**Rīgas Tālmācības Vidusskola**

**“ Bitu Piedzīvojums”**

**izstrāde Python programmēšanas valodā**

**Darba autore: Luīza Pihtina 11.klase**

**2024/2025. gads**

**Projekta ideja:**

Izveidot 2D platformera tipa digitālo spēli ar četriem dažādiem līmeņiem, kas ietver šķēršļus, lavu un mērķus. Katrs līmenis piedāvā unikālu grafisku vidi, kurā spēlētājam būs jāpārvietojas, jālec, jāizvairās no bīstamiem objektiem, jāsavāc monētas un jāsasniedz izejas durvis, lai pārietu uz nākamo līmeni.

Līmeņu struktūra un saturs tiks definēts ar JSON datnēm, kas nodrošinās elastību un vieglu līmeņu konfigurēšanu. Kā pamata bibliotēka tiks izmantota pyGame. Pirmkodā katra “Level” datne saturēs informāciju par pozīcijām, objektiem un videi raksturīgajām īpašībām – tostarp spēlētāja sākuma pozīciju, platformām, ienaidniekiem, monētām un mērķa sasniegšanas vietu “exit point”.

Šī pieeja ļaus ne tikai veidot dinamisku un interaktīvu spēles pieredzi, bet arī atvieglos jaunu līmeņu pievienošanu un rediģēšanu bez nepieciešamības mainīt pirmkodu.

**Projekta mērķis:**

Izmantojot Python programmēšanas valodu, izveidot loģisku, lietotājam draudzīgu, intuitīvi saprotamu un interesantu 2D spēli. Spēles mērķis ir veikt triviālus uzdevumus un pārbaudīt pyGame bibliotēkas sniegtās grafiskā dizaina un komandu iespējas.

1. **Programmatūras apraksts**
   1. **Projekts (spēle) ir paredzēts sekojošiem mērķauditorijām:**

* **Spēlētājiem**, kuri vēlas izklaidēties un izaicināt savas reakcijas spējas. Lai nodrošinātu aizraujošu un interaktīvu laika pavadīšanu, attīstot reakciju, uzmanību un veiklību. Spēles varonim pārvietojoties pa līmeni, savācot objektus, izvairoties no ienaidniekiem un sasniedzot galamērķi – visu to kombinējot ar jautru dizainu un progresīvo sarežģītību.
* **Programmētājiem un spēļu izstrādātājiem**, kuri vēlas iepazīt pyGame bibliotēkas un JSON datņu praktiskās iespējas un pielietojot savas zināšanas veidot līdzīgus projektus. Projekts kalpo kā praktisks piemērs spēļu izstrādes pamatos: sadursmju detektēšana, objektu loģika, animācijas, spēles cikls. Kā arī demonstrē, kā strukturēti organizēt datus (JSON datnes); izveidot modulāru spēles kodu; ievēst fiziku un loģiku spēlēs.
* **Izglītojošiem nolūkiem**, kā mācību resurss, kas kalpo par praktisku, izglītojošu, reklāmas resursu programmēšanas apgūšanai, lai apgūtu spēles loģikas izstrādi, datu struktūras un spēles līmeņu pārvaldību. Kopumā tas ir piemērots spēļu izstrādes vai datu struktūru kursu reklāmām, ja spēlei pievienot kontaktinformāciju.
  1. **Šis projekts ir nepieciešams, jo tas:**
* sniedz aizraujošu spēli izklaidējošiem mērķiem, attīstot reakciju, uzmanību un veiklību;
* kalpo par praktisku izglītojošu resursu programmēšanas apgūšanai;
* demonstrē JSON priekšrocības līdzīgu sistēmu izveidē - datu strukturēšanu, modulāru arhitektūru, vienkāršu lasīšanu un vizuālo elementu pārvaldību;
* ir piemērots gan individuālai izstrādei, gan izglītošanai un gan produktu reklamēšanai (ja monētas nomaina uz produkta foto).
  1. **Kādu problēmu projekts risina/atvieglo?**
* Iemaņu attīstīšana darbam ar JSON datnēm, kas ļauj ērti pievienot jaunus līmeņus bez pirmkoda izmaiņām.
* Atvieglo spēļu izstrādes izpratni, izmantojot skaidru arhitektūru un līmeņu datu nošķiršanu no spēles loģikas.
* Pygame bibliotēkas izmantošana.
* Piedāvā interaktīvu un relaksējošu vidi atslodzei no mācībām, kas palīdz nomainīt fokusu un pēc laika atsākt mācību procesu.
  1. **Projekta prasību specifikācija**

