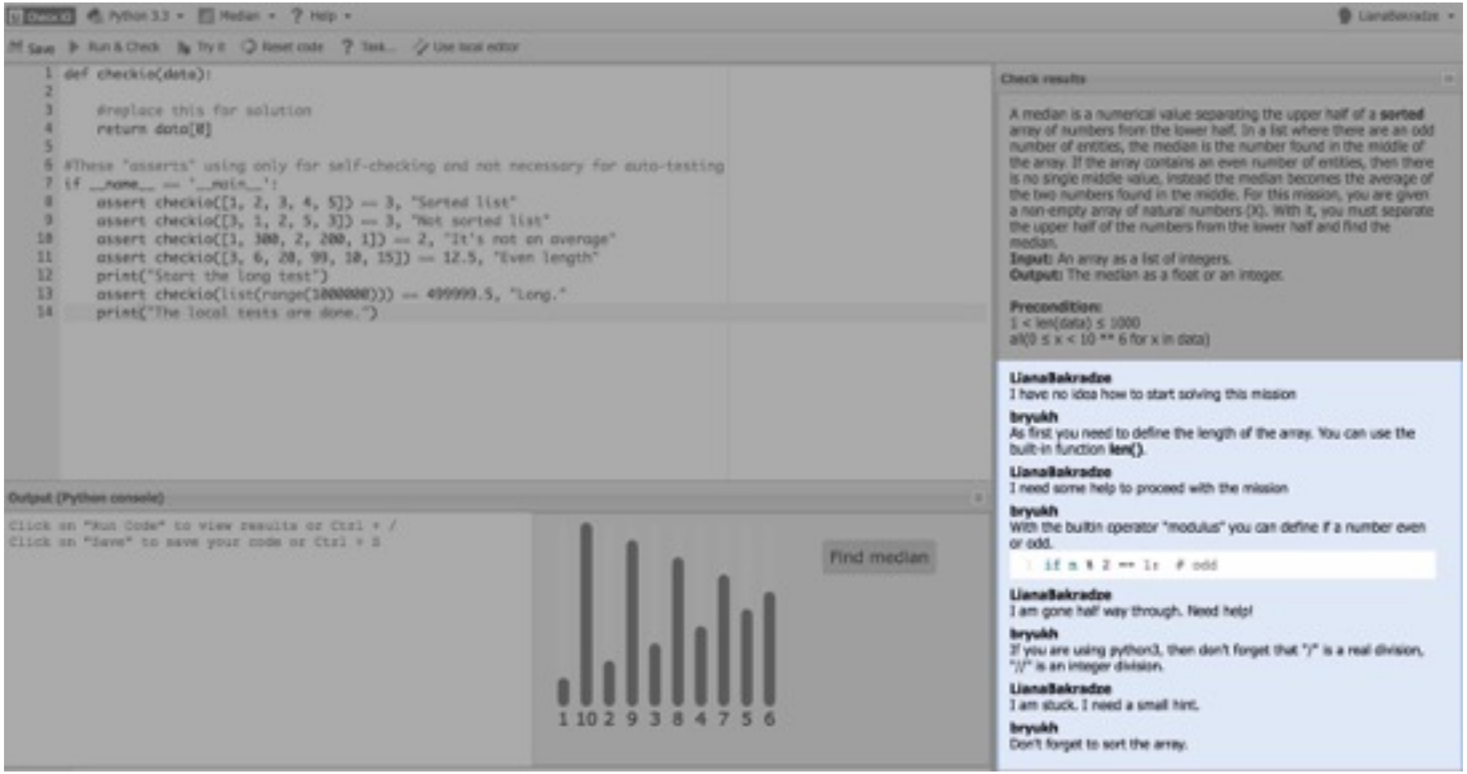
LianaBakradze
I am gone half way through. Need help!

bryukh
If you are using python3, then don't forget that "/" is a real division, "//" is an integer division.

LianaBakradze
I am stuck. I need a small hint.

bryukh
Don't forget to sort the array.

