МІНІСТЕРСТВО ОСВІТ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра АСУ

КУРСОВА РОБОТА

з дисципліни

«Методи і засоби комп'ютерних інформаційних технологій»

на тему:

«Створення 2D JAVA-гри»

Виконав:
студент групи КН-31
Яремчук Микола

Прийняв:
професор каф. АСУ
Різник В. В.

Львів – 2014

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра автоматизованих систем управління

Завдання на курсову роботу

з дисципліни *«Методи і засоби комп'ютерних інформаційних технологій»*

Прізвище, ім'я студента	Яремчук Микола
Група	КН-31
Тема курсової роботи	Створення 2D JAVA - гри

Спеціальна частина завдання:

1. Створити 2D гру мовою JAVA.

2. Описати розробку поетапно.

Керівник_____ Різник В. В.

Студент_____ Яремчук М. В.

Анотація

Яремчук М. В. «Створення 2D JAVA-гри». Курсова робота – НУ «Львівська політехніка», АСУ, дисципліна: «Методи і засоби комп'ютерних інформаційних технологій», 2014 р.

Курсова робота складається з 45 сторінок, містить графічні пояснення до методів анімації зображень. Для написання курсової роботи були використані матеріали з 10-х літературних джерел.

Метою цієї роботи є створення 2D JAVA-гри.

ЗМІСТ

Вступ…….....ст.5

1.1.Що таке JAVA?.....ст.6

1.2.Що таке Eclipse? …….....ст.7

2. Етапи розробки 2D JAVA-гри……...……ст.9

2.1 Етап 1(Створення проекту, підключення папки з зображеннями, створення головного класу main, створення головного вікна нашої гри).....ст.9
2.2 Етап 2(Створення дороги (клас Road))……ст.11

2.3 Етап 3(Створення автомобіля(клас Player))…….ст.13

2.4 Етап 4(Додавання другого шару дороги та його зациклення з першим шаром дороги + реалізація обробки натиснення на клавіші клавіатури).....ст.17
2.5 Етап 5(Організація правильної обробки натиснення клавіш).....ст.24
2.6 Етап 6(Створення суперників)…….…...…….…ст.25

2.7 Етап 7(Накладення відповідного зображення нашого автомобіля при повороті ліворуч та праворуч, створення спідометра , додавання умови перемоги).....ст.32
2.8 Етап 8(Збереження нашої гри, збірка проекту)……..…..……ст.38

Висновок…….....ст.39

Список використаної літератури…….....ст.40

Додаток А. Код програми……..….....ст.41

Вступ

Згадаймо, спочатку як з'явилися прості ігри, створені для розваги та відпочинку. Вони розвивали увагу, швидкість реакції. Дорослішаючи, дитина починала цікавитися більш складними - інтелектуальними іграми. Так, проходження лабіринтів може бути непоганим розумовим тренуванням, що розвиває логічне мислення. Їм на зміну прийшли складніші ігри - стратегічні (стратегії), вони вже істотно відрізнялися від попередніх, оскільки моделювали реальність, в якій знаходився гравець. Виходячи з назви, можна сказати, що такі ігри вчать планування, добре розвивають аналітичне мислення.[3]

Тому діти, які грають у комп'ютерні ігри, відрізняються широтою кругозору: у них добре розвинене уявлення про навколишній світ, і він більше відповідає світогляду дорослих людей. Такі "комп'ютерно-розвинені" діти, зазвичай, випереджають своїх однолітків у психічному розвитку, легше засвоюють навчальний матеріал, і впевненіші у своїх знаннях.[7]

Java-ігри — ігри написані на мові Java. Їх перевагою є те, що вони можуть запускатися в будь-якій операційній системі, де встановлена віртуальна Java-машина. Найбільшого поширення Java-ігри отримали на мобільних телефонах, смартфонах і КПК, що підтримують платформи Java, Symbian і Windows Mobile. [8]

