LATVIJAS UNIVERSITĀTES
DATORIKAS FAKULTĀTE

“SCRABBLE” SPĒLES VIDES IMPLEMENTĀCIJA

KVALIFIKĀCIJAS DARBS

Autors:
Matīss Apinis (ma17058)
Darba vadītājs:
Dr. dat. Kārlis Freivalds

RĪGA 2019

Anotācija

Šis dokuments ir Latvijas Universitātes Datorikas fakultātes pirmā līmeņa profesionālās augstākās izglītības studiju programmas “Programmēšana un datortīklu administrēšana” kvalifikācijas darba ietvaros izstrādātās ““Scrabble” spēles vides implementācija” tehniskā dokumentācija, kas ietver programmatūras prasību specifikāciju un programmatūras projektējuma aprakstu.

““Scrabble” spēles vides implementācijas” mērķis ir nodrošināt lietotājam iespēju spēlēt tradicionālo galda spēli “Scrabble” datora vidē un vienam pašam, lai praktizētu savas “Scrabble” prasmes.

Atslēgvārdi: Scrabble, galda spēles

Abstract

This document contains the technical documentation for the system “Implementation of the board game “Scrabble””, which includes the software requirement specification and the software design description, as developed as part of University of Latvia Faculty of Computing’ Qualification paper.

The purpose of the “Implementation of the board game “Scrabble”” is to provide the user the opportunity to play the traditional board game “Scrabble” in a computer environment and with himself in order to practice their “Scrabble” skills.

Keywords: Scrabble, board games

Satura rādītājs

Ievads

Nolūks

Šī dokumenta nolūks ir iepazīstināt sistēmas izstrādātājus, pasūtītāju un sistēmas prospektīvos lietotājus ar sistēmas prasībām, funkcijām un to funkcionalitātes pielietojumiem.

Darbības sfēra

““Scrabble” spēles vides implementācija” ir sistēma, kas bezsaistes lietotnes formā piedāvā vienkāršu veidu, kā lietotājam spēlēt galda spēli “Scrabble” vienam pašam, lai praktizētu savas “Scrabble” prasmes.

Galda spēle “Scrabble” tiek pārdota 121 valstī un tiek piedāvāta 29 valodās. Līdz šim ir pārdoti vairāk nekā 150 miljoni spēles oriģinālo iepakojumu un ir izveidoti aptuveni 4 000 galda spēles “Scrabble” spēļu klubi pasaulē. Galda spēle ir populāra gan starp jauniešiem, gan vecāka gada gājuma cilvēkiem.

“Scrabble” ir vārdu galda spēle, kurā divi līdz četri spēlētāji iegūst punktus, liekot vārdus no savā rīcībā esošajiem burtiem uz spēles laukuma, kas sastāv no 15x15 lauciņiem. Uz katra lauciņa var novietot vienu kauliņu, izmantojot vairākus kauliņus horizontāli un vertikāli veidojot vārdus krustvārdu mīklas veidā. Vārdiem jābūt salasāmiem no kreisās uz labo pusi un tiem ir jābūt vārdiem no vārdnīcas (par vārdnīcas veidu vienojas spēlētāji spēles sākumā). Katram kauliņam ir piešķirts punktu skaits, kurus summējot veidojas vārda vērtība. Turklāt uz laukuma ir bonusa lauciņi, kas var piešķirt burtam vai vārdam lielāku vērtību ar divkārtīgi vai trīskārtīgi reizinošu raksturu. Kā arī izspēlējot visus 7 statīvā esošos kauliņus iegūtajam punktu skaitam tiek pieskaitīti 50 punkti bonusā. Spēles mērķis ir sakrāt visvairāk punktus spēles laikā. Spēle beidzas, kad spēles laukumā vairs nav brīvi lauciņi uz kuriem novietot kauliņus, lai izveidotu vārdu, vai arī kad viens no spēlētājiem ir izspēlējis visus sava statīva kauliņus, kamēr spēles maisā vairs nav neviena kauliņa.

Spēles norise sākas ar to, ka spēlētāji vienojas par vārdnīcas veidu, ko izmantos vārdu pārbaudei. Katrs spēlētājs nejaušā kārtā izvelk 7 kauliņus, kurus novieto kauliņu statīvā, spēlētāji savā starpā vienojas, kurš būs pirmais spēlētājs. Pirmais spēlētājs izvēlas divus vai vairāk kauliņus, kas veido vārdu, un izvieto tos uz spēles laukuma horizontāli vai vertikāli tā, lai viens kauliņš atrastos uz centrālā lauciņa. Gājienā iegūtie punkti tiek aprēķināti, saskaitot visu izmantoto burtu vērtību un pieskaitot punktus, kas iegūti, ja kauliņi novietoti uz balvu lauciņiem. Katra gājiena beigās spēlētājs izvelk tik jaunu kauliņu, cik viņš izspēlēja, lai kopējais

kauliņu skaits uz kauliņu statīva vienmēr būtu septiņi. Katram spēlētājam ir tiesības nomainīt kauliņus, kas atrodas uz spēlētāja statīva, izvelkot citus nejauši izvēlētus kauliņus, izlaist gājienu un pievienot vienu vai vairākus kauliņus jau izveidotam vārdam, lai veidotu jaunu vārdu. Visos gājienos izmantotie kauliņi jāizvieto vienā nepārtrauktā līnijā horizontāli vai vertikāli. Ja tie saskaras ar citiem kauliņiem blakus, tiem ir jāveido pilns vārds pēc krustvārdu mīklas principa. Spēlētājs iegūst punktus par visiem gājienā izveidotajiem vārdiem.

Šīs sistēmas konkurence sastāv no dažādām programmatūrām, kuras piedāvā iespēju spēlēt "Scrabble" (piemēram, *Quackle*), kuras iekļauj šīs sistēmas pamatfunkcionalitātes.

Saistība ar citiem dokumentiem

Dokumenta noformēšanā ievērotas standarta LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis" prasības. [1]

Pārskats

Dokuments ir sadalīts piecās galvenajās daļās:

1. Vispārējais apraksts – aprakstīta sistēma vispārīgā līmenī, iekļauj informāciju darījumprasībām, ierobežojumiem, pieņēmumiem un atkarībām.
2. Programmatūras prasību specifikācija – iekļauj funkcionālās un nefunkcionālās prasības.
3. Programmatūras projektējuma apraksts – iekļauj daļēju funkciju un lietotāja saskarņu projektējumu.
4. Testēšanas dokumentācija – iekļauj testpiemēru projektējumu un testēšanas rezultātus funkcionālajām un nefunkcionālajām prasībām.
5. Projekta organizācija – iekļauj informāciju par izmantoto programmatūras dzīves cikla modeli, darbietilpības novērtējumu, kvalitātes nodrošināšanu un konfigurāciju pārvaldību.

1. Vispārējais apraksts

1.1. Esošā stāvokļa apraksts

Šīs sistēmas konkurence sastāv no dažādām programmatūrām, kuras piedāvā iespēju spēlēt "Scrabble" (piemēram, *Quackle*), kuras iekļauj šīs sistēmas pamatfunkcionalitātes. [3]

1.2. Pasūtītājs

Sistēma aprakstīta pēc darba autora studiju programmas "Programmēšana un datortīklu administrēšana" kvalifikācijas darba ietvaros.

1.3. Produkta perspektīva

""Scrabble" spēles vides implementācija" ir sistēma, kas nākotnē varētu tikt attīstīta, lai piedāvātu vienkāršu vidi stimulētās mašīnmācīšanās algoritmu apmācībai spēlēt spēli "Scrabble", kas ir bijusi sākotnējā motivācija šajā darbā aprakstītās sistēmas izstrādei.

1.4. Darījumprasības

Programmatūrai ir jānodrošina šādas funkcionalitātes:

- Lietotājam ir iespēja spēlēt spēli "Scrabble" kā vienam spēlētājam, lai praktizētu savas "Scrabble" prasmes, atbilstoši spēles noteikumiem.
- Lietotājs spēles gājienus var izdarīt izmantojot datora peles klikšķu un kustību kombinācijas lietotājam intuitīvā veidā.
- Lietotājs var uz spēles laukuma izmēģināt dažādus pēc noteikumiem iespējamus potenciālos gājienus pirms galīga lēmuma pieņemšanas par attiecīgā gājiena izspēli.
- Lietotnē jebkurā brīdī lietotājam ir pieejams apskatei savs līdzšinējais spēlē iegūto punktu skaits.

1.5. Vispārējie ierobežojumi

Sistēmas darbība ir jāatbalsta programmēšanas valodas *Python* implementācijai *Python* 3.x, kā arī visām lietotākajām operētājsistēmām ar grafisku lietotāja saskarni (*Microsoft*

Windows 10, Apple Mac OS X, Ubuntu Linux 16.04. un/vai jaunākām to versijām). Sadarbība ar citām operētājsistēmu ar grafisku lietotāja saskarni versijām nav nepieciešama, bet ir vēlama.

1.6. Pieņēmumi un atkarības

Lai izmantotu visas sistēmas funkcionalitātes, lietotājam jābūt nodrošinātam ar datoru, operētājsistēmu ar grafisku lietotāja saskarni, ekrānu un datora peli un tastatūru, kā arī ar vispārīgu galda spēles "Scrabble" noteikumu izpratni.

2. Programmatūras prasību specifikācija

2.1. Funkcionālās prasības

2.1.1. Funkciju sadalījums pa moduļiem

Sistēmas funkcijas tiek sadalītas pa moduļiem attiecīgi maisa moduli, laukuma moduli, vārdnīcas moduli, saskarnes moduli, spēlētāja moduli un kauliņu moduli.

Maisa modulis sastāv no funkcijām, kuras reprezentē spēles maisu, kurā atrodas neizvilktie spēles kauliņi, un tā izmantošanu. Maisa modulis atbild par to, lai tiktu izveidots sākotnējais kauliņu sadalījums, kurā ir atbilstošas vērtības burtu un punktu pāriem un atbilstošs daudzums katra kauliņa kopiju maisā. Maisa modulis arī atbild par maisa izmantošanas funkciju spēles gaitā, ļaujot spēlētājam izvilkt nejaušu kauliņu no maisa un ļaujot spēlētājam atlikt savus kauliņus atpakaļ maisā, kā arī pārliecināties, vai maiss ir tukšs, kas ir arī viens no pietiekamajiem nosacījumiem, lai pabeigtu spēli.

Laukuma modulis sastāv no funkcijām, kuras reprezentē un vizualizē spēles laukumu un tā izmantošanu gājienu izdarīšanai. Laukuma modulis atbild par to, lai tiktu izveidots sākotnējais laukums, uz kura neatrodas neviens kauliņš un kurš sastāv no atbilstoša lauciņu izvietojuma ar attiecīgajiem burtu un vārdu izspēlē iegūto punktu reizinātājiem. Laukuma modulis arī atbild par laukuma vizualizēšanu grafiskajā lietotāja saskarnē, attēlojot laukuma stāvokli un attēlojot pagaidu gājienus pirms to izspēles. Kā arī laukuma modulis atbild par to, lai mēģinātie gājieni atbilstu spēles noteikumiem un izvēlētajai pieļaujamo vārdu vārdnīcai.

Vārdnīcas modulis sastāv no funkcijām, kuras reprezentē spēles vārdnīcu un tās izmantošanu izspēlēto vārdu pārbaudei. Vārdnīcas modulis atbild par to, lai tiktu izveidota spēles vārdnīca, kura satur atbilstošos spēlē izspēlējamos vārdus un kurā var pārbaudīt, vai izspēlētie vārdi arī atrodas.

Saskarnes modulis sastāv no funkcijām, kuras reprezentē to saskarnē izmantojamo elementu reprezentēšanu un vizualizēšanu, ar kuriem mijiedarbojoties spēlētājs pieņem galīgo lēmumu izspēlēt kauliņus gājienā vai apmainīt savus kauliņus gājienā pret jauniem. Saskarnes modulis atbild par to, lai uz ekrāna spēlētājam intuitīvā veidā tiktu attēlotas pogas šo darbību veikšanai.