CheckiO



```
def checkio(data):

    #replace this for solution
    return data[0]

#These "asserts" using only for self-checking and not necessary for auto-testing
if __name__ == '__main__':
    assert checkio([1, 2, 3, 4, 5]) == 3, "Sorted list"
    assert checkio([3, 1, 2, 5, 3]) == 3, "Not sorted list"
    assert checkio([1, 300, 2, 200, 1]) == 2, "It's not an average"
    assert checkio([3, 6, 20, 99, 10, 15]) == 12.5, "Even length"
    print("Start the long test")
    assert checkio(list(range(1000000))) == 499999.5, "Long."
    print("The local tests are done.")
```

```
Click on "Run Code" to view results or Ctrl + /
Click on "Save" to save your code or Ctrl + S
```

A median is a numerical value separating the upper half of a **sorted** array of numbers from the lower half. In a list where there are an odd number of entities, the median is the number found in the middle of the array. If the array contains an even number of entities, then there is no single middle value, instead the median becomes the average of the two numbers found in the middle. For this mission, you are given a non-empty array of natural numbers (X). With it, you must seperate the upper half of the numbers from the lower half and find the median.

Input: An array as a list of integers.
Output: The median as a float or an integer.

Precondition:
1 < len(data) ≤ 1000
all(0 ≤ x < 10 ** 6 for x in data)

LianaBakradze
I have no idea how to start solving this mission

bryukh
As first you need to define the length of the array. You can use the built-in function **len()**.

LianaBakradze
I need some help to proceed with the mission

bryukh
With the builtin operator "modulus" you can define if a number even or odd.

```
if n % 2 == 1:  # odd
```

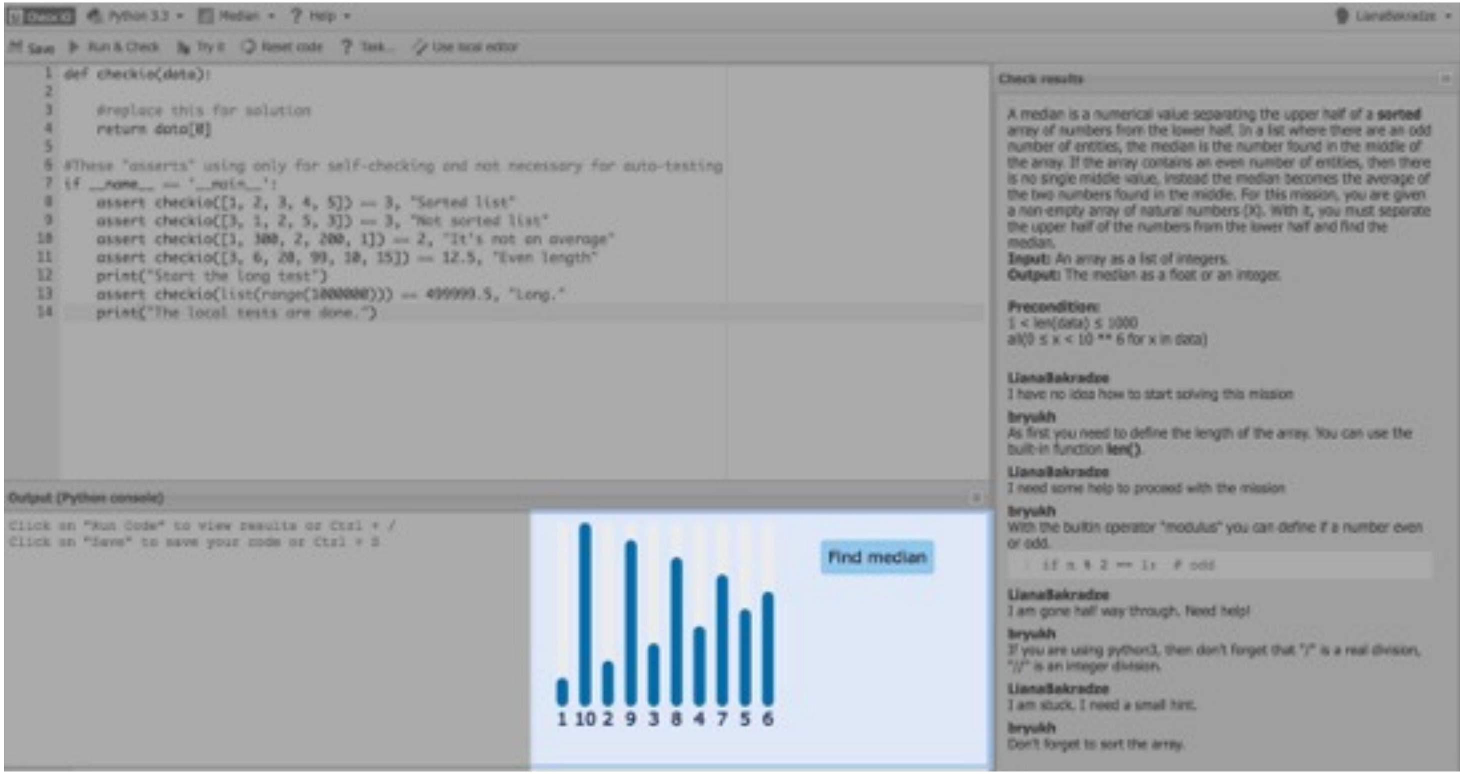
LianaBakradze
I am gone half way through. Need help!

