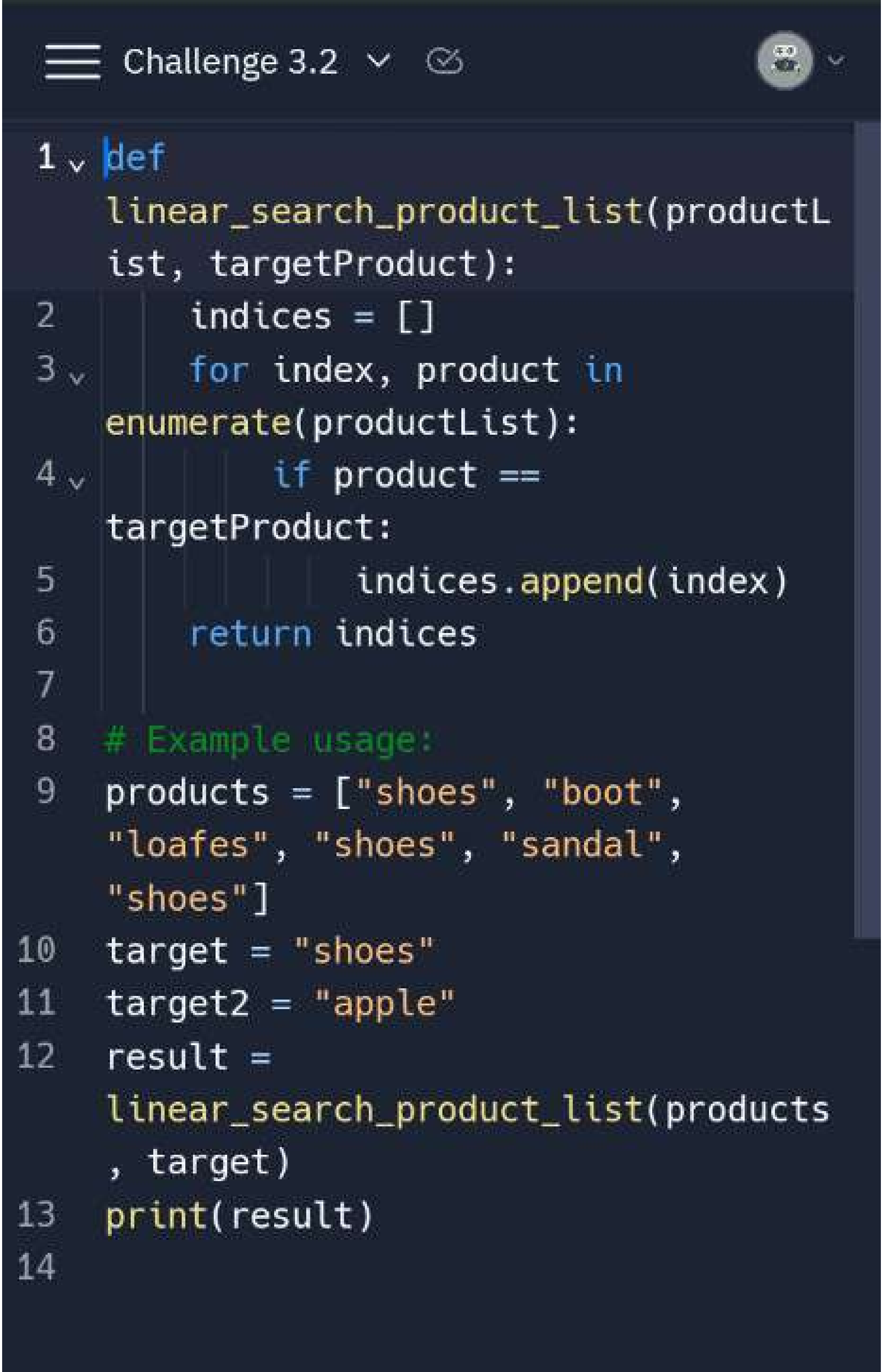
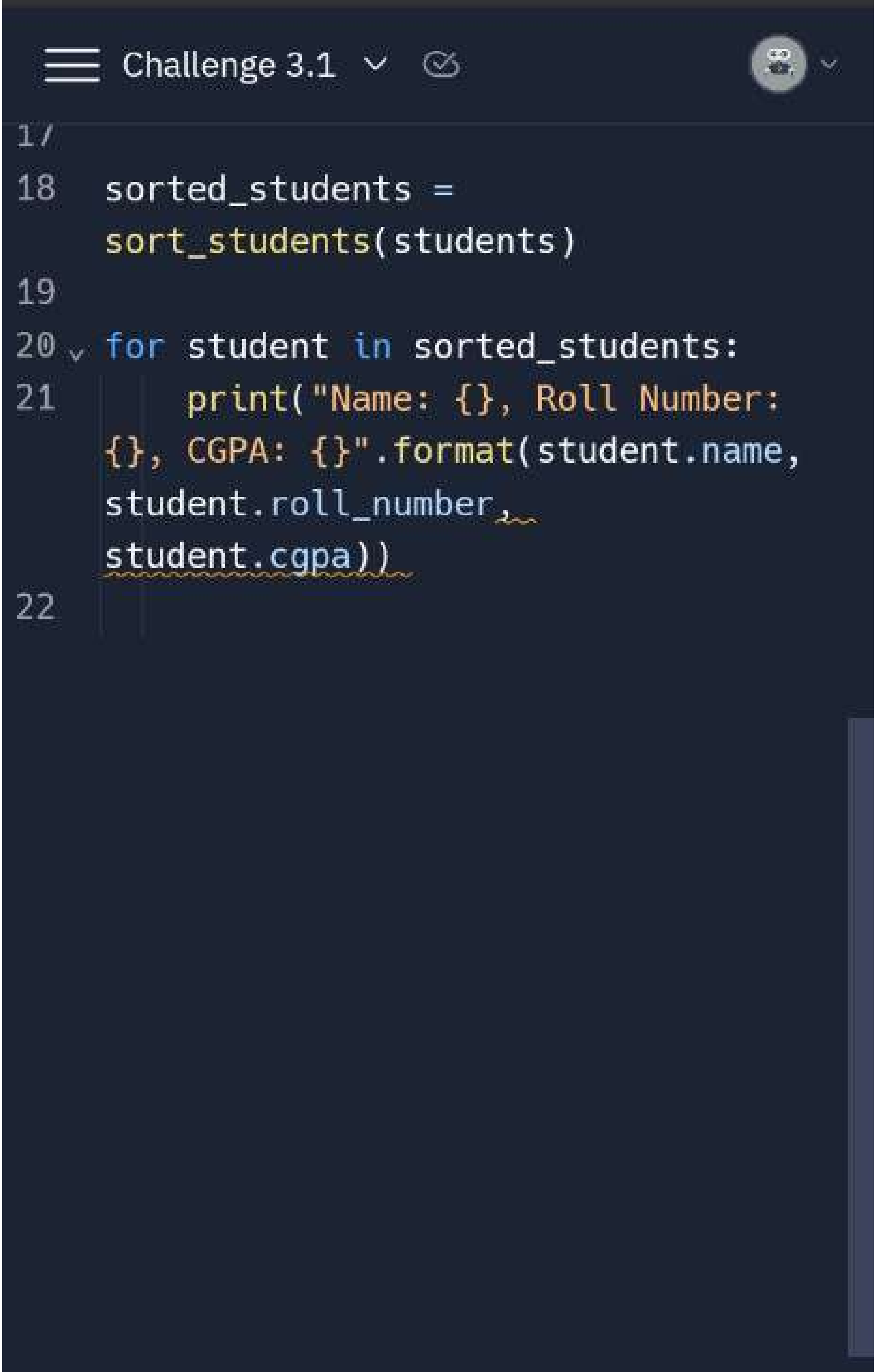
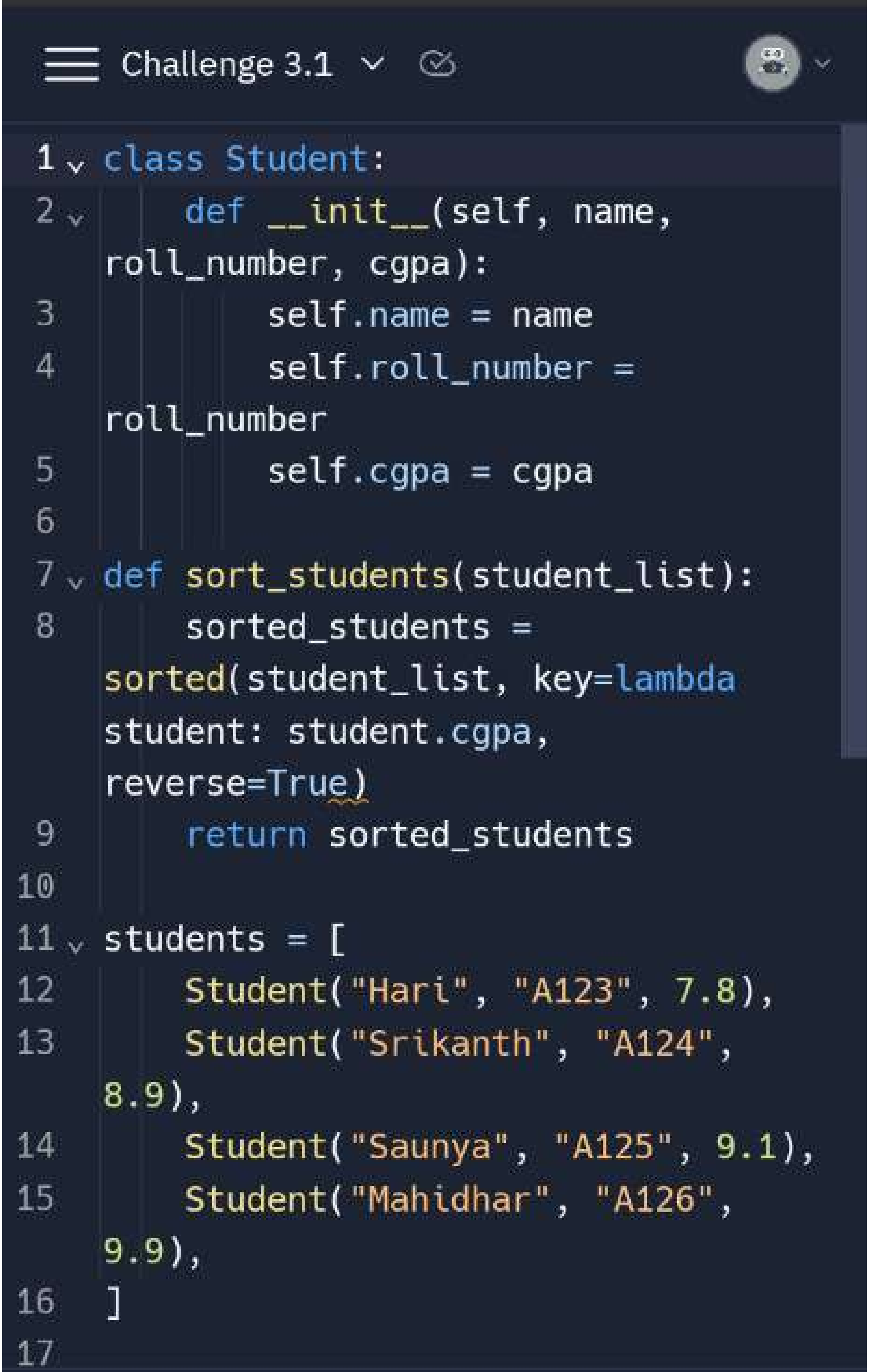

Challenge 3.2

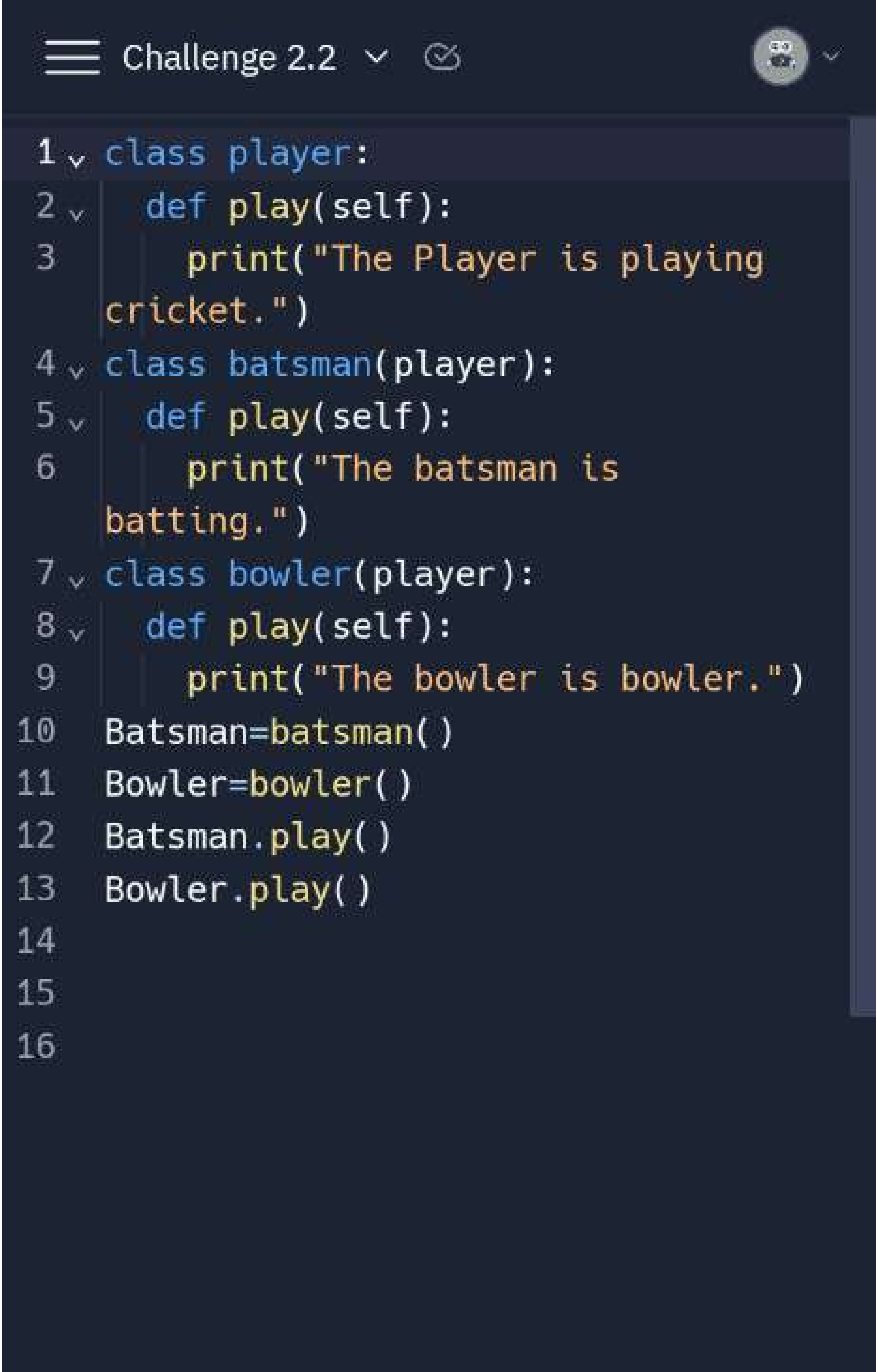
```
def linear_search_product_list(productList, targetProduct):