Spēlētāja modulis sastāv no funkcijām, kuras reprezentē spēlētāju un izpilda spēlētāja caur grafiskājā lietotāja saskarnē veiktajām darbībām. Spēlētāja modulis atbild par to, lai tiktu izveidots sākotnējais spēlētāja modelis ar spēlētājam izmantojamo spēles laukumu, spēles maisu, spēlētāja statīvu, spēlētāja roku, ar kuru var paņemt un pārvietot kauliņu, kā arī spēlētāja rezultātu. Spēlētāja modulis arī atbild par to, lai spēlētājs varētu izmantot spēlētāja statīvā

esošos kauliņus – tos pacelt no statīva rokā, tos atlikt atpakaļ no pagaidu pozīcijas laukumā atpakaļ statīvā, apmainīt 2 kauliņus statīvā vietām –, un arī par spēlētāja statīva vizualizēšanu.

Kauliņu modulis sastāv no funkcijām, kuras reprezentē un vizualizē spēles kauliņus. Kauliņu modulis atbild par to kauliņu reprezentēšanu kā burta un tā izspēlē iegūstamo punktu skaita pāri, kā arī par attiecīgo kauliņu vizualizēšanu.

2.2. Nefunkcionālās prasības

2.2.1. Veiktspējas prasības

Sistēmai jānodrošina, ka 90% no veiktajiem pieprasījumiem tiek izpildīti ne ilgāk kā 2 sekunžu laikā.

2.2.2. Izmantojamība

Lietotāja saskarsmei un izskatam ir jābūt intuitīvam un minimālistiskam. Sistēmai ir jābūt pieejamai angļu valodā.

2.2.3. Produkta izstrādes prasības

Sistēmai ir jābūt izstrādātai *Python* 3.x programmēšanas valodā un ir jāizmanto *pygame* bibliotēka grafisku lietotāja saskarņu sistēmu programmēšanai. Funkcionalitātes izstrādē ir jāizmanto modernas un pierādītas metodes, kas pēc iespējas mazāk ietekmē sistēmas veiktspēju.

3. Programmatūras projektējuma apraksts

3.1. Maisa modulis

Maisa modulis ir paredzēts, lai pārvaldītu spēles maisu un kauliņus, kas tajā atrodas. Šis modulis satur funkcijas, kas nodrošina spēlēšanai pieejamos kauliņus, to izvilkšanu no maisa un atlikšanu atpakaļ maisā, kā arī to, lai maisā kauliņi atrastos jauktā secībā.

Spēles maisa inicializācija

Identifikators	MM.MI.01
Apraksts	
Jauna spēles maisa izveidošana ar sākotnējo kauliņu sadalījumu.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc lietotnes palaišanas.	
Ievade	
Nav.	
Apstrāde	
● Izveido spēles maisu ar sākotnējo kauliņu sadalījumu ar atbilstošo skaitu kauliņu kopiju katram kauliņam kā burta un tā izspēles vērtības punktu izteiksmē. ● Spēles maisā kauliņu secību sajauc nejaušā secībā.	
Izvade	
Nav.	

Kauliņa izvilkšana no spēles maisa

Identifikators	MM.KIM.02
Apraksts	
Viena jauna kauliņa izvilkšana no spēles maisa, ja maiss nav tukšs.	

Piekļuve
Funkcija tiek izpildīta lietotnē automātiski pēc veiksmīgas kauliņu izspēles izdarot spēli nebeidzošu gājienu vai pēc lietotāja pieprasījuma apmainīt spēlētāja statīvā esošos kauliņus.
Ievade
Nav.
Apstrāde
● Tiek pārbaudīts, vai spēles maiss ir tukšs vai nav tukšs. ● Ja spēles maiss ir tukšs, tad funkcija atgriež neko. ● Ja spēles maiss nav tukšs, tad funkcija izņem no spēles maisa pirmo tajā esošo kauliņu un šo kauliņu atgriež.
Izvade
Nav.

Spēles maisa tukšuma pārbaude

Identifikators	MM.MTP.03
Apraksts	
Pārbauda, vai spēles maiss ir tukšs vai nav.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Kauliņa izvilkšana no spēles maisa” (MM.KIM.02) izsaukšanas.	
Ievade	
Nav.	
Apstrāde	
● Tiek pārbaudīts, vai spēles maiss satur kauliņus vai nesatur. ● Ja spēles maiss satur nevienu kauliņu, tad funkcija atgriež to, ka maiss ir tukšs.	

● Ja spēles maiss satur vismaz vienu kauliņu, tad funkcija atgriež to, ka maiss nav tukšs.
Izvade
Nav.

Kauliņa atlikšana spēles maisā

Identifikators	MM.KAM.04
Apraksts	
Viena spēles kauliņa atlikšana atpakaļ spēles maisā.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas izsaukšanas.	
Ievade	
Atlikšanai paredzētais kauliņš.	
Apstrāde	
Atlikšanai paredzēto kauliņu pievieno spēles maisā esošajiem kauliņiem.	
Izvade	
Nav.	

Spēles maisa samaisīšana

Identifikators	MM.SM.05
Apraksts.	
Nejaušina secību, kādā kauliņi tiktu izvilkti no spēles maisa.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles maisa inicializācija” (KM.MI.01) izsaukšanas.	

.
Ievade
Nav.
Apstrāde
Spēles maisā esošo kauliņu secība tiek nejaušināti mainīta par citu kauliņu secību.
Izvade
Nav.

Kauliņa pievienošana spēles maisam

Identifikators	MM.KPM.06
Apraksts	
Spēles kauliņa attiecīgā skaita jaunizveidoto kopiju pievienošana spēles maisam.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles maisa inicializācija” (KM.MI.01) izsaukšanas.	
Ievade	
● Kauliņa burts. ● Kauliņa burta attiecīgais punktu skaits. ● Kauliņa kopiju skaits (*n*).	
Apstrāde	
Spēles maisam pievieno *n* kauliņa kopijas ar attiecīgo burta un tā izspēlē iegūstamo punktu skaitu.	
Izvade	
Nav.	

3.2. Laukuma modulis

Laukuma modulis ir paredzēts, lai pārvaldītu spēles laukumu un tā izmantošanu gājienu izdarīšanai, kā arī spēles laukuma vizualizēšanai. Šis modulis satur funkcijas, kas nodrošina spēlēšanai izmantojamo laukumu, mēģināto gājienu spēles noteikumu atbilstības pārbaudi, kā arī spēles laukuma stāvokla un pirmsizspēles pagaidu gājienu vizualizēšanu.

Spēles laukuma inicializācija

Identifikators	LM.LI.01
Apraksts	
Jauna spēles laukuma izveidošana ar tā lauciņiem, no kuriem atbilstošie ir arī atzīmēti kā izspēlēto burtu un vārdu punktu reizinātāju lauciņi.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc lietotnes palaišanas.	
Ievade	
Nav.	
Apstrāde	
• Tiek izveidota spēles laukums, kas sastāv no spēles laukuma izmēra atbilstoša skaita lauciņu. • Īpašie lauciņi, uz kuriem izspēlēto burtu vai vārdu vērtība iegūto punktu skaita ziņā tiek reizināta, tiek attiecīgi atzīmēti spēles laukumā.	
Izvade	
Nav.	

Pagaidu gājiena veikšana

Identifikators	LM.PGV.02
Apraksts	
Pagaidu gājiena kauliņu izlikšana uz spēles laukuma.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles laukuma kauliņa pacelšana” (LM.LKP.04) izsaukšanas.	
Ievade	
• Spēlētāja rokā esošais kauliņš. • Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.	
Apstrāde	
Tiek iegūts attiecīgais spēles laukuma lauciņš, uz kura saskarnē izdarīts peles klikšķis, izmantojot funkciju “Spēles lauciņa pozīcijas iegūšana” (LM.LPI.05). Ja ticis klišķināts uz kāda spēles lauciņa, tiek pārbaudīts, vai klikšķinātais spēles lauciņš atrodas vienā rindā vai kolonnā ar pārējiem lauciņiem, uz kuriem ir izlikti pagaidu gājiena kauliņi, un nav jau aizņemts ar citu kauliņu: ● Ja neatrodas vienā rindā vai kolonnā vai ir aizņemts, tad kauliņš rokā tiek atgriezts statīvā. ● Ja atrodas vienā rindā vai kolonnā un nav aizņemts, tad kauliņš no rokas tiek novietots lauciņā. Tiek pārbaudīts, vai lauciņā novietotais kauliņš ir tukšais kauliņš: ● Ja novietots tukšais kauliņš, tad tiek lietotājam pieprasīts ievadīt izspēlējamo burtu kauliņam.	
Izvade	

Nav.

Pagaidu kauliņa lineārās izspēlējamības pārbaude

Identifikators	LM.PKL.03
Apraksts	
Pagaidu gājiena ietvaros uz spēles laukuma izliktā kauliņa atrašanās vienā rindā vai vienā kolonnā ar pārējiem pagaidu kauliņiem pārbaude.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Pagaidu kauliņa izlikšana” (LM.PGV.02) izsaukšanas.	
Ievade	
Spēles laukuma lauciņa pozīcija.	
Apstrāde	
Ja kauliņš atrodas vienā rindā vai vienā kolonnā ar pārējiem pagaidus izliktajiem kauliņiem, tad funkcija atgriež, ka gājiens ir izspēlējams, citādi nav.	
Izvade	
Nav.	

Spēles laukuma kauliņa pacelšana

Identifikators	LM.LKP.04
Apraksts	
Kauliņa pacelšanas mēģinājums no dotā spēles laukuma lauciņa.	

Piekļuve
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.
Ievade
Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.
Apstrāde
Tiek iegūts attiecīgais spēles laukuma lauciņš, uz kura saskarnē izdarīts peles klikšķis, izmantojot funkciju “Spēles lauciņa pozīcijas iegūšana” (LM.LPI.05). Ja ticis klišķināts uz kāda spēles lauciņa, tiek pārbaudīts, vai klikšķinātajā spēles lauciņā atrodas kauliņš vai neatrodas: ● Ja atrodas, tad kauliņš tiek pacelts no lauciņa un nolikts spēlētāja statīvā. ● Ja neatrodas, funkcija atgriež neko.
Izvade
Nav.

Spēles lauciņa pozīcijas iegūšana

Identifikators	LM.LPI.05
Apraksts	
Ar datora peli klikšķinātā spēles laukuma lauciņa pozīcijas iegūšana.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Pagaidu gājiena veikšana” (LM.PGV.02) izsaukšanas un pēc funkcijas “Spēles laukuma kauliņa pacelšana” (LM.LKP.04) izsaukšanas.	

Ievade
Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.
Apstrāde
Tiek pārbaudīts, vai datora peles klikšķis ticis veikts uz spēles laukuma lauciņa: ● Ja klikšķināts uz lauciņa, tad funkcija atgriež lauciņa pozīciju. ● Ja klikšķināts ne uz lauciņa, tad funkcija atgriež neko.
Izvade
Nav.

Kauliņa novietošana uz spēles lauciņa

Identifikators	LM.KNL.06
Apraksts	
Kauliņa novietošana uz spēles laukuma lauciņa pēc lauciņa pozīcijas.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Izspēlētā vārda pārbaude pret spēles vārdnīcu” (VM.VPV.02).	
Ievade	
● Spēles laukuma lauciņa pozīcija. ● Novietojamais kauliņš.	
Apstrāde	
Tiek novietots kauliņš uz spēles laukuma lauciņa.	

Izvade
Nav.

Gājiena izspēle

Identifikators	LM.GI.07
Apraksts	
Pagaidu gājiena pareizības pārbaude un galīgā gājiena izspēlēšana.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.	
Ievade	
Patiesumvērtība, vai galīgais gājiens būs pirmais spēlē izdarītais gājiens vai nebūs pirmais.	
Apstrāde	

• Tiek pārbaudīts, vai vismaz pagaidu gājiens satur vismaz vienu pagaidu kauliņu.
• Tiek pārbaudīts, vai visi pagaidu gājiena kauliņi atrodas vienā rindā vai vienā kolonnā.
• Tiek pārbaudīts, vai visi pagaidu gājiena kauliņi atrodas nepārtrauktā rindā vai nepārtrauktā kolonnā bez neaizpildītiem lauciņiem starp pagaidu gājiena kauliņiem.
• Tiek pārbaudīts, vai tiek izveidots vismaz viens krustvārds.
• Tiek pārbaudīts, vai visi pagaidu gājiena izspēlē izveidotie vārdi atrodas spēles vārdnīcā.
• Tiek pārbaudīts, vai izspēlējamais gājiens būs pirmais spēles gājiens:
  • Ja būs pirmais gājiens, tad izspēlē pagaidu gājienu kā galīgo gājienu.
  • Ja nebūs pirmais gājiens, tad atgriež pagaidu gājiena kauliņus spēlētāja statīvā.	