bryukh
If you are using python3, then don't forget that "/" is a real division, "//" is an integer division.

LianaBakradze
I am stuck. I need a small hint.

bryukh
Don't forget to sort the array.

CheckiO



```
def checkio(data):

    #replace this for solution
    return data[0]

#These "asserts" using only for self-checking and not necessary for auto-testing
if __name__ == '__main__':
    assert checkio([1, 2, 3, 4, 5]) == 3, "Sorted list"
    assert checkio([3, 1, 2, 5, 3]) == 3, "Not sorted list"
    assert checkio([1, 300, 2, 200, 1]) == 2, "It's not an average"
    assert checkio([3, 6, 20, 99, 10, 15]) == 12.5, "Even length"
    print("Start the long test")
    assert checkio(list(range(1000000))) == 499999.5, "Long."
    print("The local tests are done.")
```

```
Click on "Run Code" to view results or Ctrl + /
Click on "Save" to save your code or Ctrl + S
```

A median is a numerical value separating the upper half of a **sorted** array of numbers from the lower half. In a list where there are an odd number of entities, the median is the number found in the middle of the array. If the array contains an even number of entities, then there is no single middle value, instead the median becomes the average of the two numbers found in the middle. For this mission, you are given a non-empty array of natural numbers (X). With it, you must seperate the upper half of the numbers from the lower half and find the median.

Input: An array as a list of integers.

Output: The median as a float or an integer.

Precondition:
1 < len(data) ≤ 1000
all(0 ≤ x < 10 ** 6 for x in data)

LianaBakradze
I have no idea how to start solving this mission

bryukh
As first you need to define the length of the array. You can use the built-in function **len()**.

LianaBakradze
I need some help to proceed with the mission

bryukh
With the builtin operator "modulus" you can define if a number even or odd.