    indices = []
    for index, product in enumerate(productList):
        if product == targetProduct:
            indices.append(index)
    return indices

# Example usage:
products = ["shoes", "boot", "loafes", "shoes", "sandal", "shoes"]
target = "shoes"
target2 = "apple"
result = linear_search_product_list(products, target)
print(result)
```



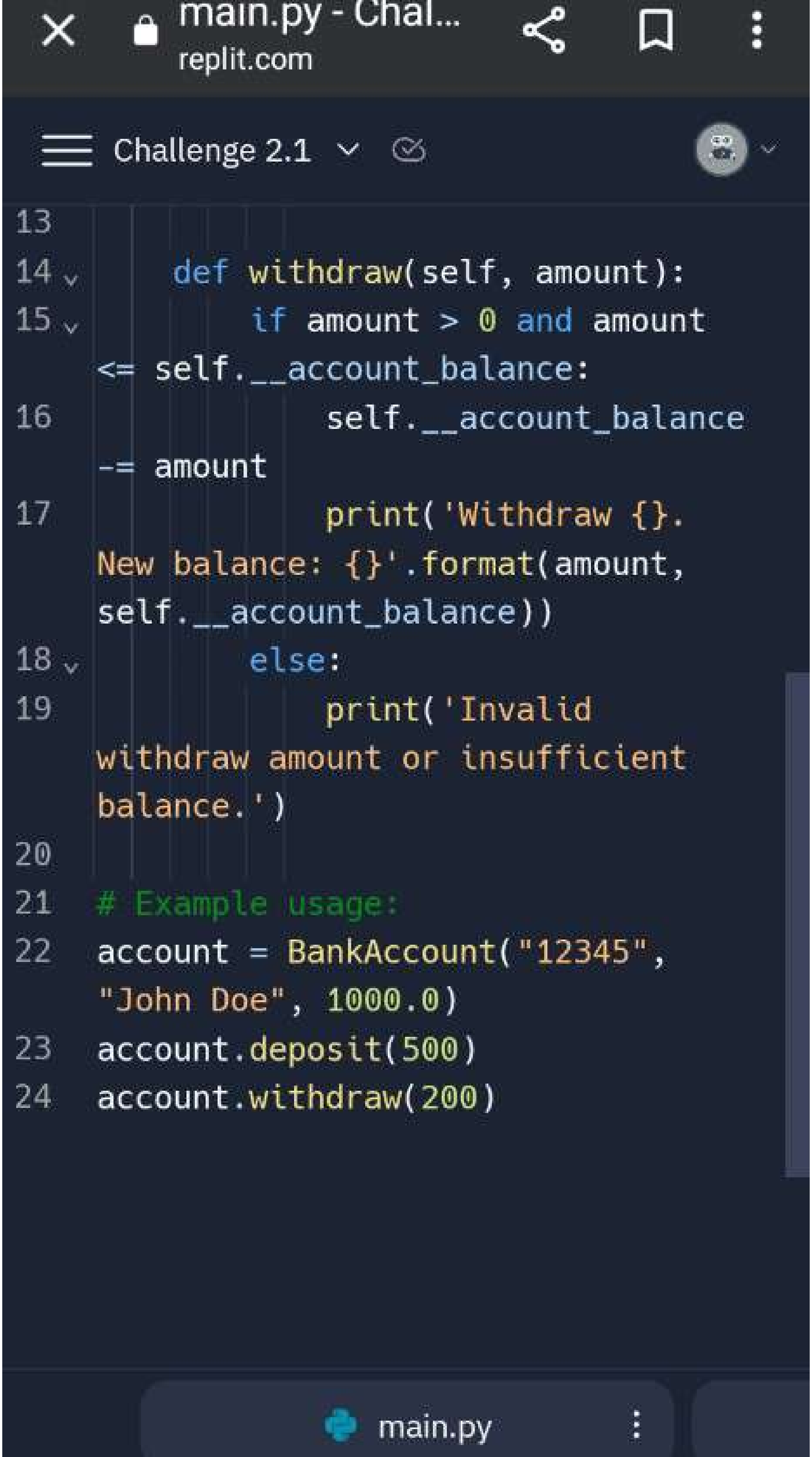
```python
sorted_students = sort_students(students)

for student in sorted_students:
    print("Name: {}, Roll Number: {}, CGPA: {}".format(student.name, student.roll_number, student.cgpa))

```



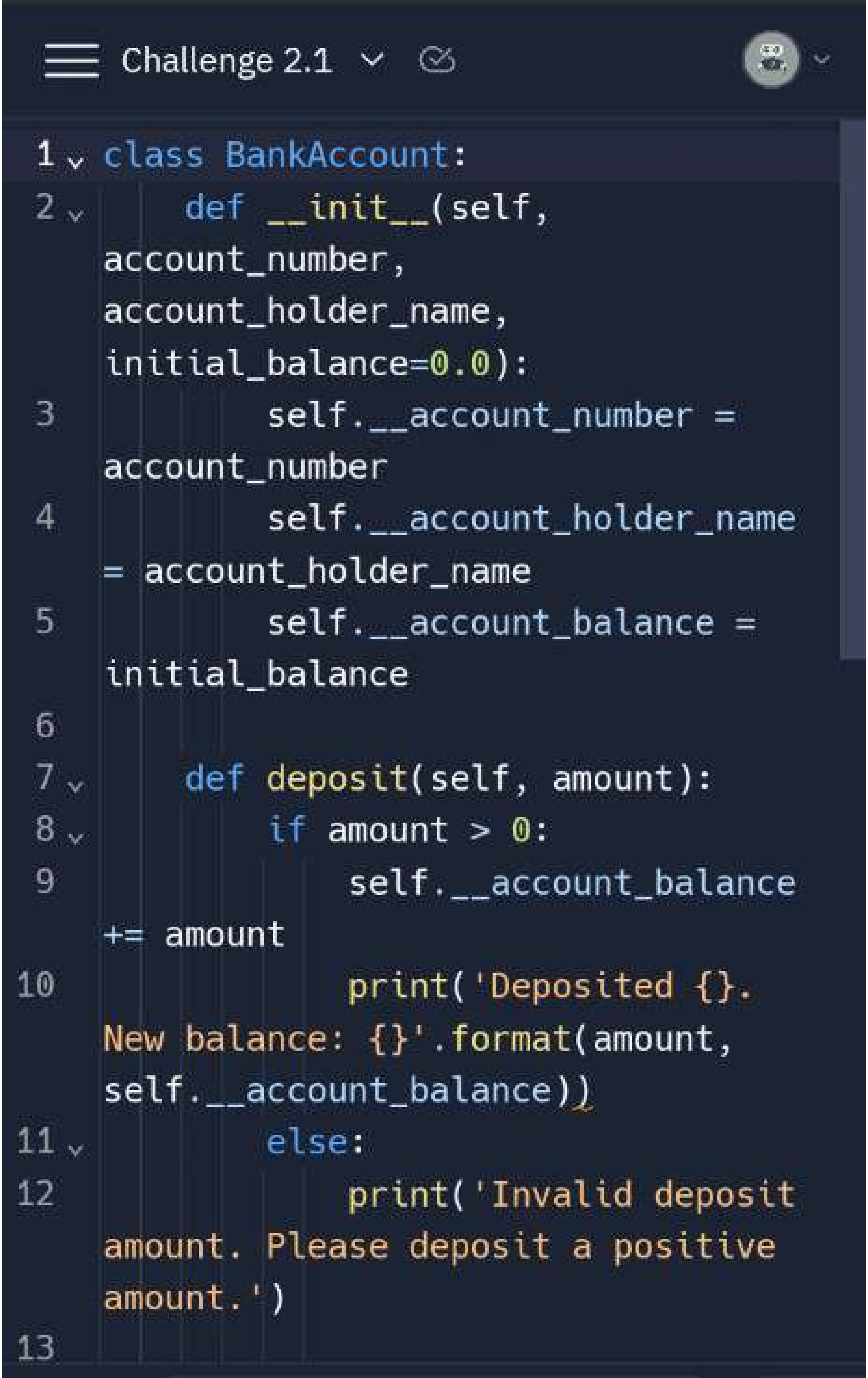
```python
class Student:
    def __init__(self, name, roll_number, cgpa):