Citādi pagaidu gājiena kauliņus atgriež spēlētāja statīvā, ja kāds no soļiem neizpildījās.
Izvade
Nav.

Gājienā izveidoto vārdu pārbaude

Identifikators	LM.VP.08
Apraksts	
Visu pagaidu gājiena izspēlē uz spēles laukuma jaunizveidoto vārdu atrašanās vārdnīcā pārbaude un veiksmīga gājiena gadījuma iegūto punktu skaita aprēķins.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Gājiena izspēle” (LM.GI.07) izsaukšanas.	
Ievade	
Patiesumvērtība, vai galīgais gājiens būs pirmais spēlē izdarītais gājiens vai nebūs pirmais.	
Apstrāde	
● Tiek pārbaudītas spēles laukuma visas rindas un visas kolonnas, kuras satur pagaidu gājiena kauliņus, lai pārliecinātos, vai tajās nav izveidoti potenciāli derīgi krustvārdi. ● Tiek pārbaudīts, vai visi potenciāli derīgie krustvārdi atrodas spēles vārdnīcā. ● Ja tiek izspēlēti visi 7 kauliņi no maksimālā spēlētāja statīvā esošo kauliņu skaita, tad gājienā iegūto punktu skaitam pieskaita 50 punktus. ● Tiek pārbaudīts, vai izveidotajos krustvārdos atrodas jau iepriekšējos gājienos izspēlēti kauliņi uz bonusu lauciņiem, kuri reizina burtu vai vārdu vērtību punktu skaita ziņā: ● Ja jau atrodas, tad attiecīgais bonusa reizinātājs netiek piešķirts.	

● Ja neatrodas, tad attiecīgais bonusa reizinātājs tiek iekļauts punktu skaita aprēķinā. ● Tiek aprēķināts visu jaunizveidoto krustvārdu kopējais iegūtais punktu skaits.
Izvade
Nav.

Pagaidu kauliņu noņemšana no spēles laukuma

Identifikators	LM.PKN.09
Apraksts	
Pagaidu gājiena kauliņu noņemšana no spēles laukuma	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Gājiena izspēle” (LM.GI.07) izsaukšanas.	
Ievade	
Nav.	
Apstrāde	
Pagaidu gājienā uz spēles laukuma izvietotie kauliņi tiek atgriezti kā kauliņu saraksts.	
Izvade	
Nav.	

Tukšā kauliņa burta izvēle

Identifikators	LM.TKI.10
Apraksts	
Spēlētāja pagaidu gājienā uz spēles laukuma izvietotā kauliņa burta izvēles aicinājums.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles laukuma kauliņa pacelšana” (LM.LKP.04) izsaukšanas.	
Ievade	
• Tukšais kauliņš. • Lietotāja nospiestais tastatūras taustiņš. • Spēles grafiskās lietotāja saskarnes virsma.	
Apstrāde	
Tiek vizualizēts tukšā kauliņa burta izvēles aicinājums. Tiek sagaidīts lietotāja nospiestais tastatūras taustiņš: • Ja tas ir kāds no burtiem, kādi tiek izmantoti spēlē, tad tukšā kauliņa burts kļūst par no tastatūras ievadīto burtu ar punktu skaita vērtību “0”.	
Izvade	
Uz lietotnes grafiskās lietotāja saskarnes virsmas tiek vizualizēts tukšā kauliņa burta izvēles aicinājums. Pēc derīgas tukšā kauliņa burta izvēles aicinājums pazūd, un uz tukšā kauliņa ir uzzīmēts burts un tā punktu skaita vērtība “0”.	

Spēles laukuma vizualizācija

Identifikators	LM.LV.11

Apraksts
Spēles laukuma stāvokļa ar visiem uz tā esošajiem spēles kauliņiem vizualizācija.
Piekļuve
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.
Ievade
Spēles grafiskās lietotāja saskarnes virsma.
Apstrāde
Tiek vizualizēts spēles laukuma stāvoklis ar spēles laukuma pamatnes attēlojumu, uz kura ir izvietoti spēles lauciņi attiecīgi ar vai bez to bonusa reizinātāju attēlojumu un lauciņā esošo kauliņu, ja tāds ir novietots.
Izvade
Nav.

3.3. Vārdnīcas modulis

Vārdnīcas modulis ir paredzēts, lai pārvaldītu spēles vārdnīcu. Šis modulis satur funkcijas, kas nodrošina spēles vārdnīcu ar pieņemami izspēlējamajiem vārdiem un izspēlē radīto vārdu apstiprināšanu pēc šīs vārdnīcas standarta.

Spēles vārdnīcas inicializācija

Identifikators	VM.VI.01
Apraksts	
Spēles vārdnīcas izveide ātrai izspēlēto vārdu pārbaudei.	
Piekļuve	

Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles laukuma inicializācija” (LM.LI.01) izsaukšanas.
Ievade
Vārdnīcas teksta datnes nosaukums.
Apstrāde
Tiek apstrādāta teksta datnes pa rindiņai, kur katrā rindiņā atrodas viens vārds, kurš tiek ievietots spēles vārdnīcā.
Izvade
Nav.

Izspēlētā vārda pārbaude pret spēles vārdnīcu

Identifikators	VM.VPV.02
Apraksts	
Izspēlētā vārda atrašanās spēles vārdnīcā pārbaude.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Gājiena izspēle” (LM.GI.07) izsaukšanas.	
Ievade	
Pārbaudāmais izspēlētais vārds.	
Apstrāde	
Tiek pārbaudīts, vai izspēlētais vārds atrodas spēles vārdnīcā.	
Izvade	
Nav.	

3.4. Saskarnes modulis

Saskarnes modulis ir paredzēts, lai pārvaldītu saskarnes elementus, kurus klikšķinot spēlētājs pieņem galīgos lēmumus izspēlēt gājiena kauliņus vai sava statīva kauliņus apmainīt pret jauniem kauliņiem. Šis modulis satur funkcijas, kuras atbild par to, lai tiktu vizuāli attēlotas izmantojamas pogas šo darbību veikšanai.

Saskarnes pogu inicializācija

Identifikators	SM.PI.01
Apraksts	
Saskarnes pogu kauliņu izspēlei vai apmaiņai inicializācija.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.	
Ievade	
Nav.	
Apstrāde	
Tiek izveidotas saskarnes pogas un attēlotas klikšķināma taisnstūra formā ar pogas darbību aprakstošu tekstu tajā.	
Izvade	
Nav.	

Saskarnes pogas nosaukuma iegūšana

Identifikators	SM.PNI.02
Apraksts	
Saskarnes pogas uzspiešanas gadījumā notiek tās nosaukuma atgriešana.	
Piekļuve	

Lietotājs. Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.
Ievade
Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.
Apstrāde
Tiek iterēts cauri visām pogām, lai noskaidrotu, vai attiecīgā poga tika klikšķināta: • Ja poga tika klikšķināta, tad tiek atgriezts pogas nosaukums.
Izvade
Nav.

3.5. Spēlētāja modulis

Spēlētāja modulis ir paredzēts, lai pārvaldītu spēlētāja reprezentāciju un izpildītu caur grafisko lietotāja saskarni spēlētāja pieprasītās darbības. Šis modulis satur funkcijas, kas nodrošina spēlētāja caur grafisko lietotāja saskarni pieprasīto darbību izpildi, tajā skaitā spēles laukumu, spēles maisu, spēlētāja statīvā esošajiem kauliņiem un spēlētāja roku kauliņu paņemšanai un pārvietošanai, kā arī spēlētāja rezultāta uzglabāšanu.

Spēlētāja inicializācija

Identifikators	PM.SI.01
Apraksts	
Jauna spēles spēlētāja ar tam izmantojamo spēles laukuma, spēles maisa, spēlētāja statīva, spēlētāja rokas un spēlētāja rezultāta inicializācija.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.	
Ievade	

• Spēles laukums. • Spēles maiss.
Apstrāde
Tiek inicializētas aprakstīto objektu sākotnējās vērtības (spēlētāja rezultāts “0”, tukša spēlētāja roka, spēlētāja statīvā 7 kauliņi no spēlētāja maisa).
Izvade
Nav.

Spēlētāja statīva kauliņa pacelšana

Identifikators	PM.SKP.02
Apraksts	
Spēlētāja statīvā esošā kauliņa pacelšana spēlētāja rokā ar kauliņu apmainīšanu vietām jau aizņemtas rokas gadījumā.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.	
Ievade	
Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.	
Apstrāde	
Tiek pārbaudīts, vai spēlētāja rokā jau atrodas statīva kauliņš: • Ja rokā jau atrodas kauliņš, tad to apmaina ar klikšķināto statīva kauliņu. Ja rokā neatrodas kauliņš, tad to paceļ rokā. Tiek iegūts peles klikšķim atbilstošais kauliņš no spēlētāja statīva: • Ja klikšķa koordinātās neatrodas statīvā kauliņš, tad atgriež neko un esošo kauliņu noliek no rokas atpakaļ statīvā. • Ja klikšķa koordinātās atrodas statīvā kauliņš, tad mēģina pacelt kauliņu rokā.	

Izvade
Nav.

Spēlētāja statīva pozīcijas iegūšana

Identifikators	PM.SPI.03
Apraksts	
Spēlētāja statīvā esošā kauliņa pozīcijas statīvā iegūšana.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēlētāja statīva kauliņa pacelšana” (PM.SKP.02) izsaukšanas.	
Ievade	
Lietotāja saskarnē izdarītā datora peles klikšķa 2D koordinātas.	
Apstrāde	
Tiek pārbaudīts, vai datora peles klikšķis atrodas spēlētāja statīva kauliņa attēlojumā. • Ja klišķis ir izdarīts uz kauliņa statīvā, tad atgriež kauliņa pozīciju statīvā. • Ja klišķis nav izdarīts uz kauliņa statīvā, tad atgriež neko.	
Izvade	
Nav.	

Spēlētāja statīva vizualizācija

Identifikators	PM.SV.04
Apraksts	
Spēlētāja statīva un tajā esošo kauliņu vizualizācija.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc galvenās funkcijas izsaukšanas.	
Ievade	
Spēles grafiskās lietotāja saskarnes virsma.	
Apstrāde	
Tiek vizualizēta spēlētāja statīva pamatne un katrs statīvā esošais kauliņš.	
Izvade	
Nav.	

Spēles pabeigšana

Identifikators	PM.SP.05
Apraksts	
Spēles pabeigšana ar neizspēlēto kauliņu punktu vērtības atņemšanu no spēlētāja rezultāta.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas galvenās funkcijas izsaukšanas.	
Ievade	
• Spēlētāja rezultāts. • Spēlētāja statīvs ar tajā esošajiem kauliņiem.	
Apstrāde	

Tiek aprēķināts par neizspēlēto kauliņu iegūtais soda punktu skaits, kas tiek atņemts no spēlētāja galarezultāta.
Izvade
Nav.

Spēlētāja statīva kauliņu vērtības aprēķins

Identifikators	PM.SKV.06
Apraksts	
Spēlētāja statīvā esošā kauliņu kopējās vērtības punktu skaita aprēķins, lai spēles beigās atņemtu šos punktus par visiem neizspēlētajiem kauliņiem.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles pabeigšana” (PM.SP.04) izsaukšanas.	
Ievade	
Nav.	
Apstrāde	
Tiek saskaitīts un galā atgriezts punktu skaits pāri statīvā atlikušajiem kauliņiem.	
Izvade	
Nav.	

Spēlētāja rezultāta palielināšana

Identifikators	PM.RP.07
Apraksts	
Spēlētāja rezultāta palielināšana piešķirot gājiena izspēlē iegūtos punktus.	

Piekļuve
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles pabeigšana” (PM.SP.04) izsaukšanas.
Ievade
Spēlētāja pēdējā gājienā iegūto punktu skaits.
Apstrāde
Spēlētāja kopējam punktu skaitam pieskaita spēlētāja pēdējā gājienā iegūto punktu skaitu.
Izvade
Nav.