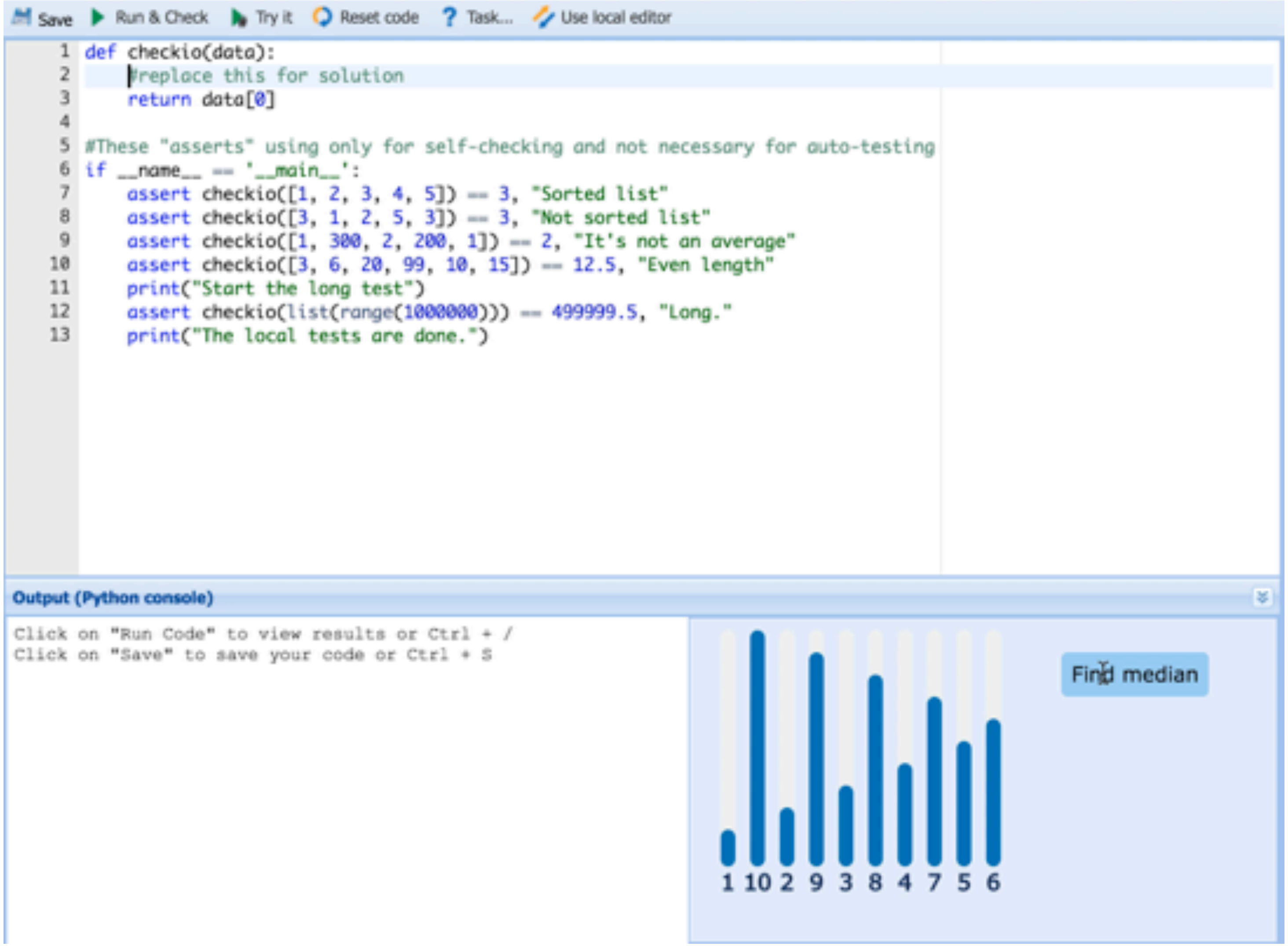
```
if n % 2 == 1:  # odd
```

LianaBakradze
I am gone half way through. Need help!

bryukh
If you are using python3, then don't forget that "/" is a real division, "//" is an integer division.

LianaBakradze
I am stuck. I need a small hint.