Створення 2D ігор сьогодні є надзвичайно корисним. Вони не викликають залежності, а вміщують у собі лише найкращі якості : пізнавальні, логічні, розважальні та багато інших. Також великою перевагою створення 2D ігор є те, що вони завжди були популярними серед людей різного віку, адже кожен зможе знайти щось цікаве для себе. Такі ігри не потребують величезних характеристик вашого комп’ютера, і вони з легкістю встановлюються на мобільні пристрої навіть старіших версій (платформи Java, Symbian і Windows Mobile). І звідси випливає ще одна перевага : такі ігри з легкістю наповнять бюджет нашої країни, адже вони є популярними і їх вартість не буде великою, тобто кожен, незалежно від матеріального становища, зможе придбати собі таке задоволення. І ще одною позитивною характеристикою створення 2D ігор є те, що вони не вимагають величезних капіталовкладень і довготривалих термінів розробки[1]. Отже, враховуючи сьогоднішню ситуацію в Україні, створення 2D ігор є необхідним у такий нелегкий час. Саме вони допоможуть українцям у такі важкі роки розслабитись та відпочити після важкого буденного життя і виведуть Україну на світовий ринок у ІТ сфері та покажуть іншим країнам, що українці – ерудовані, творчі й талановиті та заслуговують на краще європейське життя.

1.1 Що таке JAVA?

Java (*джава* , *ява* , жарг. *жаба*) - об'єктно-орієнтована мова програмування, що розробляється компанією Sun Microsystems з 1991 року і офіційно випущений 23 травня 1995 року.[1]

Спочатку нова мова програмування називалася Oak і розроблялася для побутової електроніки, але згодом була перейменована у Java і стала використовуватися для написання аплетів, додатків і серверного програмного забезпечення.[5]

Програми на Java можуть бути трансльовані в особливий байт-код, що виконується на віртуальній джава-машині (JVM) - програмі, що обробляє байтовий код у інструкції, що передаються устаткуванню, як інтерпретатор, але з тією відмінністю, що байтовий код на відміну від тексту обробляється значно швидше.

Перевага такого способу виконання програм - у повній незалежності байт-кода від ОС і устаткуванні, що дозволяє виконуватись Java додатку на будь-якому пристрої, який підтримує віртуальну машину. Іншою важливою особливістю технології **Java** є вельми гнучка система безпеки, завдяки тому, що виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження програми (наприклад спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером) викликають негайне переривання.[4]

Це дозволяє користувачам легко завантажувати програми написані на Java на їх комп'ютери (або інші пристрої, наприклад мобільні телефони) з невідомих джерел, при цьому не побоюючись зараження вірусами, чи втраті цінної інформації, і т.д.

Часто до недоліків цього підходу відносять те, що виконання байт-коду віртуальною машиною може знижувати продуктивність програм і алгоритмів, реалізованих на мові Java. Дане твердження можна сміливо назвати справедливим для перших версій віртуальної машини Java, проте останнім часом воно практично втратило актуальність.

Цьому сприяв ряд удосконалень: застосування технологій JITs (Just-In-Time compilers) , які дозволяють переводити байт-код в машинний код під час виконання програми з можливістю збереження версій класу в машинному коді, широке використання native-коду в стандартних бібліотеках, а також апаратні засоби, що забезпечують прискорену обробку байт-коду (наприклад технологія Jazelle, яка підтримується деякими процесорами фірми ARM).

У Java існують 3 основні технології:[9]

- J2EE - Java Enterprise Editon, для створення серверного забезпечення рівня підприємства;
- J2SE - Java Standard Editon, для створення звичайних, не-серверних додатків;
- J2ME - Java Micro Edition, для використання в пристроях, обмежених після обчислювальної потужності, в т.ч. мобіл ьних телефонів.

Основні можливості:

- вбудовані прості класи, такі як масив, список, стек і т.п.
- наявність простих засобів створення мережевих додатків (в т.ч. використовуючи протокол RMI)
- наявність класів, що дозволяють створювати http-запити і відповіді.
- вбудовані в мову засобу створення багатопотокових додатків
- уніфікований доступ до баз даних на основі JDBC і SQLJ

1.2 Що таке Eclipse?