3.6. Kauliņu modulis

Kauliņu modulis ir paredzēts, lai pārvaldītu spēles kauliņu reprezentāciju un izpildītu spēles kauliņu vizualizāciju. Šis modulis satur funkcijas, kas nodrošina spēles kauliņa izveidi, spēles kauliņa attēlošanu uz grafiskās lietotāja saskarnes, kā arī pārbaudi, vai spēles kauliņš ir tukšs.

Spēles kauliņa inicializācija

Identifikators	KM.KI.01
Apraksts	
Jauna spēles kauliņa ar tā burta un attiecīgi burtam iegūstamo punktu skaita inicializācija.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles maisa inicializācija” (MM.MI.01) izsaukšanas.	
Ievade	
• Spēles kauliņa burts. • Spēles kauliņa izspēlē iegūstamo punktu skaits.	

Apstrāde
Tiek izveidots jauns spēles kauliņš ar burtu un attiecīgi iegūstamo punktu skaitu par kauliņa izspēli.
Izvade
Nav.

Tukšā kauliņa pārbaude

Identifikators	KM.TKP.02
Apraksts	
Atgriež patiesumvērtību tam, vai dotais spēles kauliņš ir tukšais kauliņš.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Gājiena izspēle” (LM.GI.07) izsaukšanas.	
Ievade	
Nav.	
Apstrāde	
Tiek atgriezts, vai spēles kauliņa burts ir tukšs un vai punktu skaits ir “0”.	
Izvade	
Nav.	

Spēles kauliņa vizualizācija

Identifikators	KM.KV.03
Apraksts	
Spēles kauliņa vizualizācijas grafiskajā lietotāja saskarnē.	

Piekļuve
Funkcija tiek izpildīta lietotnē automātiski pēc funkcijas “Spēles laukuma vizualizācija” (LM.LV.11) izsaukšanas un pēc “Spēlētāja statīva vizualizācija” (PM.SV.03) izsaukšanas.
Ievade
• 2D koordinātas punktam, kurā sākt attēlot spēles kauliņu. • Spēles grafiskās lietotāja saskarnes virsma.
Apstrāde
Tiek pārbaudīts, vai spēles kauliņš ir tukšais kauliņš: • Ja kauliņš ir tukšs, tad netiek attēlots burts un punktu skaits. • Ja kauliņš nav tukšs, tad tiek attēlots burts un attiecīgais punktu skaits.
Izvade
Nav.

3.7. Galvenā funkcija

Galvenā funkcija

Identifikators	GF.00
Apraksts	
Lietotnes galvenā funkcija, ar kuru sākas un beidzas programmatūras koda izpilde.	
Piekļuve	
Funkcija tiek izpildīta lietotnē automātiski pēc programmas izpildes.	
Ievade	
Nav.	
Apstrāde	
Tiek inicializētas lietotnes darbināšanai izmantotās *Python* bibliotēkas un pārējo moduļu lokālās *Python* datnes.	

Tiek inicializētas konstantes lietotnes grafiskās lietotāja saskarnes loga izmēriem, loga krāsām, saskarnes vizuālo elementu virsmas un krāsas.

Tiek izsauktas funkcijas spēles saskarnes, spēles laukuma, spēles maisa, spēles spēlētāja inicializācijai un vizualizējamo objektu attēlošanai grafiskās lietotnes saskarnē uz ekrāna.

Tiek darbināta *pygame* vidē implementētā šī projekta sistēma – "Scrabble" spēles versija –, kur tiek reģistrēti lietotnē peles kustību vai klikšķu notikumi viens pēc otra, kur pēc katra:

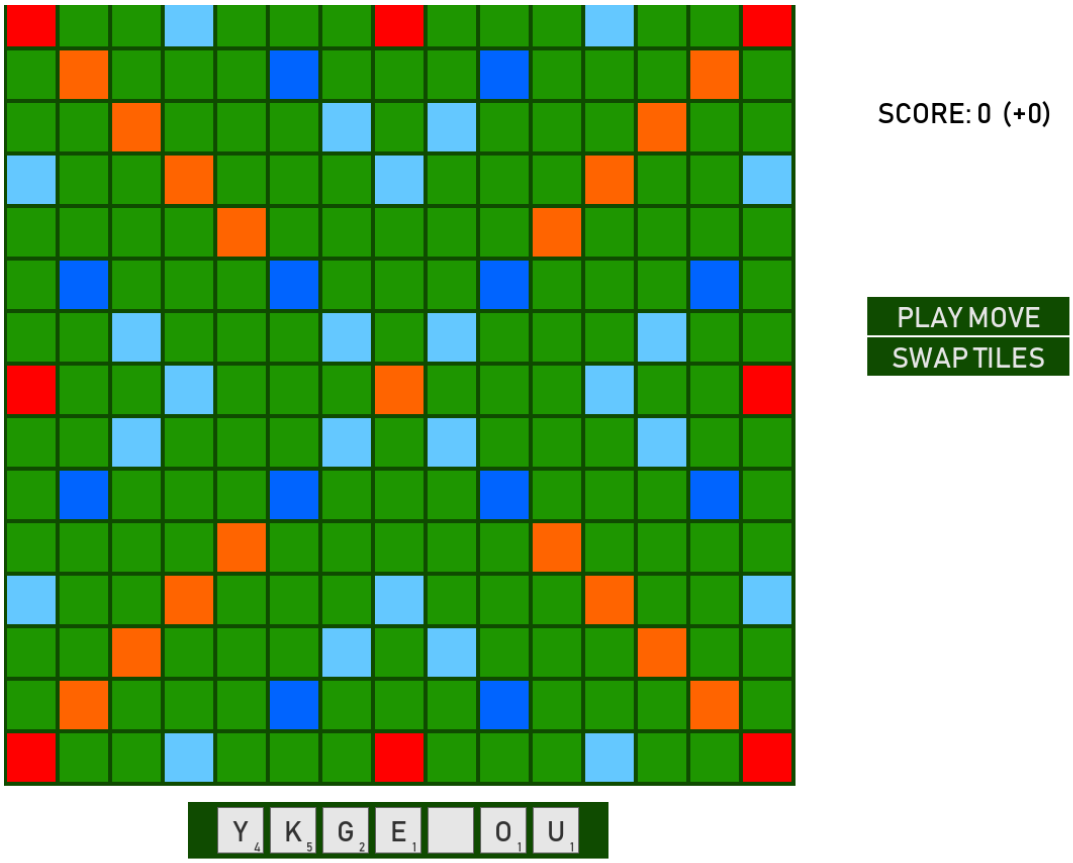
- Peles kustību gadījumā, ja tās veicot ir nonācis uz saskarnes elementiem, ar kuriem lietotājs var mijiedarboties, tad elements tiek vizuāli izcelts.
- Peles klikšķu gadījumā, ja tas ir veikts uz saskarnes elementiem, ar kuriem lietotājs var mijiedarboties, tad elementam atbilstošo darbību funkcijas tiek izsauktas.

Peles kustību un klikšķu rezultātā lietotnes funkcijas tiek izmantotas līdz tiek pieprasīta lietotnes procesa pārtraukšana vai arī tās tiek pastāvīgi lietotas nenoteiktu laiku.

Izvade
Nav.

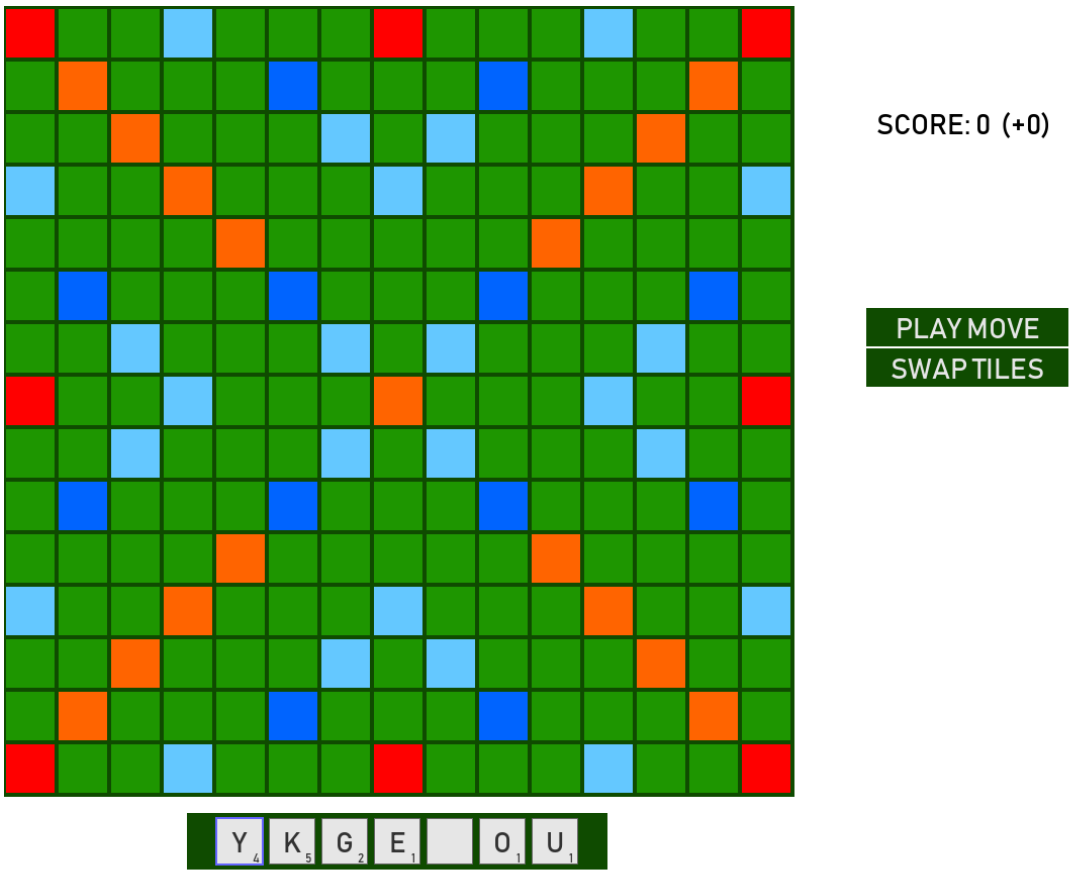
4. Testēšanas dokumentācija

4.1. Testpiemēru projektējums



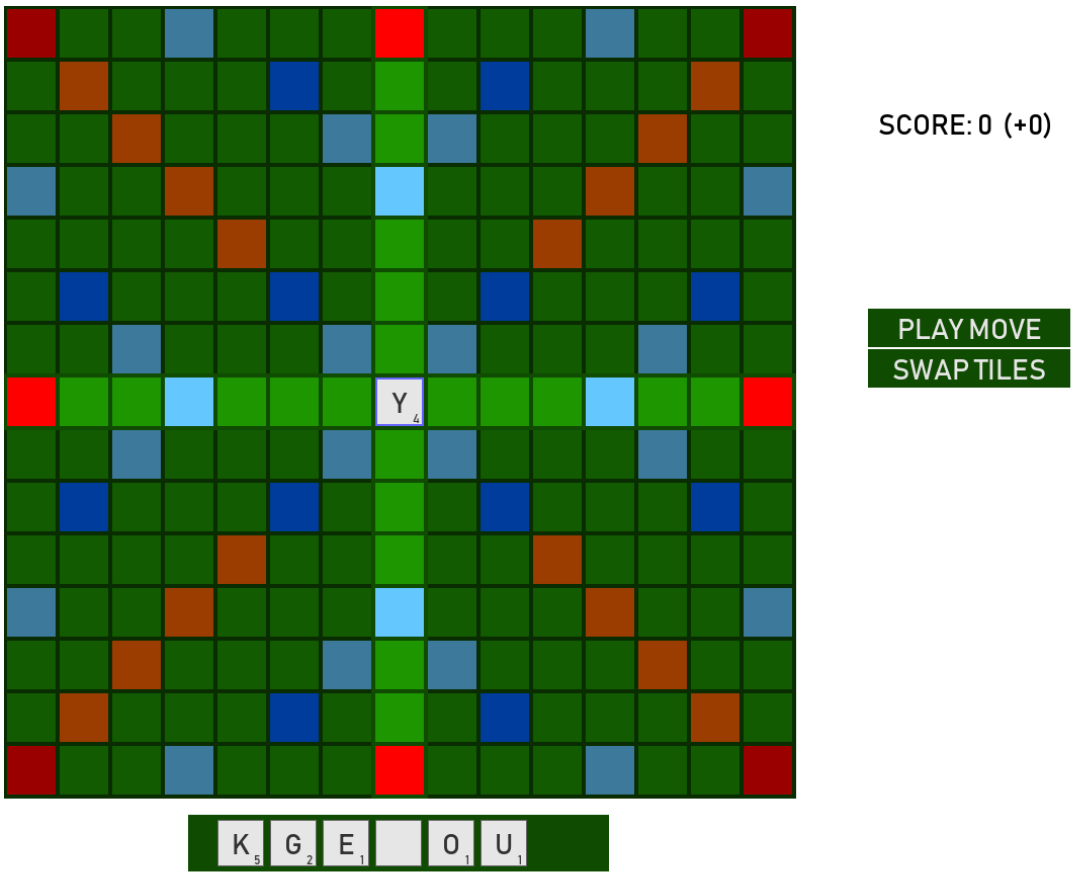
4.1. att. – **Skats pēc lietotnes atvēršanas.**

Ticis inicializēts un vizualizēts spēles laukums ar iekrāsotiem spēles lauciņiem un zem tā spēles statīvu, kas satur nejauši no spēles maisa izvilktus 7 kauliņus, kur šie elementi ir attēloti līdzīgās krāsās oriģinālajai galda spēlei "Scrabble". Labajā pusē ir redzamas pogas galīgās kauliņu izspēles izdarīšanai un kauliņu apmainīšanai pret jauniem, virs kurām atrodas spēlētāja rezultāts kā kopējo punktu skaita un arī pēdējā rezultatīvajā gājienā iegūto punktu skaits.