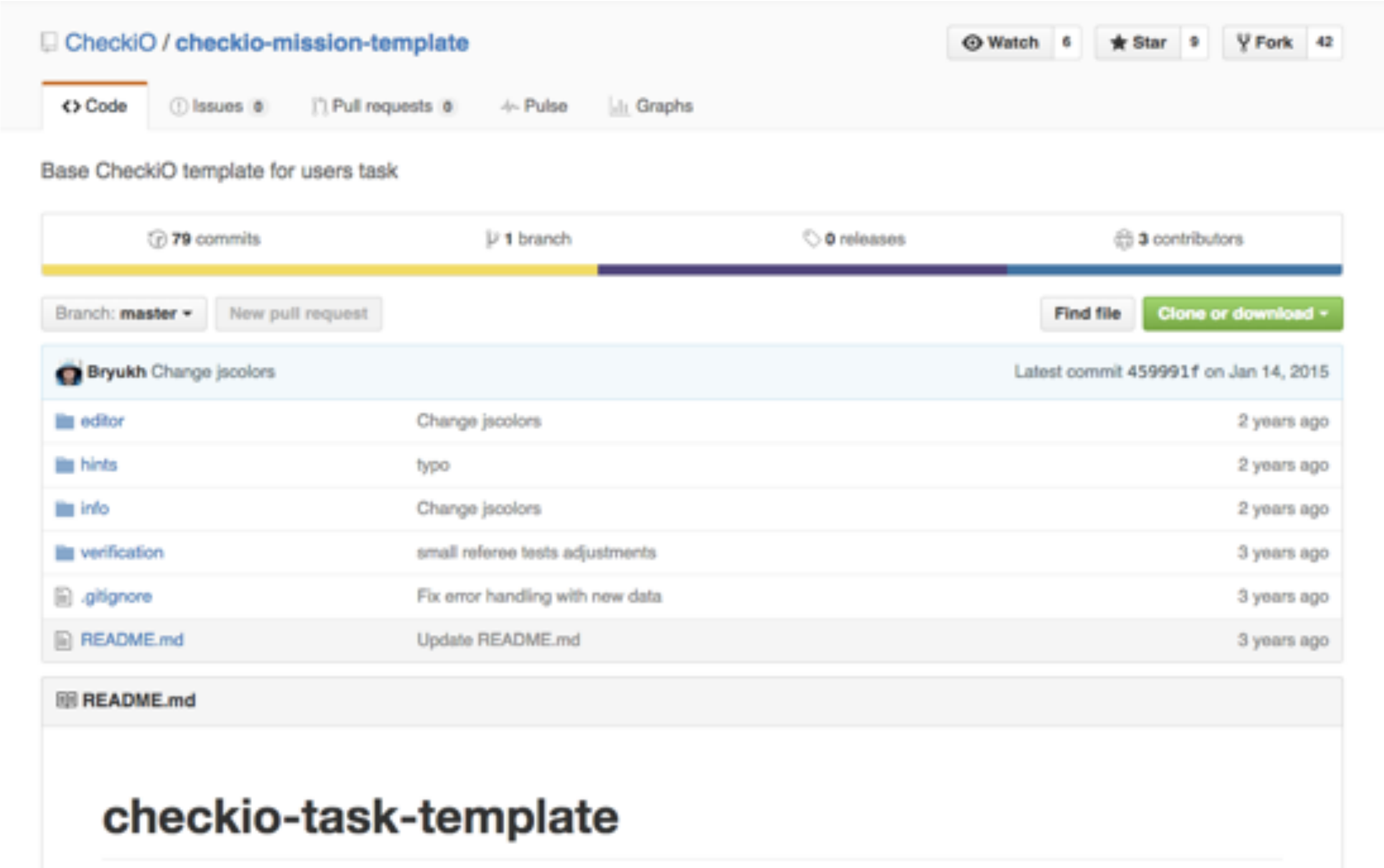
bryukh
Don't forget to sort the array.



```python
def checkio(data):
    #replace this for solution
    return data[0]

#These "asserts" using only for self-checking and not necessary for auto-testing
if __name__ == '__main__':
    assert checkio([1, 2, 3, 4, 5]) == 3, "Sorted list"
    assert checkio([3, 1, 2, 5, 3]) == 3, "Not sorted list"
    assert checkio([1, 300, 2, 200, 1]) == 2, "It's not an average"
    assert checkio([3, 6, 20, 99, 10, 15]) == 12.5, "Even length"
    print("Start the long test")
    assert checkio(list(range(1000000))) == 499999.5, "Long."
    print("The local tests are done.")
```

```
Click on "Run Code" to view results or Ctrl + /
Click on "Save" to save your code or Ctrl + S
```

CheckiO



Base CheckiO template for users task

Bryukh Change jscolors   Latest commit 459991f on Jan 14, 2015

editor	Change jscolors	2 years ago
hints	typo	2 years ago
info	Change jscolors	2 years ago
verification	small referee tests adjustments	3 years ago
.gitignore	Fix error handling with new data	3 years ago
README.md	Update README.md	3 years ago

README.md

checkio-task-template

Conclusion

- 3 great projects
- try yourself
- contribute
- invent something cool

image: www.freeimages.co.uk

Thank You!

Liana Bakradze

liana.bakradze@jetbrains.com
@med_vector