**Tabula Nr.1**

| **Galvenās spēles funkcijas** | **Prasības** |
| --- | --- |
| Spēlētāja vadība | Spēlētājs var kustēties pa kreisi, pa labi, lēkt, saskarties ar citiem objektiem (lava). |
| Objekti līmenī | Cilvēciņš – spēlētājs, monētas, grīdas/platformas, izejas durvis. |
| Mērķis | Savākt monētas un sasniegt izejas durvis, lai pārietu uz nākamo līmeni. |
| Spēles loģika | * Spēlētājs pārvietojas pa noteiktām platformā horizontāli pa labi vai pa kreisi, kā arī spēj palekties jeb pārvietoties vertikāli uz augšu. * Saskarsme ar lavu izraisa spēles līmeņa beigšanos ar zaudējumu. * Ja izpildās AND princips jeb gan savāktas visas monētas, gan sasniegtas durvis – spēlēs līmeņa uzdevums ir izpildīts un spēlētājs dodas uz nākamo līmeni. |
| Līmeņu dati un struktūra | Līmeņi ir definēti atsevišķos JSON failos, t:i ka katrā līmenī tiek izmantots JSON fails ar   * + - Objektu veidiem un koordinātām.     - Grīdu, šķēršļu, monētu un lavu pozīcijām.     - Izejas durvju atrašanās vietu |
| Līmeņu pārvaldība | Spēle dinamiski lasa un izveido līmeņus no JSON datiem. |
| Lietotāja saskarne | Uz displeja tiek norādīts savākto monētu skaits, iespēju (dzīvību) vai neveiksmju skaitu, līmeņa atjaunošanas (restart) taustiņš R vai ekrāna poga.  Ziņojumi: Tu zaudēji! Uzvara! Līmenis ielādēts! |
| Grafika | Vienkārša 2D grafika ar atšķirīgiem vizuālajiem objektiem. |
| Paplašināmība | Iespēja pievienot jaunus līmeņus pievienojot jaunu JSON datni. |
| Atskaite par kļūdām | Konsolē tiek ziņots par kļūdām JSON datnē vai objekta ielādēšanā. |
| Uzvara/zaudējums | Uzvara – sasniegt durvis; zaudējums – pieskaras lavai. |
| Platforma | Paredzēta darbībai datorā (ar tastatūras ievadi). |

1. **Programmatūras projektējuma apraksts**
   1. **Ievads**

Šis dokumenta mērķis ir aprakstīt 2D platformera spēles arhitektūru, loģiku un komponentes, kas nepieciešamas spēles izstrādei, pamatojoties uz prasību specifikāciju. Spēle paredzēta datoram ar tastatūras ievadi un dinamisku līmeņu ielādi no JSON failiem.

* 1. **Sistēmas pārskats**

Spēle sastāv no vairākiem komponentiem:

* Spēlētāja vadības modulis
* Fizikas dzinējs (kustība, sadursmes)
* Līmeņu ielādes un pārvaldības sistēma
* Objektu pārvaldība (platformas, monētas, lava, durvis)
* Spēles stāvokļa kontrole (uzvara/zaudējums)
* Kļūdu ziņošana
  1. **Galvenās Komponentes**
     1. **Spēlētāja Vadība**
* **Ievade**: Tastatūra (bultiņas vai WASD).
* **Kustības funkcijas**:
  + moveLeft()
  + moveRight()
  + jump()
* **Sadursmes** **noteikšana** ar vidi (platformas, lava, objekti).
  + 1. **Objekti Līmenī**
* **Grīdas/Platformas**: Stacionāri objekti, uz kuriem spēlētājs var staigāt/lēkt.
* **Monētas**: Savācamas, uzlabo rezultātu.
* **Lava**: Saskare izraisa zaudējumu.
* **Izejas durvis**: Ja visas monētas savāktas – ļauj pāriet uz nākamo līmeni.
  + 1. **Spēles Loģika**
* **Atkārtots mēģinājums** : Ja spēlētājs pieskaras lavā – atjaunojas līmenis, tam izmantota komanda (restartLevel()).
* **Uzvaras noteikumi**:
  + Savāktas visas monētas.
  + Spēlētājs pieskaras durvīm.
  + Funkcija: check Victory() – pārbauda uzvaras nosacījumus.
* **Zaudējums**: Saskare ar lavu izraisa zaudējumu (Game Over()).
  + 1. **Grafika**
* 2D režīms, vienkārši vizuālie attēli vai krāsu apzīmējumi.
* Atšķirami objekti (cilvēciņš-spēlētājs, platformas, lava, monētas, durvis).
  1. **Kļūdu Apstrādes un Ziņošanas**
     1. **Kļūdu tipi:**
* Trūkst JSON datni
* Nepareiza JSON struktūra
* Nevar atrast līmeņa objektus
* Nespēj ielādēt datni
  + 1. **Kļūdu ziņošana:**
* Ja JSON struktūra ir bojāta:
  + Ziņojums konsolē: console.error ("Level load error").
* Ja objekts nav atrasts vai netiek ielādēts pareizi:
  + Reģistrē kļūdu konsolē.
* Ekrānā (lietotājam draudzīgs ziņojums)
  1. **Tehniskā Platforma**
* **Valoda**: Python
* **Bibliotēka** : Pygame
* **Datu struktūra** : JSON līmeņiem
* **Palaišana**: python main.py
* **Ievades ierīces**: Tastatūra
* **Darbojas operacionālā sistēmā: (Windows, Mac, Linux)**
  1. **Stiepļ – rāmju diagramma**

+----------------------------------------------------+

| Monētas: 3/5 Neveiksmes: 1 [Restarts] |

| |

| |

|  |

| ░░░░░░░░░░░░░░░░ |

|  |

| ░░░░░░░░░░░░░░░ |

| 🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥 |

| |

| [ Zaļas durvis ] |

+----------------------------------------------------+

**Simboli:**

*  – spēlētājs
*  – monēta
* ░ – platformas
* 🔥 – lava

[ Zaļas durvis ] – izejas durvis

* 1. **Paplašināmība un Uzturēšana**
* Jaunu līmeņu pievienošana bez pirmkoda izmaiņām.
* Jaunu objektu (piemēram, lava) pievienošana ar paplašinājumiem līmeņu JSON un objektu klasēs.
* Spēles loģiku var paplašināt ar punktu sistēmu, taimeri u.c.
* Pateicoties JSON spēles paplašinājuma var veikt dažādas izmaiņas un uzlabojumi, taču būtiski jauninājumi vadībā vai spēles loģikā izdarāmi pirmkoda izmaiņās.
  1. **Testēšanas plāns**

**Tabula.Nr.2**

| **Funkcija / Komponents** | **Testa scenārijs** | **Sagaidāmais rezultāts** |
| --- | --- | --- |
| Spēlētāja kustība | Nospiež taustiņus ←, →, Space | Spēlētājs kustas un lec |
| Saskarsme ar lavu | Spēlētājs pieskaras lavā | Līmenis restartējas, parādās "Tu zaudēji!" |
| Monētu savākšana | Spēlētājs savāc monētu | Monētas pazūd, UI atjauninās |
| Izejas durvis | Spēlētājs savāc visas monētas un ieiet durvīs | Spēle pāriet nākamajā līmenī |
| JSON faila kļūda | Ielādē bojātu JSON failu | Konsolē tiek parādīts kļūdas ziņojums |
| Restartēšanas funkcija | Nospiež taustiņu R vai restart pogu | Līmenis sākas no jauna |
| UI informācija | Spēles laikā savāc monētas, krīt lavā | UI pareizi rāda monētas un neveiksmju skaitu |
| Līmeņu maiņa | Izejas durvis sasniegtas ar visām monētām | Tiek ielādēts nākamais līmenis no JSON |
| Programmas palaišana | Palaist main.py failu | Spēle startējas kā darbvirsmas lietotne |
| Ieeja durvīs bez monētām | Spēlētājs iet durvīs nesavācot visas monētas | Līmenis nepārslēdzas, notiek ignorēšana |
| Nepareizi objekta dati | Trūkst koordinātas objektam JSON failā | Konsolē tiek parādīta kļūda, objekts netiek ielādēts |