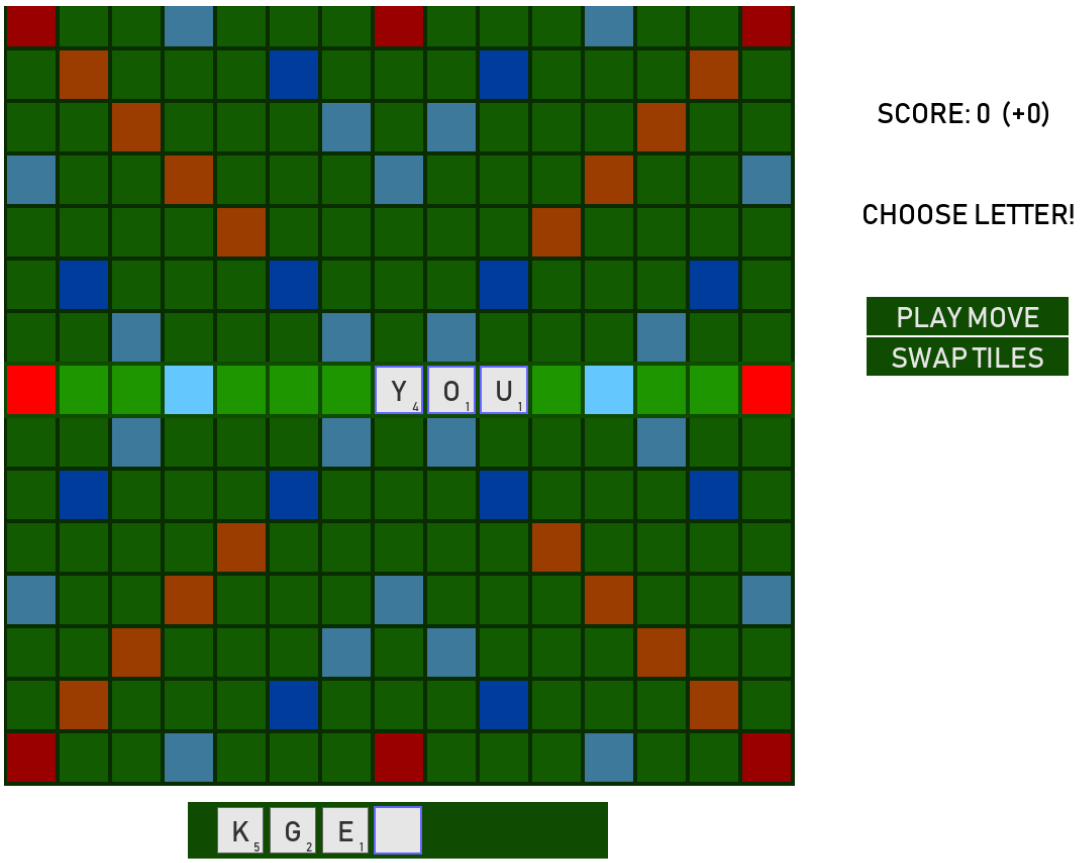
4.2. att. – **Skats pēc kauliņa paņemšanas spēlētāja rokā no spēlētāja statīva.**

Ticis uzklikšķināts uz kauliņa ar burtu “Y”, tātad tas ir ticis paņemts no spēlētāja statīva un ir nonācis spēlētāja rokā, kas tiek apzīmēts ar zilās krāsa ierāmējumu ap šo kauliņu.



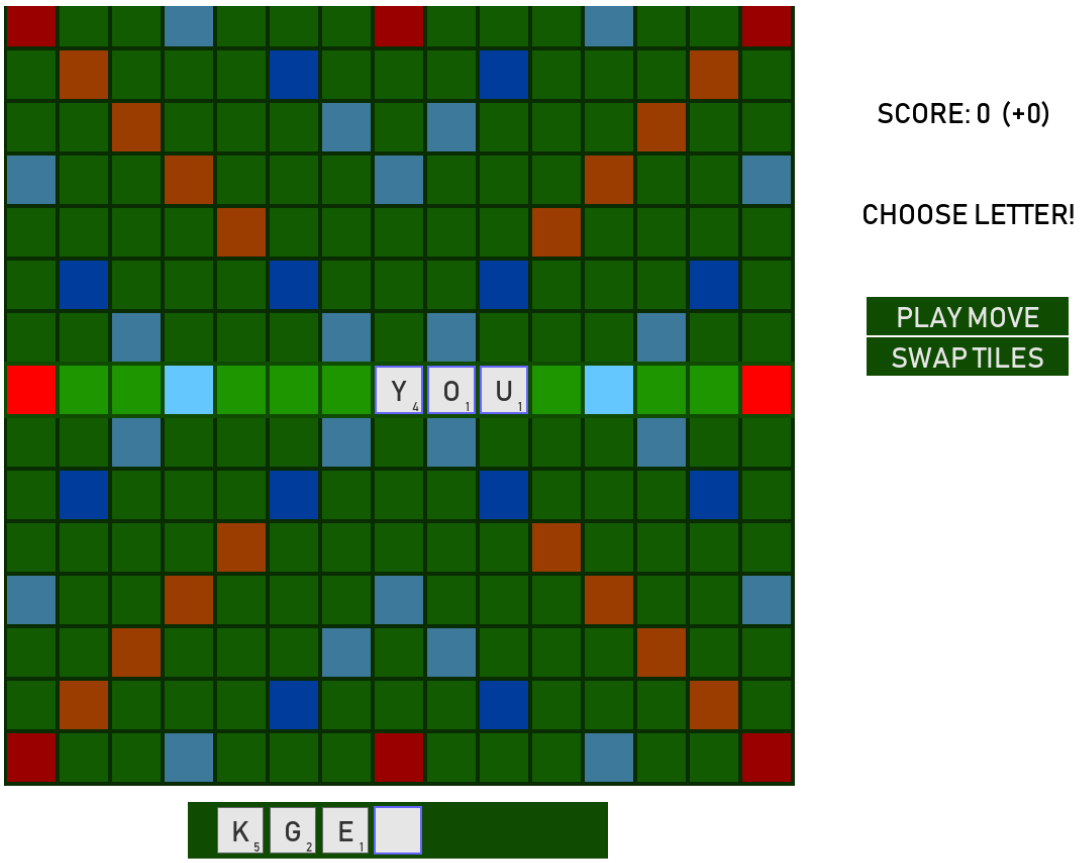
4.3. att. – **Skats pēc pirmā kauliņa izlikšanas uz spēles laukuma pirmajā pagaidu gājienā.**

Pēc kauliņa paņemšanas no spēlētāja statīva un nonākšanas spēlētāja rokā, lietotājs ir uzspiedis uz centrālā spēles laukuma lauciņa, tādējādi noliekot kauliņu uz spēles laukuma kā pagaidu gājienu. Pēc šī kauliņa nolikšanas uz spēles laukuma vizuāli tiek attēlota rinda un kolonna, kuras lauciņos ir iespējams izvietot turpmākos kauliņus pagaidu gājienam, kā arī spēlētāja statīvā vairs nav sastopams izliktais kauliņš.



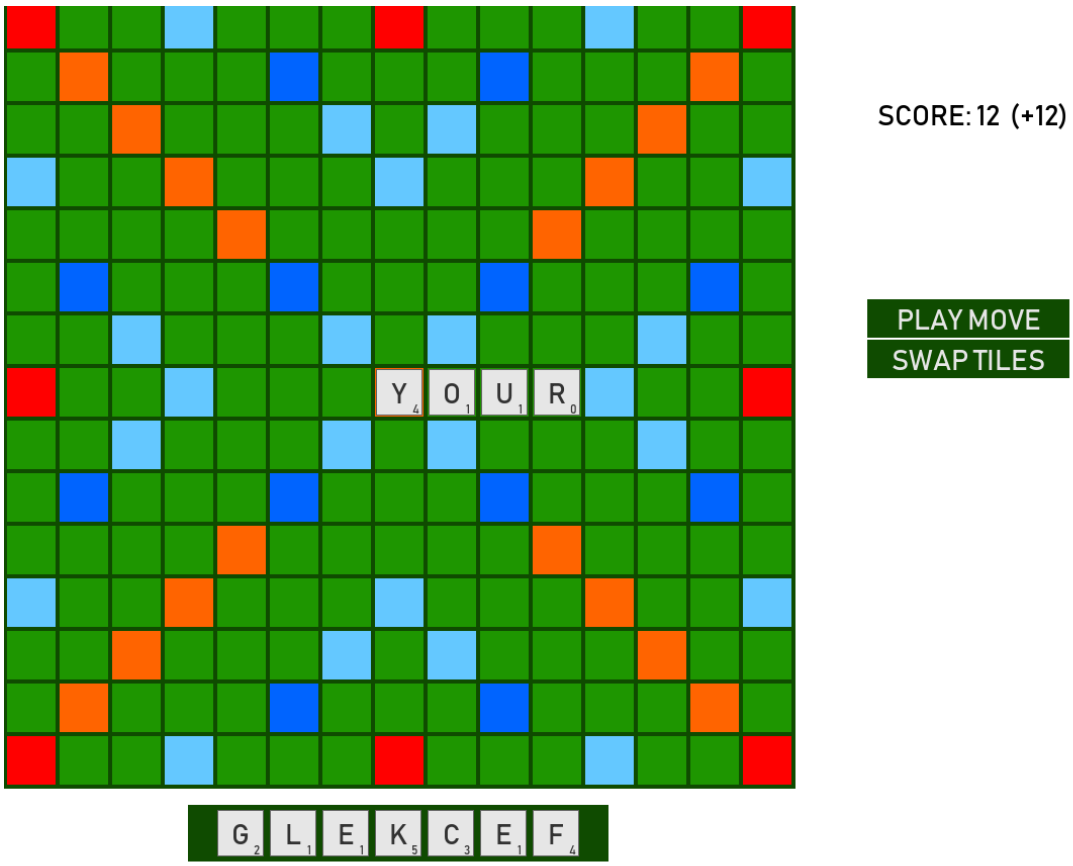
4.4. att. – **Skats pēc vairāku kauliņu izlikšanas uz spēles laukuma pirmajā pagaidu gājienā.**

Pēc vairāku kauliņu paņemšanas no spēlētāja statīva un izliekšanas uz spēles laukuma kā jau cita pagaidu gājiena, spēles laukums vizuāli attēlo to rindu, kuras lauciņos joprojām ir iespējams izvietot turpmākos kauliņus pagaidu gājienam, kā arī spēlētāja statīvā vairs nav sastopami izliktie kauliņi.