        self.name = name
        self.roll_number = roll_number
        self.cgpa = cgpa

def sort_students(student_list):
    sorted_students = sorted(student_list, key=lambda student: student.cgpa, reverse=True)
    return sorted_students

students = [
    Student("Hari", "A123", 7.8),
    Student("Srikanth", "A124", 8.9),
    Student("Saunya", "A125", 9.1),
    Student("Mahidhar", "A126", 9.9),
]

```


Challenge 2.2

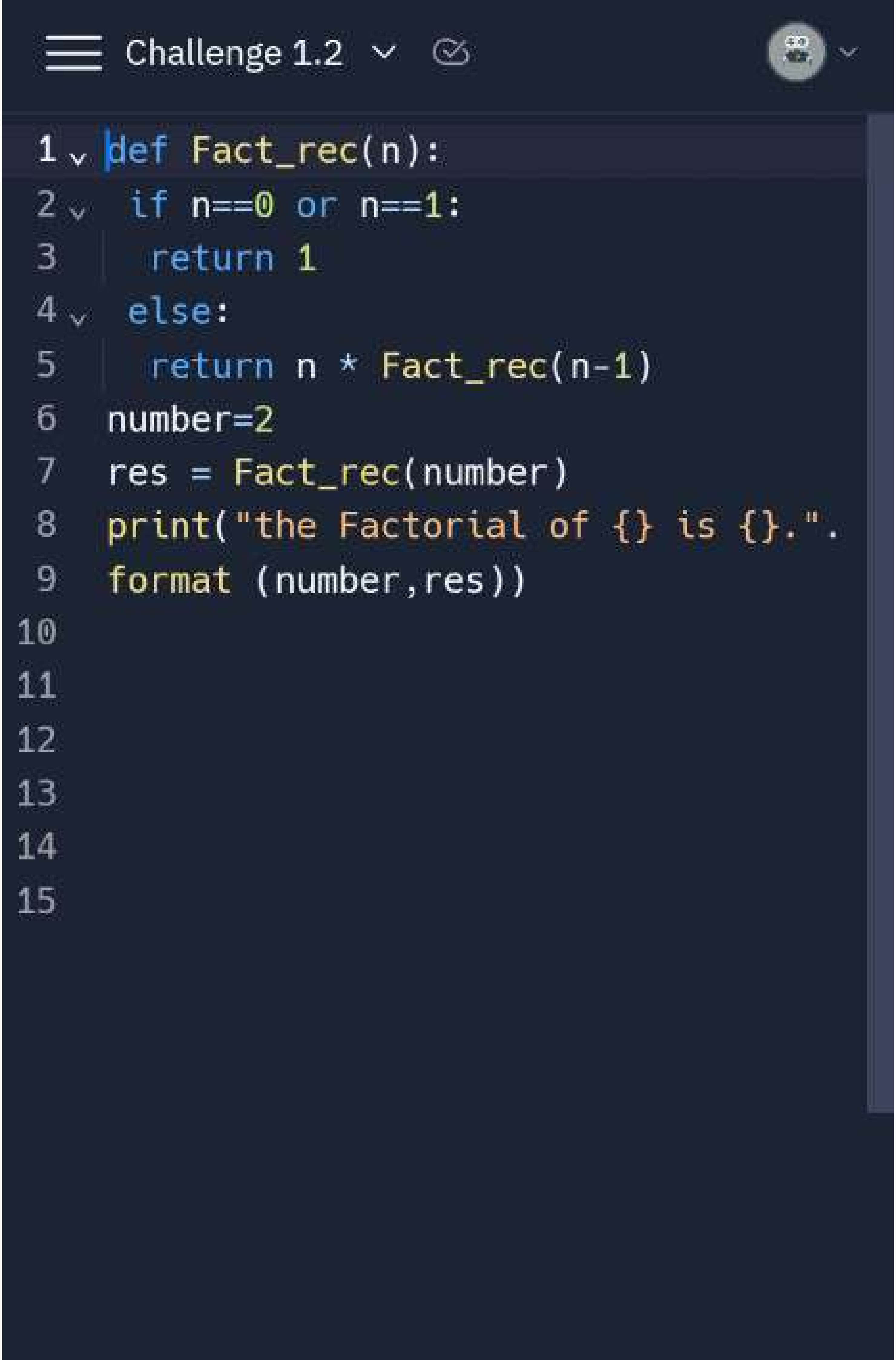
```
class player:
  def play(self):
    print("The Player is playing cricket.")
class batsman(player):
  def play(self):
    print("The batsman is batting.")
class bowler(player):
  def play(self):
    print("The bowler is bowler.")