* 1. **Lietotāja rokas grāmata**

**Tabula.Nr.3**

| **Sadaļa** | **Informācija** |
| --- | --- |
| **Spēles nosaukums** | Bitu Piedzīvojums |
| **Spēles mērķis** | Savākt visas monētas un nokļūt līdz durvīm, lai pārietu uz nākamo līmeni. |
| **Platforma** | Datorprogramma (darbvirsmas lietotne), vadāma ar tastatūru |
| **Prasības** | Python 3.10+, Pygame 2.x, Windows/Linux/MacOS |
| **Programmas palaišana** | * + Instalē Python un Pygame → pip install pygame   + Izpildi: python main.py |
| **Vadība** | ← / → — kustība  Space — lēciens  R — restartēt līmeni  Esc — iziet |
| **Spēles objekti** | Spēlētājs  Monētas  Lava  Durvis  Platformas |
| **UI informācija** | Monētu skaits  Neveiksmju skaits  Restartēšanas poga |
| **Uzvara** | Savāktas visas monētas un sasniegtas izejas durvis |
| **Zaudējums** | Saskarsme ar lavu – līmenis restartējas |
| **JSON līmeņi** | Līmeņi glabājas mapē /levels/ kā .json faili ar objektu koordinātām |
| **Līmeņu paplašināšana** | Pievieno jaunu .json failu ar pareizu struktūru. Spēle to nolasīs automātiski. |
| **Biežākās kļūdas**  **X !!!** | **X !!!** Spēle nepalaižas – pārbaudi Python & Pygame  **X !!!** JSON kļūda – pārbaudi faila sintaksi  **X !!!** Nereāģē taustiņi – fokusējies uz spēles logu |
| **Kļūdu atspoguļošana** | Kļūdas parādās konsolē |
| **Spēles beigas** | Kad visi līmeņi ir izieti – tiek parādīts uzvaras ziņojums |

1. **Projekta Dizains – 2D Platformera Spēle**
   1. **Lietotāja Saskarnes Dizains**

* Ekrāna izmērs: *800x600 px*
* Kadrēšana režīms: *60 fps*
* Monētu skaits*: Monētas: 5/10*
* Dzīvību skaits vai neveiksmju skaits: *Neveiksmes: 2*
* Restart poga (vai R taustiņš)
* Ziņojumi: *Uzvara!, Tu zaudēji!, Kļūda ielādējot līmeni*
  1. **Grafikas un Vizualizācijas Plāns**
     1. **Vienkārši attēli vai krāsaini bloki**:
     + Objekti tiek ielādēti no PNG paplašinājuma datnēm:
       - Spēlētājs - cilvēciņš
       - Platformas –brūnas ar zaļu taisnstūra formas
       - Monētas – dzelteni apļi
       - Lava – oranžs laukums
       - Durvis – brūnas
       - Izvēlnes pogas
     + Spēles karte tiek uzģenerēta katru reizi no jauna (world.draw() u.c.).
     + Tiek izmantoti skaņas faili, attēli, kolīzijas pārbaudes (vai tēls pieskaras kādam objektam).
  2. **Pygame 2D bibliotēka grafikai**

Šī bibliotēka atbalsta attēlu ielādēšanu ar paplašinājumiem JPG, PNG, BMP, spēlē tika izmantots PNG paplašinājums. Kā arī viegli ir zīmēt dažāda veida ģeometriskas formas, piem. izvēlnes pogas. Ar to var noteikt izšķirtspēju ar komandu – **(*diplay.set\_mode*).** Izmantojot spirte. Sprite var noteikt definēto objektu izvietojumi un loģiku. Savukārt ar spirte. Group var pārvaldīt vairākus objektus vienlaicīgi.

* 1. **Programmēšana** **Valodas/platformas:**

Izmantojot programmēšanas valodu Python ar Pygame, JSON bibliotēku palīdzību, šī ir atvērtā pirmkoda programmatūra, kura pieejama par brīvu. Tā dod iespējas salīdzinoši viegli veidot pietiekoši sarežģītas spēles, ar sazarotu (daudz apakšklašu) loģiku.

