RUNNING TESTS FOR ASSIGNMENT SHEET06

STUDENT IS Haag

ATTEMPTED TASKS ARE Ausgabe.py Primzahlen.txt JedeZweite.py

STARTING TESTING Ausgabe.py

THE ATTEMPTED SOLUTION IS

```
f=open("Ausgabe.py", "r")
for n in f.readlines():
    print(n,end="")
f.close()
```

THE MASTER SOLUTION IS

```
with open("Ausgabe.py", "r") as f:
  for line in f.readlines():
    print(line,end="")
```

THE INPUT DATA CONFIG FOR THIS FILE IS:

```
{ }
```

TEST RUN

chosen input

Das Programm soll sich bei Ausführung selbst ausgeben, braucht also keinen expliziten Input.

your output

```
celinemueller@MacBook-Air-Buro sheet06 % python3 Ausgabe.py
f=open("Ausgabe.py", "r")
for n in f.readlines():
    print(n,end="")
f.close() %
celinemueller@MacBook-Air-Buro sheet06 %
```

expected output

```
celinemueller@Air-Buro sheet06 % python3 Ausgabe.py
with open("Ausgabe.py", "r") as f:
  for line in f.readlines():
    print(line,end="")%
```

COMMENTS ON YOUR SOLUTION:

- die Datei wurde manuell geöffnet (ohne with open - Scope) - aber auch daran gedacht, sie wieder zu schliessen +1
- Die zusätzlichen Zeilenumbrüche werden durch das manuelle Setzen von `end=` entgegengewirkt. +1

STARTING TESTING Primzahlen.txt

THE ATTEMPTED SOLUTION IS

```
2
3
5
7
```

```
11
13
# ...
3539
3541
3547
3557
3559
3571
3581
```

THE MASTER SOLUTION IS

```
2
3
5
7
11
13
# ...
3539
3541
3547
3557
3559
3571
```

THE INPUT DATA CONFIG FOR THIS FILE IS:

```
{
    "compare_with": "Primzahlen_Master.txt"
}
```

TEST RUN

txt submission are not run.

COMMENTS ON YOUR SOLUTION:

Du hast die richtigen Primzahlen ausgegeben, allerdings eine zu viel! Überprüfe nochmal in dem Code, mit dem du die Zahlen erzeugt hast, wie es genau 500 werden können.

STARTING TESTING JedeZweite.py

THE ATTEMPTED SOLUTION IS

```
import sys

a=[]
for zeile in sys.stdin:
    a.append(zeile)
print(a)

i=0
for b in a:
    if i%2==0:
        print(b)
    i+=1
```

THE MASTER SOLUTION IS

```
import sys

a = sys.stdin.readlines()
```

```
for i in range(1,len(a),2):
  print(a[i],end="")
```

THE INPUT DATA CONFIG FOR THIS FILE IS:

```
{
    "stream": [
        {
            "type": "str",
            "repeatable": true,
            "min_repeats": 1,
            "max_repeats": 20,
            "requires_eof": true
        }
    ]
}
```

TEST RUN

chosen input

"I
input
a
stream
end
with
ctrl-d"
das sind Strings, die ich über die Standardeingabe eingeben kann.

your output

```
celinemueller@MacBook-Air-Buro sheet06 % python3 JedeZweite.py
I
input
a
stream
end
with
ctrl-d
['I\n', 'input\n', 'a \n', 'stream\n', 'end\n', 'with\n', 'ctrl-d\n']
I

a

end

ctrl-d

celinemueller@MacBook-Air-Buro sheet06 %
```

expected output

```
celinemueller@Air-Buro sheet06 % python3 JedeZweite.py
I
input
a
stream
end
with
ctrl
d
input
stream
with
d
celinemueller@Air-Buro sheet06 %
```

COMMENTS ON YOUR SOLUTION:

- Du hast den Eingabestrom mit dem sys.stdin - Modul eingelesen +1
- Nach Einlesen des Stroms in eine Liste, wird die ganze Liste ausgegeben (print(a)). Dies wurde vielleicht zu debugging-Zwecken gemacht und vor Einreichen vergessen zu entfernen.

- Es wird mit der ersten statt der zweiten Zeile angefangen zu printen (nutze lieber `if i%2 != 0: print(b)` -1
- Du hast jede zweite Zeile des Eingabestroms wieder ausgegeben +1
- readlines() fügt am Ende jeder Zeile (jedes Strings) ein "\n" ein. print() fügt aber auch am Ende jeder Ausgabe ein "\n" ein, die eingelesenen Zeilen werden also mit doppeltem Zeilenumbruch wieder ausgegeben. Vergleiche nochmal deine und die Musterlösung, und finde den kleinen Unterschied. -1