Batsman=batsman()
Bowler=bowler()
Batsman.play()
Bowler.play()
```



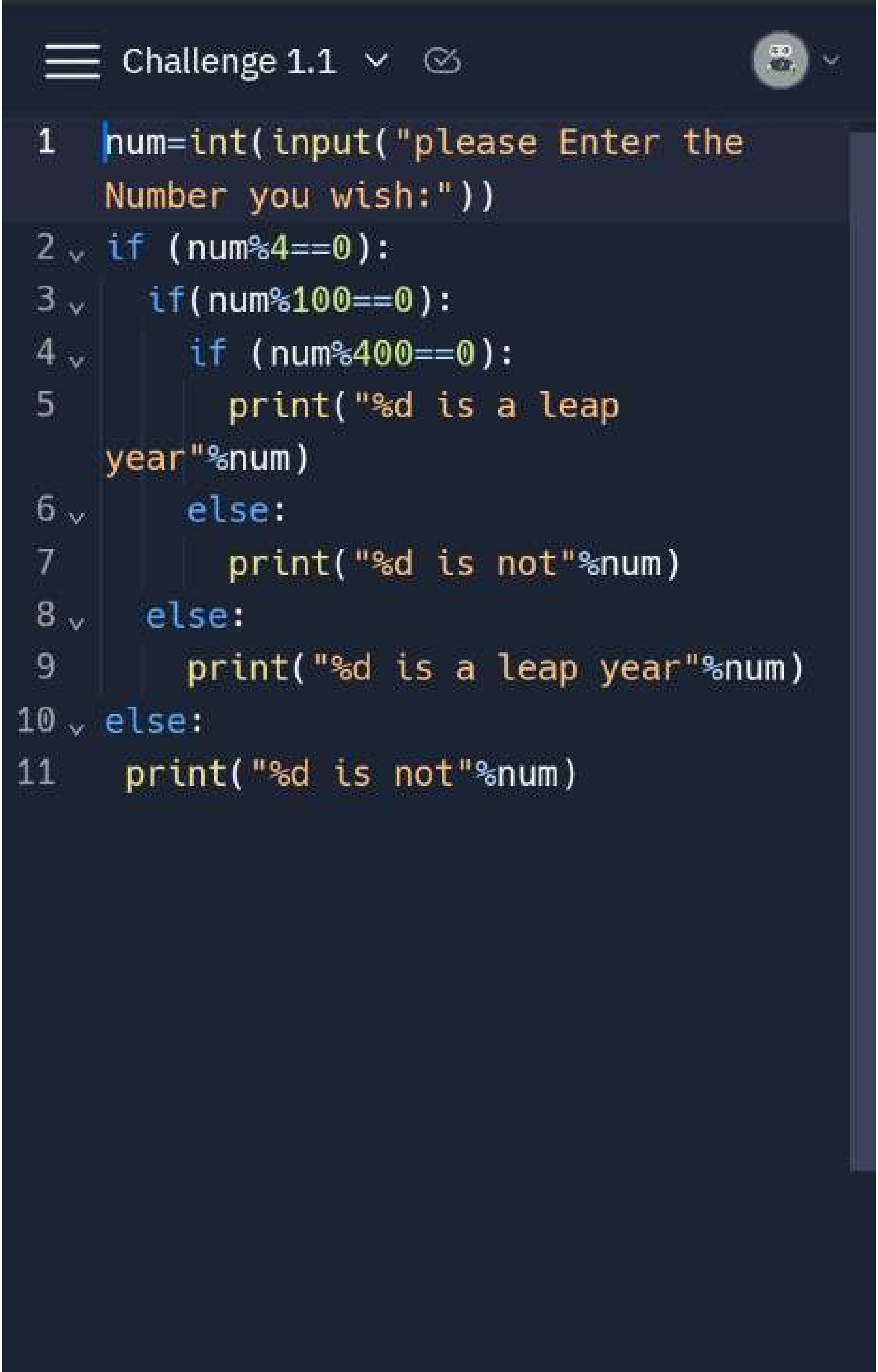
```python

    def withdraw(self, amount):
        if amount > 0 and amount <= self.__account_balance:
            self.__account_balance -= amount
            print('Withdraw {}. New balance: {}'.format(amount, self.__account_balance))
        else:
            print('Invalid withdraw amount or insufficient balance.')

# Example usage:
account = BankAccount("12345", "John Doe", 1000.0)
account.deposit(500)
account.withdraw(200)
```



```python
class BankAccount:
    def __init__(self, account_number, account_holder_name, initial_balance=0.0):
        self.__account_number = account_number
        self.__account_holder_name = account_holder_name
        self.__account_balance = initial_balance

    def deposit(self, amount):
        if amount > 0:
            self.__account_balance += amount
            print('Deposited {}. New balance: {}'.format(amount, self.__account_balance))
        else:
            print('Invalid deposit amount. Please deposit a positive amount.')

```



```python
def Fact_rec(n):
 if n==0 or n==1:
  return 1
 else:
  return n * Fact_rec(n-1)
number=2
res = Fact_rec(number)
print("the Factorial of {} is {}.".
format (number,res))
```



```python
num=int(input("please Enter the Number you wish:"))
if (num%4==0):
  if(num%100==0):
    if (num%400==0):
      print("%d is a leap year"%num)
    else:
      print("%d is not"%num)
  else:
    print("%d is a leap year"%num)
else:
 print("%d is not"%num)
```