Eclipse (вимовляється «і-кліпс», від англійського «затемнення») — вільне модульне інтегроване середовище розробки програмного забезпечення. Розробляється і підтримується Eclipse Foundation і включає проекти, такі як платформа Eclipse, набір інструментів для розробників на мові Java, засоби для управління сирцевими кодами, візуальні побудовники GUI тощо.[3]

Написаний в основному на Java, може бути використаний для розробки застосунків на Java і, за допомогою різних плагінів, на інших мовах програмування,включаючи Ada, C, C++, COBOL, Fortran, Perl, PHP, Python, R, Ruby (включно з каркасом Ruby on Rails), Scala, Clojure та Scheme. Середовища розробки зокрема включають Eclipse ADT (Ada Development Toolkit) для Ada, Eclipse CDT для C/C++, Eclipse JDT для Java, Eclipse PDT для PHP.

Eclipse являє собою фреймворк для розробки модульних кросплатформових застосунків із низкою особливостей:

- можливість розробки ПЗ на багатьох мовах програмування (рідною є Java);

- крос-платформова;

- модульна, призначена для подальшого розширення незалежним розробниками;

- з відкритим сирцевим кодом;

- розробляється і підтримується фондом Eclipse, куди входять такі постачальники ПЗ, як IBM, Oracle, Borland.

Спочатку проект розроблявся в IBM як корпоративний стандарт IDE, настановлений на розробки на багатьох мовах під платформи IBM. Потім проект було перейменовано на Eclipse і надано для подальшого розвитку спільноті.[6]

Eclipse насамперед повноцінна Java IDE, націлена на групову розробку, має засоби роботи з системами контролю версій (підтримка CVS входить у поставку Eclipse, активно розвиваються кілька варіантів SVN модулів, існує підтримка VSS та інших). З огляду на безкоштовність, у багатьох організаціях Eclipse — корпоративний стандарт для розробки ПЗ на Java.

Друге призначення Eclipse — служити платформою для нових розширень. Такими стали C/C++ Development Tools (CDT), розроблювані інженерами QNX разом із IBM, засоби для підтримки інших мов різних розробників. Безліч розширень доповнює Eclipse менеджерами для роботи з базами даних, серверами застосунків та інших.

З версії 3.0 Eclipse став не монолітною IDE, яка підтримує розширення, а набором розширень. У основі лежать фреймворки OSGi, і SWT/JFace, на основі яких розроблений наступний шар — платформа і засоби розробки повноцінних клієнтських застосунків RCP (Rich Client Platform). Платформа RCP є базою для розробки різних RCP програм як торент-клієнт Azareus чи File Arranger. Наступний шар — платформа Eclipse, що є набором розширень RCP — редактори, панелі, перспективи, модуль CVS і модуль Java Development Tools (JDT).

Eclipse написана на Java, тому є платформо-незалежним продуктом, крім бібліотеки графічного інтерфейсу SWT, яка розробляється окремо для більшості поширених платформ. Бібліотека SWT використовує графічні засоби платформи (ОС), що забезпечує швидкість і звичний зовнішній вигляд інтерфейсу користувача.

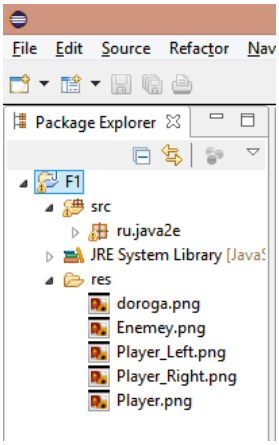
Відповідно до IDC, із Eclipse працюють 2,3 мільйона розробників.[3]

2. Етапи розробки 2D гри

Створимо 2D JAVA гру – гонки. Ми будемо керувати автомобілем. Нашим завданням буде їхати по дорозі так, щоб не спричиняти зіткнень з іншими авто.

