

Informatikos fakultetas

**T120B162 Programų sistemų testavimas**

**2 laboratorinis darbas**

|  |  |  |
| --- | --- | --- |
|  | Studentas: | Ignas Jasonas IFF-6/6 |
| Dėstytojas: | doc. Šarūnas Packevičius |

Kaunas 2019

Turinys

[1. Testuojama sistema 3](#_Toc27523801)

[2. Unit testai 3](#_Toc27523802)

[2.1. resetButton 4](#_Toc27523803)

[2.2. guessedWord 4](#_Toc27523804)

[2.3. getMistake 4](#_Toc27523805)

[2.4. handleGuess 5](#_Toc27523806)

[2.5. getGameState 5](#_Toc27523807)

[2.6. getDisplayText 6](#_Toc27523808)

[2.7. randomWord 6](#_Toc27523809)

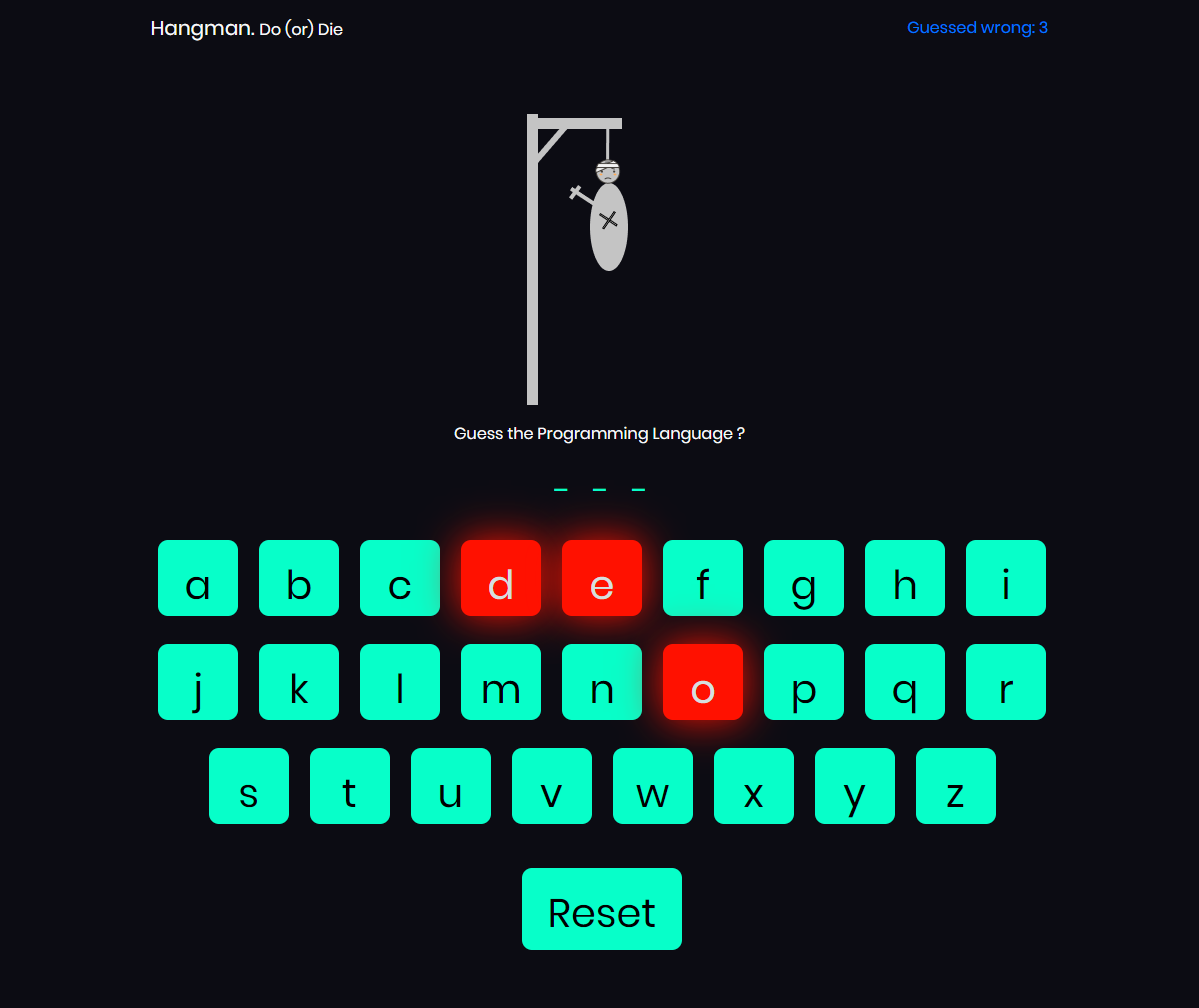
[3. Mock’ai 7](#_Toc27523810)