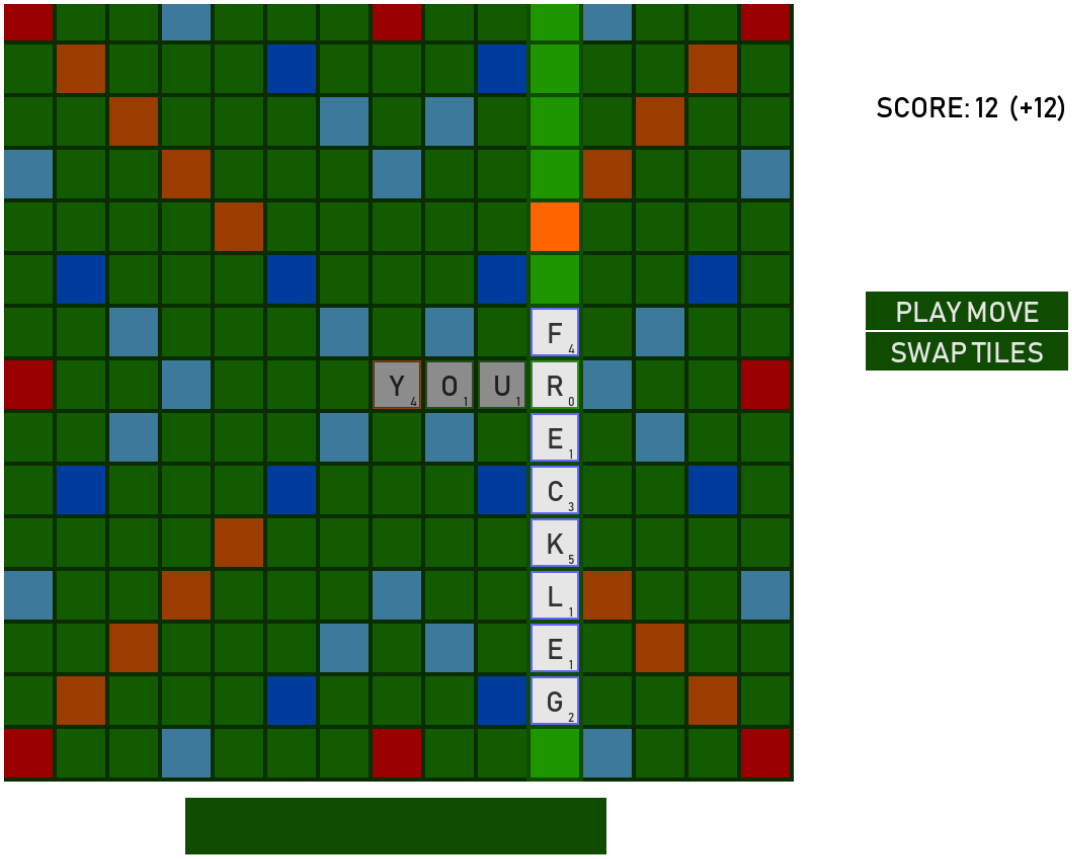
4.5. att. – **Skats pēc mēģinājuma izlikt tukšo kauliņu pirmajā pagaidu gājienā.**

Ticis uzklikšķināts uz tukšā kauliņa (bez burta un tā punktu skaita skaitļa), kurš ir iezīmēts ar zilās krāsas ierāmējumu, un ticis uzklikšķināts uz nākamā lauciņa pa labi no spēles laukumā izvietotā kauliņa ar burtu "U", tātad tas saskarnes labajā pusē tiek parādīts aicinājums izvēlēties burtu, kuru piedēvēt izliktajam tukšajam kauliņam.



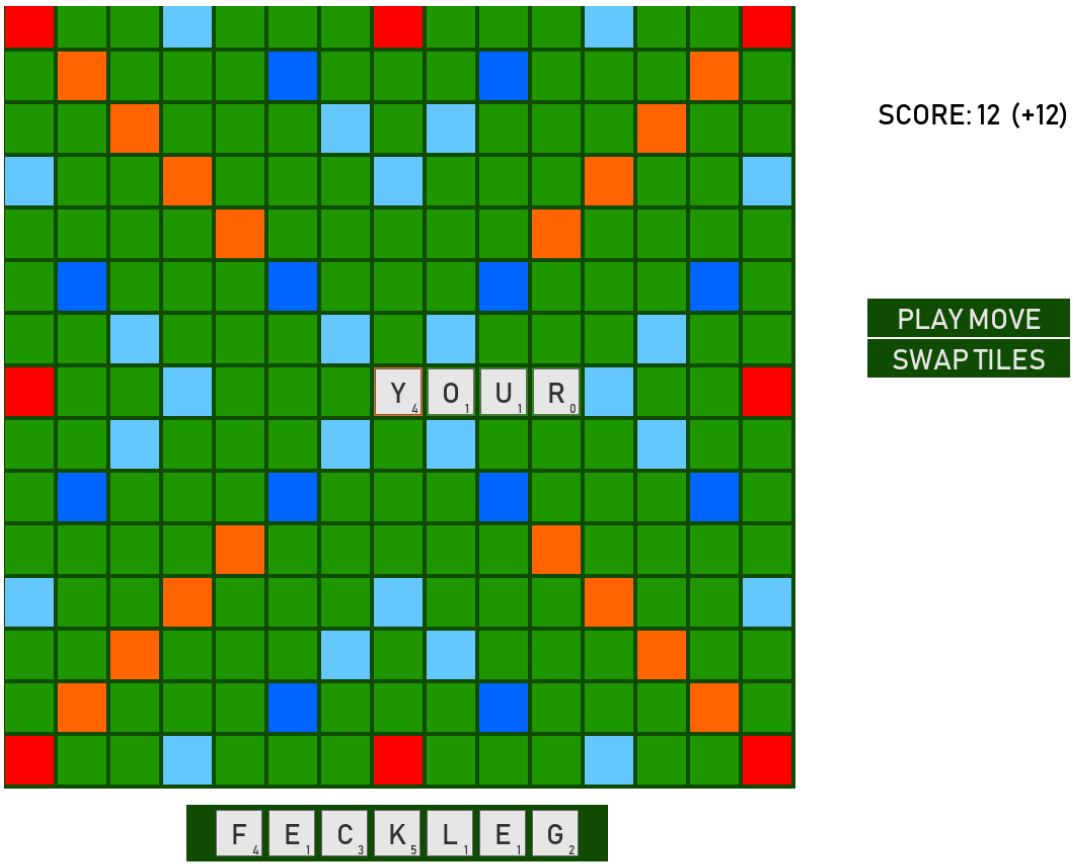
4.6. att. – **Skats pēc tukšā kauliņa burta izvēlēšanās un pagaidu gājiena kauliņu veiksmīgas galīgās izspēles.**

Pēc aicinājuma izvēlēties tukšajam kauliņam burtu uz tastatūras lietotājs ir nospiedis pogu "R", kā rezultātā uz spēles laukuma ir novietots pagaidu gājienam papildu kauliņš ar burtu "R" un tā punktu skaita vērtību "0". Pēc tam pagaidu gājiens ir mēģināts tikt izspēlēts, klikšķinot uz pogas ar tekstu "PLAY MOVE". Tā kā izveidotais vārds "YOUR" atrodas spēles vārdnīcā (šīs spēles partijas gadījumā pēc "Scrabble" turnīros bieži lietotās SOWPODS angļu valodas vārdnīcas) un kāds no tā kauliņiem atrodas centrālajā lauciņā, tad gājiena galīgās izspēles mēģinājums ir veiksmīgs – uz spēles laukuma paliek izspēlētie kauliņi (nav vairs redzami pagaidu gājiena ierobežojumi, ir redzams kauliņiem, kas atrodas uz bonusa lauciņiem, ierāmējums bonusa lauciņa krāsā) un spēlētāja rezultāts ir mainījies atbilstoši rezultatīvajā gājienā ar vārdu "YOUR" iegūtajam punktu skaitam: (4 + 1 + 1 + 0) * 2 = 6 * 2 = 12, kur reizinātājs ir pateicoties centrālajam bonusa laukumam, kas dubulto par vārdu iegūto punktu skaitu. Pēc izdarītā gājiena ir automātiski arī izvilkti 4 jauni kauliņi no spēles maisa, lai spēlētāja statīvs būtu pilns (jo maisā kauliņu vēl pietiek).



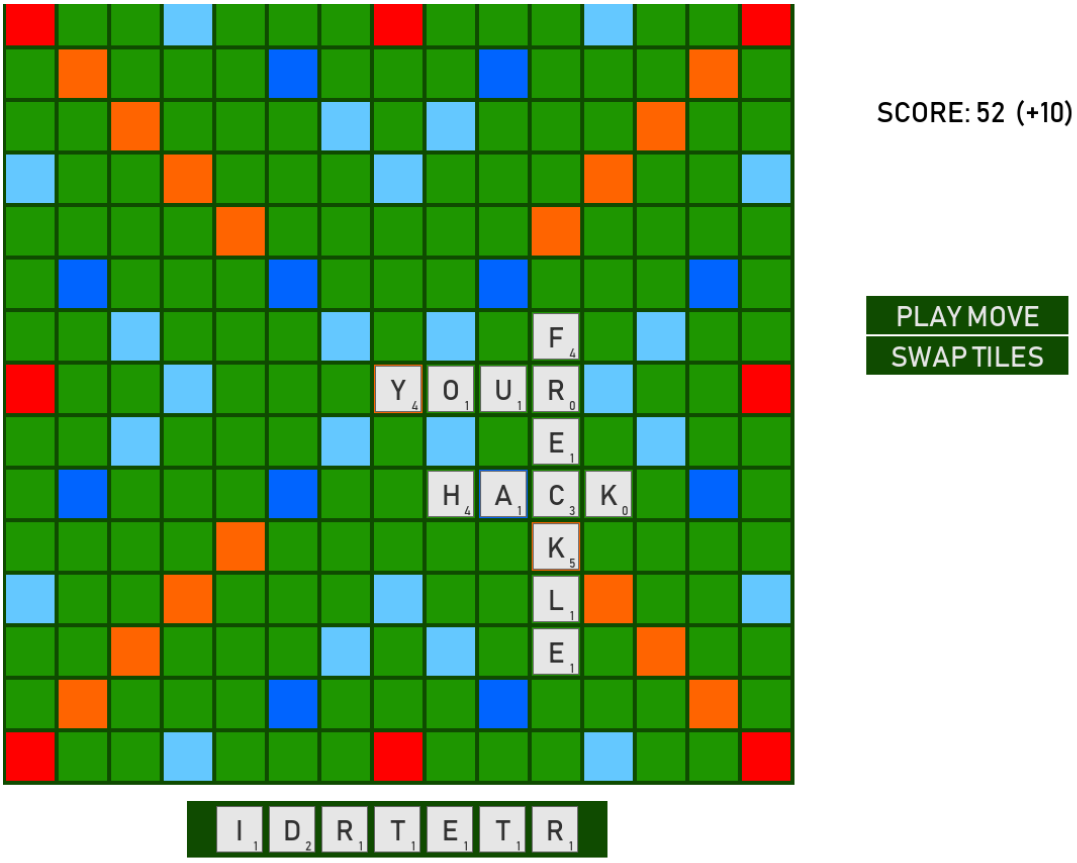
4.7. att. – **Skats ar pagaidu gājienu spēles laukumā pirms mēģinājuma neveiksmīgi to galīgi izspēlēt.**

Uz laukuma vertikāli kolonnā pēc krustvārdu principa ir izlikti visi kauliņi, lai veidotu jauno krustvārdu “FRECKLESG”. Taču šis vārds neatrodas izraudzītajā spēles vārdnīcā.