*2.1. Eman 1(Створення проекту, підключення папки з зображеннями, створення головного класу **main**, створення головного вікна нашої гри)*

Відкриваємо середовище *Eclipse* . Створюємо новий проект за такою директорією : *File – New – Java Project* . Далі задаємо ім'я проекту: *Project name: F1*. Далі натискаємо клавішу *Next* і в наступному вікні натискаємо *Finish*. **Чудово, тепер ми створили проект F1.**



Утворилась у нас папка *src* – це папка з вихідними файлами. Коли ми будемо створювати якісь класи, то це означає, що ми будемо натискати правою клавішею миші на цій папці *src* і вибиратимемо *New – Class* .У наступному вікні вказуємо назву пакету.

Пакет – це механізм каталогізації. Він нам потрібний для того, щоб всі наші класи вміщувати в одній директорії. [7]

То ж задаємо ім'я нашого пакету. Я обрав ім'я *ru.java2e.*

Далі вказуємо ім'я нашого класу: *Name – Main*. Так як це в нас Main, то ставимо галочку у полі public static void main(String[] args) для того,щоб наша система вже вивела нам готовий шаблон для головного класу main. Далі натискаємо *Finish*.

Чудово! Тепер ми створили клас *main*, який є головним у нашому проекті і з якого розпочинається робота всієї програми.

Тепер створюємо нову папку, в якій будуть зберігатись наші ресурси, тобто зберігатимуться наші зображення дороги, автомобіля, суперників : натискаємо на проекті (F1) правою клавішею миші – *New – Folder* – називаємо її *«res»* (скорочено від *resources*) – *Finish*.

Створили папку з ресурсами і тепер розміщуємо сюди заздалегідь підготовлені зображення для нашої гри. Я їх обробляв у редакторі фотографій **Adobe Photoshop.** Було вирізано потрібне нам зображення, а задній фон було видалено. У папку *res* було розміщено такі зображення :

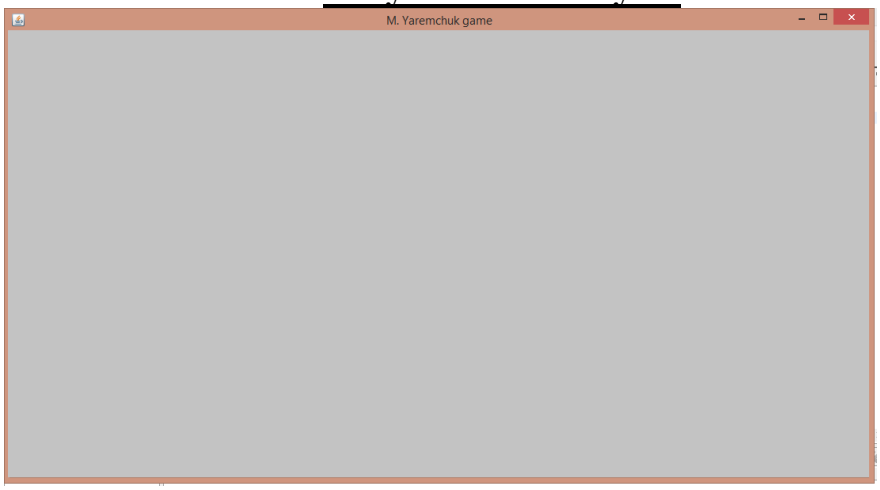
1) *doroga.png* (розміри : ширина 1200 пікселів, висота 600 пікселів) – зображення нашої дороги;
2) *Player.png* (розміри : ширина 160 пікселів, висота 56 пікселів) – зображення нашого автомобіля;
3) *Player_Left.png* (розміри : ширина 158 пікселів, висота 59 пікселів) – зображення автомобіля, коли він рухається вгору нашого вікна (коли повертає ліворуч);
4) *Player_Right.png* (розміри : ширина 158 пікселів, висота 60 пікселів) – зображення автомобіля, коли він рухається вниз нашого вікна (коли повертає праворуч);
5) *Enemey.png* (розміри : ширина 138 пікселів, висота 62 пікселів) – зображення автомобіля нашого суперника.