[4. Rezultatai 8](#_Toc27523811)

[5. Išvados 8](#_Toc27523812)

# Testuojama sistema

Testuojama sistema pasirinka React technologija realizuotas paprastas kartuvių žaidimas. Žaidimo tikslas – atspėti programavimo kalbą, spaudžiant ant spėjamą raidę. Projektas yra pateiktas kaip pavyzdys react github repozitorijoje. Testai realizuoti pasindauojant jest testavimo įrankiu.

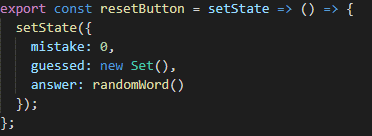


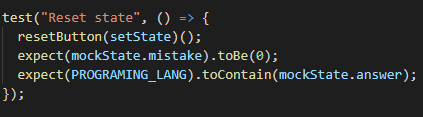
# Unit testai

Projekte buvo apie dešimt funkcijų, jas išsikėliau į atskirą failą, kad būtų patogiau testuoti ir sukūriau unit testus.

## resetButton

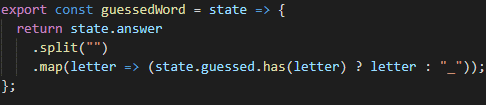
Ši funkcija suveikia paspaudus ant reset mygtuko. Jos esmė atstatyti būsenos objektą į pradinę būseną.

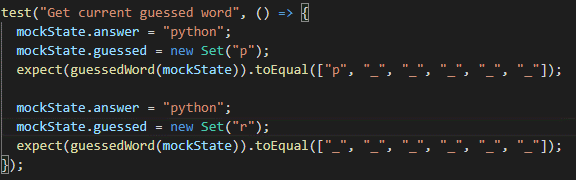




## guessedWord

Ši funkcija išskaido spėjamą žodį į raidžių masyvą ir pagal spėtas raides parodo kiek žodžio yra jau atspėta.

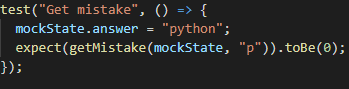




## getMistake

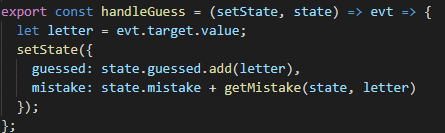
Ši funkcija pagal spėtą raidę, gauna reikiamą pridėti padarytų klaidų kiekį

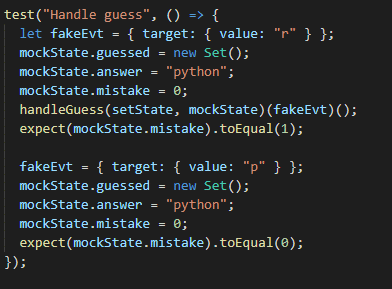




## handleGuess

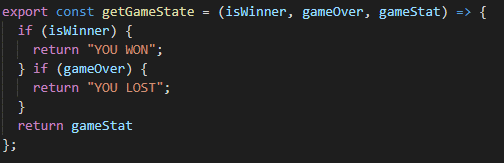
Ši funkcija apdoroja spėjimą, atnaujina spėjamą žodį ir gautą klaidų kiekį