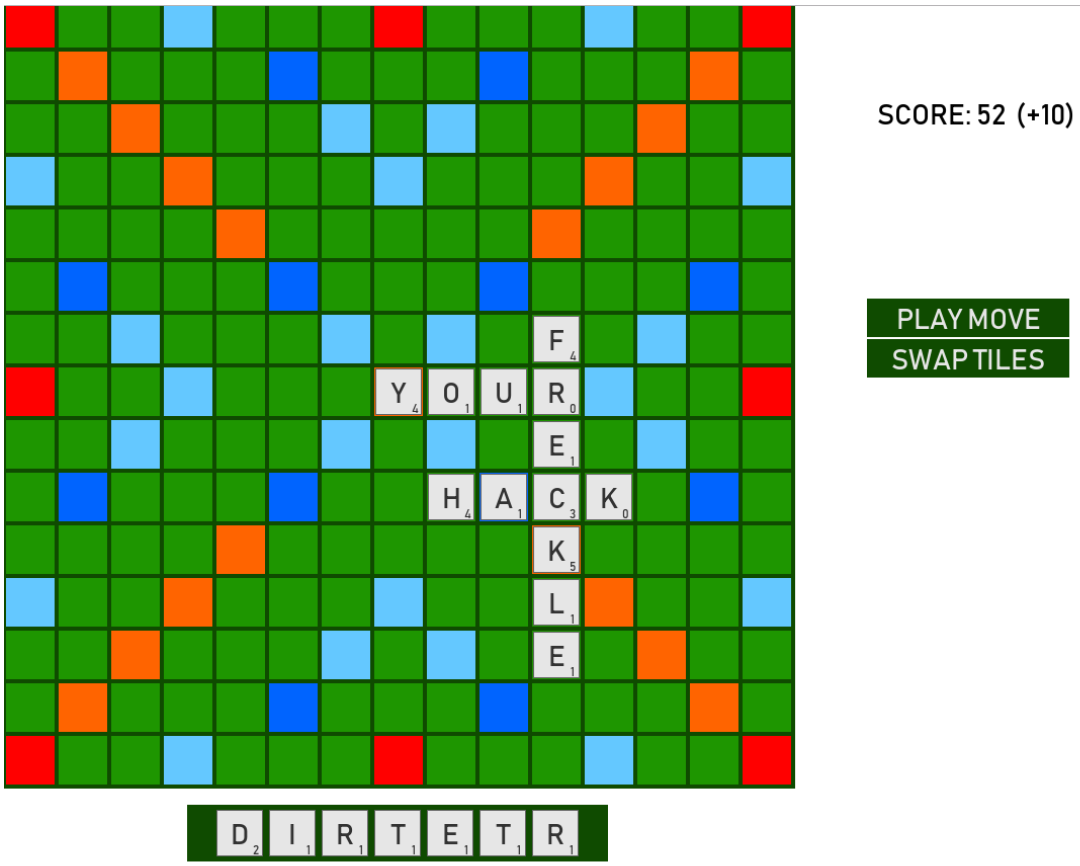


4.8. att. – **Skats pēc neveiksmīga pagaidu gājiena izspēles mēģinājuma ar spēlētāja statīvā atgrieztiem tā kauliņiem.**

Tā kā vārds “FRECKLESG” neatrodas izraudzītajā vārdnīcā, tad tā izlikšanā izmantotie kauliņi tiek atgriezti spēlētāja statīvā un punkti par šo nerezultatīvo gājienu netika piešķirti.

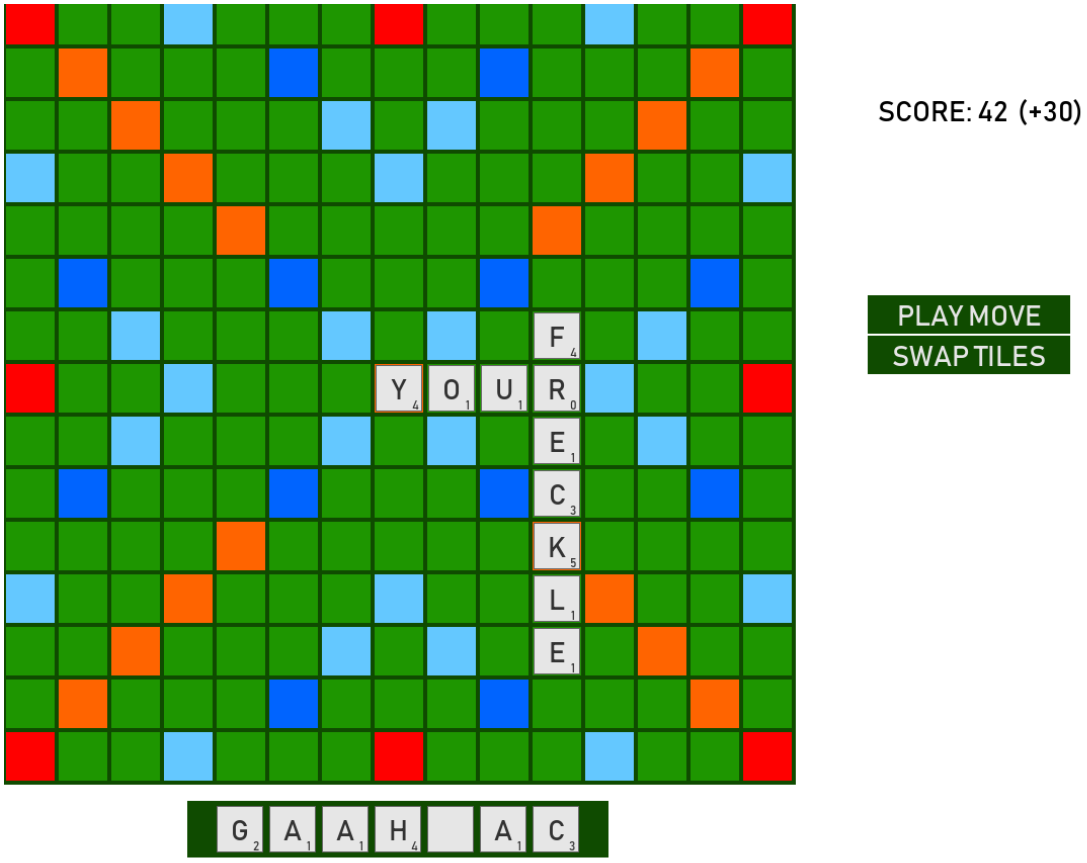


4.9. att. – **Skats pirms divu kauliņu spēlētāja statīvā apmainīšanas vietām.**



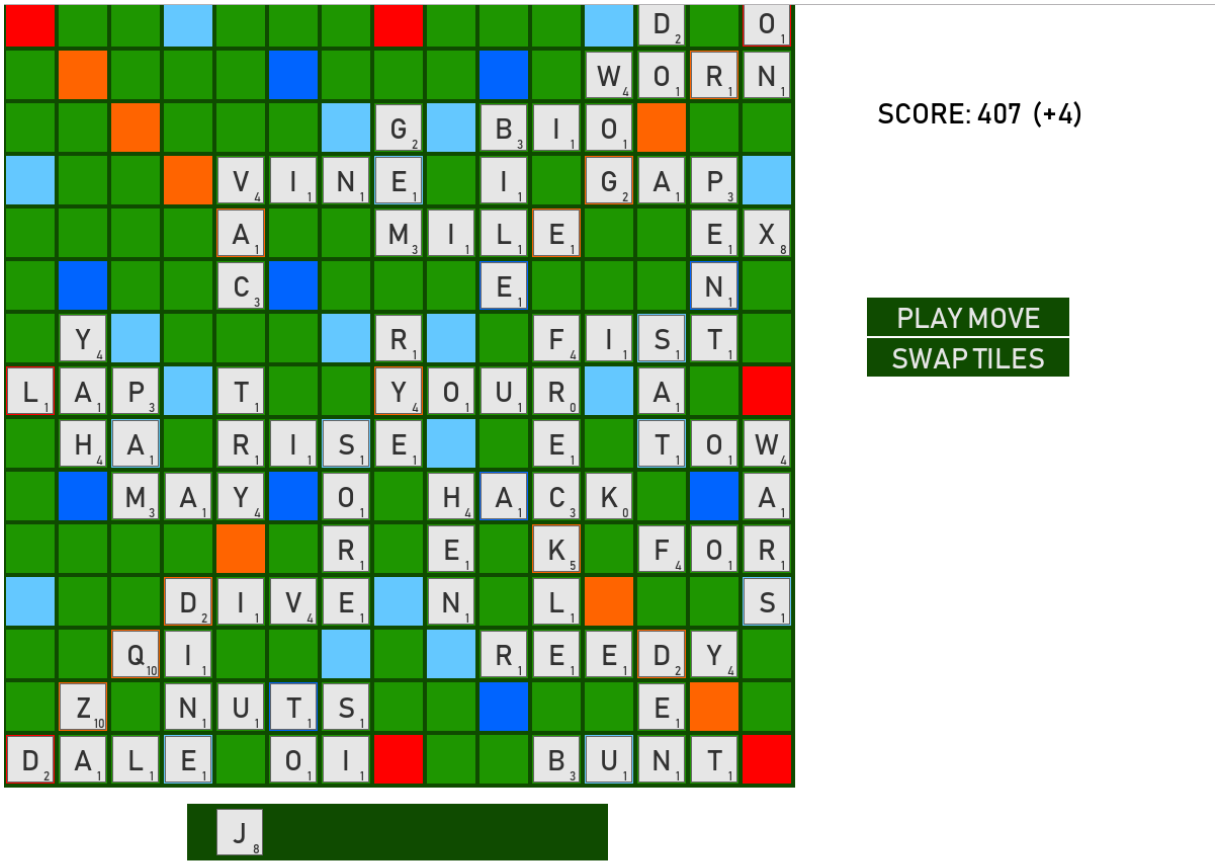
4.10. att. – **Skats pēc divu kauliņu spēlētāja statīvā apmainīšanas vietām.**

Vēlāk spēles partijas gaitā divi spēlētāja statīvā esošie kauliņi – ar burtu “I” un ar burtu “D” – tikuši apmainīti vietām statīvā atrašanās secības ziņā, vispirms uzklikšķinot uz viena no tiem, tādējādi tam nonākot spēlētāja rokā, un tad uzklikšķinot uz otra kauliņa, tādējādi tos apmainot vietām, kas palīdz lietotājam organizēt savu spēlētāja statīvu ērtākai spēlēšanai.



4.11. att. – **Skats pēc veiksmīgas jaunizveidotā krustvārda “FRECKLE” izspēles.**

Savukārt vārds “FRECKLES” atrodas izraudzītajā vārdnīcā un veido jaunu krustvārdu, tāpēc kauliņi paliek laukumā, punkti tiek piešķirti ar vārda punktu dubultotāja bonusa lauciņa ņemšanu vērā, kā arī ir spēlētāja statīvā tikuši pievienoti jauni 6 kauliņi, un kopējam spēlētāja rezultātam ir pievienoti 30 punkti, kā arī šis punktu skaits tiek attēlots kā pēdējā rezultatīvā gājiena punktu skaits.



4.12. att. – **Skats pēc atsevišķas "Scrabble" spēles partijas dziļas izspēlēšanas.**

Šajā skata uzņēmumā var redzēt, ka maiss ir tukšs, jo nenoris pagaidu gājiena izlikšana uz spēles laukuma un jo spēlētāja statīvā atrodas tikai viens neizspēlēts kauliņš, kā arī pēdējā izdarītajā galīgajā gājienā spēlēs laukuma lejas daļā tika izspēlēts kauliņš ar burtu "I", lai izveidotu reizē vairāk nekā vienu jaunu krustvārdu – derīgos krustvārdus "SI" (vertikāli) un "OI" (horizontāli), par kuriem tika iegūts pareizs punktu skaits: (1+1) + (1+1) = 2 + 2 = 4.

Testēšanas procesā netika novēroti nefunkcionālo prasību pārkāpumi – visas lietotnes darbības izpildījās ne ilgāk kā 2 sekunžu laikā, saskarnes iespējas uzskatāmas par vienkāršām un intuitīvām, kā arī sistēma ir izstrādāta prasībās izvirzītajā programmēšanas valodā *Python* 3.x ar *pygame* bibliotēkas palīdzību.

5. Projekta organizācija

Projektā izveidotās sistēmas ““Scrabble” spēles vides implementācija” izstrāde notika vadoties pēc spējās izstrādes programmatūras dzīves cikla modeļa, balstoties tieši *Agile* spējās izstrādes modelī. Projekta gaitā pakāpeniski un iteratīvi tika nospraustas prasības nepieciešamajai sistēmai, kurām ātri tika strādājošs un sākotnējām prasībām atbilstošs prototips ar nedetalizētiem programmatūras prasību specifikācijas, programmatūras projektējuma apraksta un testēšanas dokumentācijas papildinājumiem. Reflektējot par aizvadītās iterācijas rezultātiem, iepriekšējās prasības un dokumentācija tika mainīta un pilnveidota – prasības vairāk papildinātas un precizētas, projektējuma apjoms un detalizācijas pakāpe palielināta –, veicot jaunus testus un atkļūdošanas aktivitātes. Šādā veidā projekts pakāpeniski tika izstrādāts līdz tā gala veidolam.