Тепер створюємо базовий фрейм ,в якому вся наша гра буде відбуватись, тобто створюємо головне вікно.

Main.java

```
package ru.java2e;
import javax.swing.*;  //спочатку імпортуємо бібліотеку

public class Main {

	public static void main(String[] args) {

		JFrame f = new JFrame ("M. Yaremchuk game");  //створюємо наш фрейм і задаємо конструктору ім'я нашого фрейма (нашого вікна).
		f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); //задаємо метод, який встановлює дії, які будуть відбуватись при закритті нашого фрейму (тобто що буде відбуватись, якщо користувач натисне на клавішу X (у правому верхньому куті нашого вікна)). У параметрах задаємо константу(JFrame.EXIT_ON_CLOSE), яка означає, що при натисненні клавіші X буде відбуватись вихід з програми (робимо це,бо за замовчуванням вихід не відбувається)[2]
		f.setSize(1100, 600);     //задаємо розміри вікна за допомогою методу setSize, який приймає ширину і висоту фрейму. Так,як в нас зображення з дорогою має розширення 1200x600, то вказуємо параметри (1100, 600), щоб залишити деяку частину дороги невидимою, бо потім дорога буде рухатись.
		f.setVisible(true);     //тепер робимо наш фрейм видимим,бо за замовчуванням його не видно.[1]
	}}
```

Результат етапу 1:



Ми отримали наший фрейм заданих розмірів з ім'ям «M. Yaremchuk game».

2.2. ***Eman 2****(Створення дороги (клас Road))*

Тепер створюємо щось більше, ніж просто пустий фрейм. Для цього створюємо дорогу.

Створюємо новий клас : натискаємо правою клавішею миші на папці *src* і вибираємо *New – Class – Name – "Road" – Finish* .

Road.java

```
package ru.java2e;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JPanel;

public class Road extends JPanel {  //клас Road розширяє JPanel (тобто дорога в нас займатиме всю область нашого головного фрейму, і ,відповідно, це і буде панель, бо щоб щось помістити на фрейм, треба спочатку його помістити в панель (така особливість))[1]

      Image img = new ImageIcon ("res/doroga.png").getImage();  //(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об'єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення.[2]

      public void paint(Graphics g){  //метод викликається автоматично кожного разу, коли нам потрібно перемалювати нашу панель
            g = (Graphics2D) g;    //виконали приведення типів, бо малювати вміє лише об'єкт Graphics2D
            g.drawImage(img, 0, 0, null);  //тепер малюємо нашу дорогу. Перший параметр – це те,що ми малюємо, 2-й і 3-й – де ми малюємо (координати), 4-й параметр – параметр оновлення зображення (нам його не потрібно, тому null) }}
```