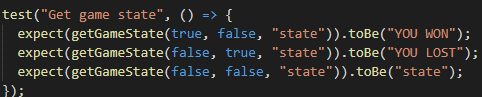




## getGameState

Ši funkcija gauna žaidimo statusą

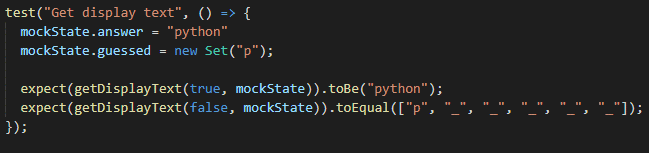




## getDisplayText

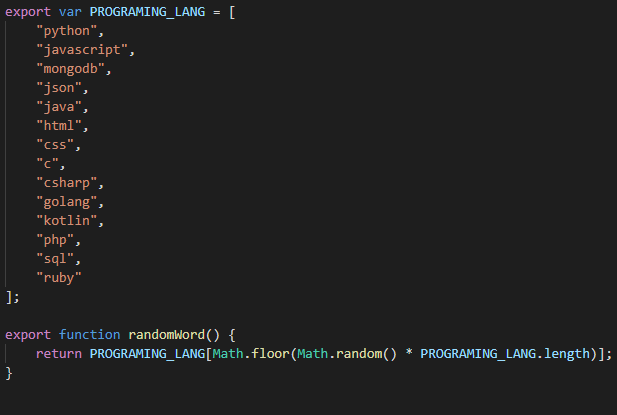
Ši funkcija gauna tekstą, kurį rodo ekrane

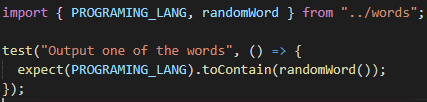




## randomWord

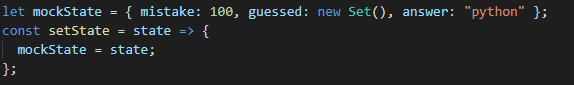
Ši funkcija iš programavimo kalbų masyvų išrenka atsiktinį žodį





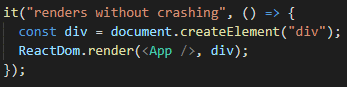
# Mock’ai

Rašant testus buvo privaloma imituoti aplinką, kurioje paprastai butų vykdomi testai. Tam įgyvendinti buvo naudojami mock’ai. Šiuo atveju testuojamas komponentas turėjo savo būseną, ir jos nustatymo funkciją. Buvo sukurta dirbtinė būsena ir jos nustatymo funkcija.

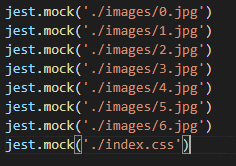


Kai kuriais atvejai reikėjo sukurti ir kitų dirbtinių objektų, pvz. mygtuko paspaudimo event’o objektą. Kadangi testuojama funkcija naudoja tik dalį to objekto, tai ir buvo sukurtas dalinis dirbtinis event’o objektas

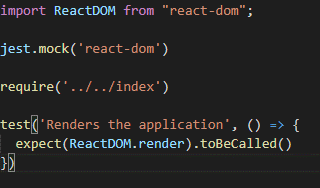


Taip pat buvo testuojamas ar komponentas teisingai atvaizduojamas ir sukuriamas aplinkoje. 

Bet komponentas naudojo paveiksliukus, kurie buvo importuojami. Jest biblioteka negalėjo atkurti tikrų komponento naudojamų paveiksliukų bei stiliaus failo, todėl buvo sukurtos jų dirbtiniai variantai, pasinaudojant mock funkcija.

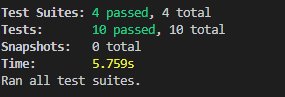
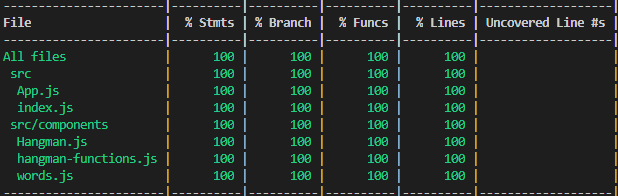


Testuojant pagrindinio komponento sukurimą, reikėjo sukurti dirbtinę DOM aplinką.



# Rezultatai

Visi testai įvykdomi tvarkingai, be jokių klaidų. Testais padengta 100% testuojamos sistemos.

# Išvados

Laboratorinio darbo metu buvo išsirinkta testuojama sistema, ji išanalizuota ir pilnai ištestuota. Išmokta naudotis jest testavimo įrankiu, bei kurti mock‘us. Testais padengta 100% kodo pagal jest testavimo įrankį.