5.1. Darbietilpības novērtējums

Projekta darbietilpības novērtējums tika veikts vadoties pēc *Intermediate Constructive Cost Model* jeb *COCOMO II* darbietilpības novērtēšanas modeļa. [4] Šis modelis ļauj aplēst sagaidīto projekta darbietilpību personmēnešos pēc sekojošās formulas:

$$E = a \cdot KLoC^{b} \cdot EAF$$

Šajā formulā E apzīmē sagaidīto projekta darbietilpību personmēnešos, *KLoC* apzīmē sagaidīto programmatūras pirmkoda kilorindiņu (tūkstošu rindiņu) skaitu (no angļu val. – “kilolines of code”), *EAF* apzīmē darbietilpības regulējošo faktoru (no angļu val. – “effort adjustment factor”), kas ir atkarīgs 15 dažādiem projekta parametriem, a un b apzīmē koeficientus, kuri ir atkarīgi no projekta veida pēc komandas izmēra, iepriekšējās pieredzes un sarežģītības pakāpes.

Koeficientiem a un b tika izraudzītas vērtības attiecīgi $a = 2{,}4$ un $b = 1{,}05$, jo projekta veids ir vērtējams kā organisks – ar mazu projekta komandu, vienkārši izprotamu būtību un uzdevumiem, ar pietiekamu pieredzi šādu projektu īstenošanā. No priekšstatiem saskaroties ar līdzīga rakstura projektiem, par *KLoC* vērtību tika prognozēta $KLoC = 1{,}6$. *EAF* vērtība tika aplēsta $EAF = 0{,}77$ kā noteicošo projekta parametru reizinājums, kuri tika aplēsti sekojoši:

1) produkta atribūti:
    - nepieciešamās sistēmas paļaujamība – zema (0,88);
    - lietojuma datu bāzes izmērs – zems (0,94);
    - sistēmas sarežģītība – vidēja (1,00);

2) aparatūras atribūti:
    - veiktspējas ierobežojumi – vidēji (1,00);
    - atmiņas ierobežojumi – vidēji (1,00);
    - virtuālās vides mainība – zema (0,87);
    - programmas izpildes laika prasības – zemas (0,94);
3) darbaspēka atribūti:
    - analītiskās spējas – pietiekamas (1,00);
    - programmatūras izstrādes pieredze – ļoti zema (1,29);
    - programmatūras izstrādes spējas – vidējas (1,00);
    - programmēšanas valodas pieredze – zema (1,07);
4) projekta atribūti:
    - programminženierijas metožu pielietojamība – augsta (0,91);
    - programmatūras izstrādes rīku pielietojamība – augsta (0,91);
    - izstrādes laika ierobežojumi – vidēji (1,00).

Pēc *COCOMO II* modelī izmantotās formulas sagaidītā darbietilpība personmēnešos tika iegūta sekojoši:

$$E = a \cdot KLoC^{b} \cdot EAF = 2{,}4 \cdot 1{,}6^{1{,}05} \cdot 0{,}77 = \underline{3{,}03 \text{ sagaidītie personmēneši}}$$

5.2. Kvalitātes nodrošināšana

Kvalitātes nodrošināšanas nolūkos šis dokuments ir ticis izstrādāts balstoties pēc standartu LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis” un LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai” prasībām, šīs sistēmas pirmkods ir veidots pēc vispārpieņemtām labās programmēšanas praksēm un ir atbilstoši, funkciju skaidrojoši komentēts, šī sistēma izstrādes laikā tika rūpīgi testēta un atkļūdota.

5.3. Konfigurāciju pārvaldība

Sistēmas ““Scrabble” spēles vides implementācija” konfigurāciju pārvaldībai tika izmantots konfigurāciju pārvaldības sistēmā *Git* balstītais rīks *GitKraken* 5.0.4., caur kuru tīmeklī balstītā mākoņpakalpojumu vietnē *GitHub* tika augšuplādētas sistēmas izstrādē veiktās izmaiņas, tādējādi novēršot iespējamo konfliktu risku starp datnēm un to saturu un nodrošinot iespējas atjaunināt iepriekšējas datņu versijas, kā arī ar augšupielāžu ziņām veidot pārskatāmu izstrādes procesa vēsturi.

Izmantotā literatūra un avoti

1. R. Čevere, M. Luèkina, LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis”.
2. *Mattel*, “Scrabble”™: Krustvārdu spēle – Spēles noteikumi, 2012.
3. J. Katz-Brown, J. O'Laughlin, “*Quackle Version 1.0.3 Released*”, 5 August 2016. Skat. Internetā (27.05.2019.): http://people.csail.mit.edu/jasonkb/quackle/
4. B. Boehm (1981). *Software Engineering Economics*. Prentice-Hall. ISBN 0-13-822122-7.

Sistēmas ““Scrabble” spēles vides implementācija” programmatūras koda fragmenti:

```
'''
University of Latvia Faculty of Computing Qualification paper:
Implementation of the board game "Scrabble"

RLScrabble.py:
Contains the main part of the application that initiates the "Scrabble"
game.

Author:         Matīss Apinis (ma17058)
Date created:   2019/04/28
Date edited:    2019/05/27
'''

# Libraries:
import pygame as pg

# Local files:
import bag
import board
import player
import menu
import tile

pg.init()
pg.font.init()

SIZE_DISPLAY_WIDTH = 1200
SIZE_DISPLAY_HEIGHT = 900

COLOR_DISPLAY = (255, 255, 255)

pg.display.set_caption("RLScrabble")
DISPLAY_SCRABBLE = pg.display.set_mode((SIZE_DISPLAY_WIDTH,
SIZE_DISPLAY_HEIGHT))
DISPLAY_OVERLAY = DISPLAY_SCRABBLE.convert_alpha()
DISPLAY SCRABBLE.fill(COLOR DISPLAY)

tile.Tile.initialize() #N#

SCORE_LEFT = 865
SCORE_TOP = 100
SCORE_FONT = pg.font.SysFont('bahnschrift', 28)
SCORE_COLOR = (0, 0, 0)

BACKGROUND_COLOR = (255, 255, 255)

### The main function of the application which initiates and terminates
running the software code.
'''
Initializes the Python libraries and other local Python files used for
running the application.
Initializes constants for the application's GUI features.
Calls functions for initializing the game window, game board, game bag,
game player and visually representing these objects as GUI elements.
Runs the pygame environment this system is implemented in continuously
registering mouse motions and mouse clicks made in the environment:
•  If a mouse motion moves the cursor to a user-interactable GUI element,
then it is highlighted.
```

```
•  If a mouse click is executed on a user-interactable GUI element, then
its respective functions are called.
'''
def RLScrabble():
   game_window = menu.Menu(DISPLAY_SCRABBLE)
   game_board = board.Board()
   game_bag = bag.Bag()
   game_player = player.Player(game_board, game_bag)

   game_running = True
   game_over = False

   first_move = True

   tile_picked = None

   draw_game(game_board, game_player, game_over, game_window)

   while game_running:
      mouse_moved = False
      mouse_clicked = False

      pressed_play = False
      pressed_swap = False

      for event in pg.event.get():
         if event.type == pg.QUIT:
            game_running = False
         elif event.type == pg.MOUSEMOTION:
            move_x_position, move_y_position = event.pos
            mouse_moved = True
         elif event.type == pg.MOUSEBUTTONUP:
            click_x_position, click_y_position = event.pos
            mouse_clicked = True

      if mouse_moved:
         game_window.set_button_highlight(DISPLAY_SCRABBLE,
move_x_position, move_y_position)

      if mouse_clicked:
         pressed_button = game_window.get_button_name(click_x_position,
click_y_position)

         if pressed_button == menu.Menu.PLAY_MOVE:
            pressed_play = True
         elif pressed_button == menu.Menu.SWAP_TILES:
            pressed_swap = True

      if pressed_play and not game_over:
         move_success = game_player.play(first_move)
         if move_success == "GAME_OVER":
            game_over = True
            game_finalize(game_player)
         elif move_success:
            first_move = False

         draw_game(game_board, game_player, game_over, game_window)

      if pressed_swap and not game_over:
         game_board.discard_move()
         game_player.shuffle()

         draw_game(game_board, game_player, game_over, game_window)
```

```
      if mouse_clicked and not game_over:  #C#
         tile_picked = pick_tile(click_x_position, click_y_position,
tile_picked, game_board, game_player)
         draw_game(game_board, game_player, game_over, game_window)

      if game_over:
         game_running = False

      draw_parts(game_board, game_player, game_over) #C#
      pg.display.update()

RLScrabble()
pg.quit()
quit()
```

```
'''
University of Latvia Faculty of Computing Qualification paper:
Implementation of the board game "Scrabble"

bag.py:
Contains the class modelling the game bag of tiles.

Author:         Matīss Apinis (ma17058)
Date created:   2019/04/28
Date edited:    2019/05/27
'''

# Libraries:
from random import shuffle

# Local files:
import tile

class Bag:
   # Dictionary of tiles by letter as key with 2-tuple as value containing
its score value and copy count in bag:
   def __init__(self):
      self.tiles = []

      self.add('A', 1, 9)
      self.add('B', 3, 2)
      self.add('C', 3, 2)
      self.add('D', 2, 4)
      self.add('E', 1, 12)
      self.add('F', 4, 2)
      self.add('G', 2, 3)
      self.add('H', 4, 2)
      self.add('I', 1, 9)
      self.add('J', 8, 1)
      self.add('K', 5, 1)
      self.add('L', 1, 4)
      self.add('M', 3, 2)
      self.add('N', 1, 6)
      self.add('O', 1, 8)
      self.add('P', 3, 2)
      self.add('Q', 10, 1)
      self.add('R', 1, 6)
      self.add('S', 1, 4)
      self.add('T', 1, 6)
      self.add('U', 1, 4)
      self.add('V', 4, 2)
```

```
      self.add('W', 4, 2)
      self.add('X', 8, 1)
      self.add('Y', 4, 4)
      self.add('Z', 10, 1)
      self.add(' ', 0, 2)

      shuffle(self.tiles)

   '''
   Pulls a new tile from the game bag if it's not empty.
   '''
   def pull_tile(self):
      if self.is_bag_empty():
         return None

         return self.tiles.pop()

   '''
   Checks whether the game bag is empty or not.
   '''
   def is_bag_empty(self):
      return len(self.tiles) == 0

   '''
   Randomizes the sequence in which tiles would be pulled out of the game
bag.
   '''
   def shuffle(self):
      shuffle(self.tiles)

   def return_tile(self, tile):
      self.tiles.append(tile)

   '''
   Adds n copies of a newly created game tile to the game bag.
   '''
   def add_tiles(self, letter, points, n):
      for i in range(n):
         self.tiles.append(tile.Tile(letter, points))

''' Unused Latvian tile set:
tile_set = {
        'A': (1, 11),
        'Ā': (2, 4),
        'B': (5, 1),
        'C': (5, 1),
        'Č': (10, 1),
        'D': (3, 3),
        'E': (1, 6),
        'Ē': (4, 2),
        'F': (10, 1),
        'G': (5, 1),
        'Ģ': (10, 1),
        'H': (10, 1),
        'I': (1, 9),
        'Ī': (4, 2),
        'J': (4, 2),
        'K': (2, 4),
        'Ķ': (10, 1),
        'L': (2, 3),
        'Ļ': (8, 1),
        'M': (2, 4),
        'N': (2, 4),
```

```
        'Ņ': (6, 1),
        'O': (3, 3),
        'P': (2, 3),
        'R': (1, 5),
        'S': (1, 8),
        'Š': (6, 1),
        'T': (1, 6),
        'U': (1, 5),
        'Ū': (6, 1),
        'V': (3, 3),
        'Z': (3, 2),
        'Ž': (8, 1),
        '':  (None, 2)
}
'''
```

Kvalifikācijas darbs „***"Scrabble" spēles vides implementācija***" izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: ***Matīss Apinis*** _____ **___.05.2019**.

Rekomendēju darbu aizstāvēšanai
Darba vadītājs/a: ***Dr. dat. Kārlis Freivalds*** _____ **___.05.2019.**

Recenzents: ***Jānis Baiža***

Darbs iesniegts 27.05.2019.
Kvalifikācijas darbu pārbaudījumu komisijas sekretāre: *Darja Solodovņikova* _____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē
___.06.2019. prot. Nr. _____

Komisijas sekretārs(-e): _____