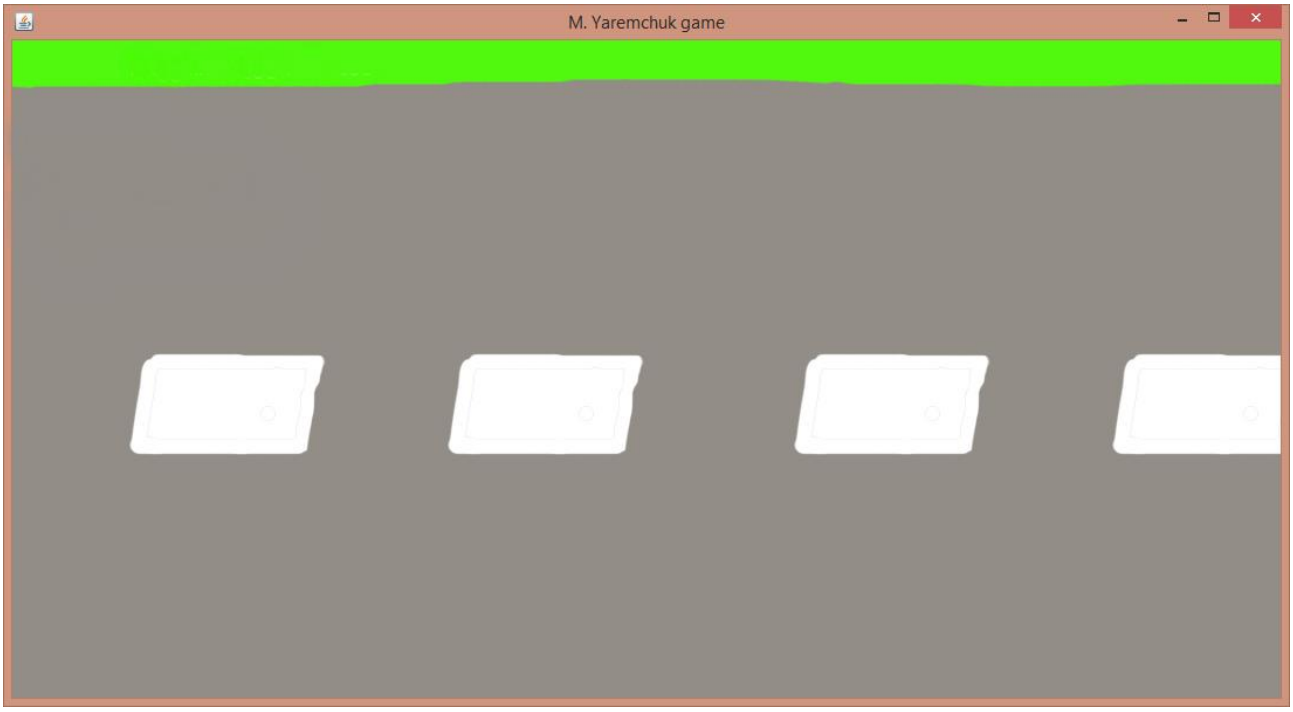
Тепер зберігаємо наший клас *Road* і **переходимо в клас *Main* та додаємо нашу панель**:

Main.java

```
package ru.java2e;

import javax.swing.*;  //спочатку імпортуємо бібліотеку

public class Main {

      public static void main(String[] args) {

            JFrame f = new JFrame ("M. Yaremchuk game");  //створюємо наш фрейм і задаємо конструктору ім'я нашого фрейма (нашого вікна).
            f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); //задаємо метод, який встановлює дії, які будуть відбуватись при закритті нашого фрейму (тобто що буде відбуватись, якщо користувач натисне на клавішу X (у правому верхньому куті нашого вікна)). У параметрах задаємо константу(JFrame.EXIT_ON_CLOSE), яка означає, що при натисненні клавіші X буде відбуватись вихід з програми (робимо це,бо за замовчуванням вихід не відбувається)[2]
            f.setSize(1100, 600);     //задаємо розміри вікна за допомогою методу setSize, який приймає ширину і висоту фрейму. Так,як в нас зображення з дорогою має розширення 1200x600, то вказуємо параметри (1100, 600), щоб залишити деяку частину дороги невидимою, бо потім дорога буде рухатись.
             f.add(new Road());          //додаємо нашу панель
            f.setVisible(true); //тепер робимо наш фрейм видимим,бо за замовчуванням його не видно.
```

```
}}
```

Результат етапу 2:



Наша дорога намалювалась. Етап успішно виконано.