* 1. **Programmas atkļūdošana:**

Kolīzija starp pārbaudīto spēlētāju un spēles vidi, monētas, lava – pārbauda ar funkciju (*colliderect*).

Print() funkciju izmanto paziņojumos ģenerēšanai, kuros iekļauti: uzvaras nosacījumi, punktu skaits. Testēta poga funkcionalitāte, spēlētāja lēkšana, spēļu līmeņu nomainīšana un skaņas.

Pārbaudītas spēles funkcijas, izmantojot **print() funkciju** un **vizuālu novērošanu** spēles laikā:

* Kolīzijas ar monētām, lavām un durvīm pārbaudītas ar **colliderect()** un **spritecollide().**
* Spēlētāja lēkšana un skaņas testētas ar **print()** pirms **Sound.play().**
* Līmeņu maiņa pārbaudīta, drukājot līmeņa numuru.
* Pogas (**start/restart/exit**) testētas ar klikšķi un ziņojumu konsolē.
* Punktu skaits un spēles beigu rezultāti rādīti ar **print()** un uz ekrāna.
  1. **Galvenās klases un komponentes:**
* “World” - klase, kas pārvalda līmeņa izkārtojumu no JSON datnes
* Player ir galvenais spēles varonis, ar lekšanas spējām, smaguma spēku fiziku un sadursmēm (kolīzijām)
* “Lava”, “Exit”, “Coin” – objektu grupas ar savām īpašībām un funkcijām
* Button - starta, restartēšanas un izejas pogas

Zeme (grass/dirt) – pa kurām spēlētājs var staigāt  
Lava (tile6.png) – ja pieskaras, spēle beidzas  
Dimanti (coin.png) – spēlētājs tos vāc, lai iegūtu punktus  
Durvis (door1.png) – tām pieskaroties, tiek nākamajā līmenī

1. **Secinājumi un nākotnes ierosinājumi**

Paveicot projektu un analizējot apgūto – secinu, ka šādi projekti palīdz apkopot un sistematizēt teorētiskās zināšanas par programmēšanu. Tieši šādi uzdevumi ir interesanti un praktiski palīdz veidot komandu un loģikas virknējumus. Papildus motivāciju rada savu iztēles, izvēlēto attēlu un grafikas paņēmienu pārnešana digitālā pasaulē un vēl vairāk motivē, tas ka spēle patīk apkārtējiem.

Šajā dokumentā esmu apkopojusi principus un spēles izklāstu, uz kādiem principiem tas ir veidots. Lielu priekšrocību dod JSON un Pygame bibliotēku izmantošana, tas gan dod iespēju rediģēt tieši spēles loģisku un izklaidējošo aspektu, pievienojot klāt līmeņus un uzdevumus, gan dod iespēju ar jau gataviem prototipiem. Pygame platforma ļava pietiekoša ātri izprast komandas, funkcijas un struktūru, lai veidotu spēli, kura risina sākotnēji paredzētos mērķus.

Uzsākot projektu, bija svarīgi veikt spēles cikla un JSON līmeņu datu apsvērumus, lai nerastos kļūdas tālākā darbība. Galvenais varonis tika animēts atsevišķi, tas palīdz veidot tādu vizualizāciju, kura ir saistoša mērķauditorijai. Kolīziju pārbaudes sistēma apvienojot ar loģiku rada spēles noteikumus, pārveidojot katru darbību koordinātu plaknē. Vairāku objektu mijiedarbība un fizikas likumu modulis palīdz veikt reālistiskāku spēles varoņa darbību. Ir kombinētas programmēšanas un dizaina prasmes grafikas datu ievadīšanā, skaņas efekti lecot, zaudējot vai uzvarot. Mani izveidotais prototips palīdz sekmīgāk attīstīt loģisko domāšanu un radīt pamatus programmēšanas mācīšanai interaktīvā veidā.

Manuprāt, nākotnes ierosinājumi var būt šādi: vairāk līmeņu ar paaugstinātu grūtību. Spēles varoņa animēta dzīvība un spēka uzlabojumi; uzvaras ekrāns kopējo punktu skaitīšanai. Piešķirt papildus priekšmetus par ideāli paveiktiem uzdevumiem, un radīt unikālus līmeņus (BOSS level), lai radītu intrigu un motivētu spēlētājus dalībai spēlē.