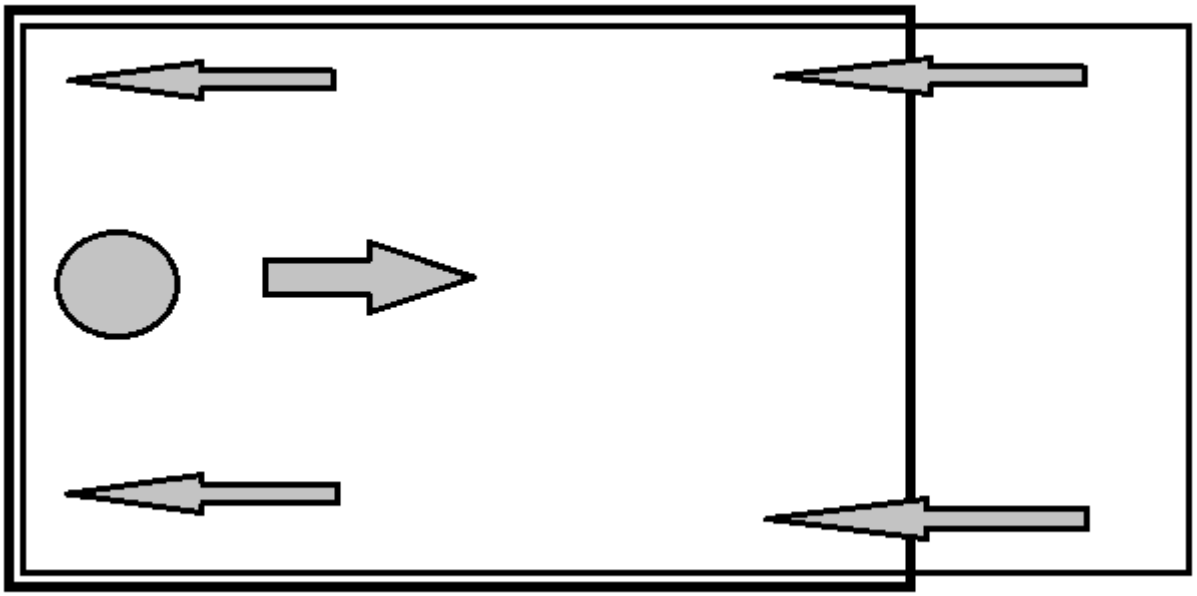
*2.3. **Eman 3**(Створення автомобіля(клас Player))*

Створюємо новий клас : натискаємо правою клавішею миші на папці *src* і вибираємо *New – Class – Name – "Player" – Finish* . Клас *Player* - це машина нашого гравця.

Player.java

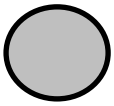
```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;


public class Player {

	Image img = new ImageIcon ("res/Player.png").getImage();  //(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об'єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення.[2]

	int v = 0;    //швидкість автомобіля
	int dv = 0;  //прискорення автомобіля (тобто коли натискаємо на газ – швидкість збільшується на величину dv)
	int s = 0;    //повний наший шлях (абсолютна величина) (тобто нам потрібно контролювати скільки ми проїхали)

	int x = 30;    //координати розміщення нашого автомобіля на дорозі
	int y = 100;

	int layer1 = 0;   //початкова координата нашого 1-го шару (картинки) дороги

	public void move() {  //метод руху (переміщення) нашого автомобіля
		s += v;  //плюсуємо наший шлях (абсолютну величину), який ми накопичуємо, щоб знати скільки ми загалом проїхали
		layer1 -= v;  //зменшуємо координати нашого шару (при переміщенні шлях наший збільшується, а координати шару дороги зменшуються – таким чином ми створюємо ілюзію руху автомобіля)
} }
```

Пояснення анімації нашого зображення:

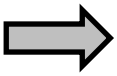


Прямокутник такої лінії демонструє нам головне вікно гри (розширення 1100x600);

Прямокутник такої лінії демонструє наше зображення дороги (розширення 1200x600);



Наший автомобіль;

Ілюзія того, що автомобіль їде  вперед – це те,що вона стоїть на місці, а шар із зображенням дороги рухається  в другу сторону, тобто на зустріч автомобілю.

Звичайно шар закінчиться якщо ми будемо його так пересувати, тому нам потрібно буде ще другий шар дороги, щоб їх чергувати і створити таке, так би мовити, зациклення шарів. Однак для початку, на даному етапі створимо один шар дороги.

Тепер переходимо в клас Road і оголошуємо на дорозі наший автомобіль та редагуємо метод Paint() , малюємо в ньому автомобіль :

Road.java

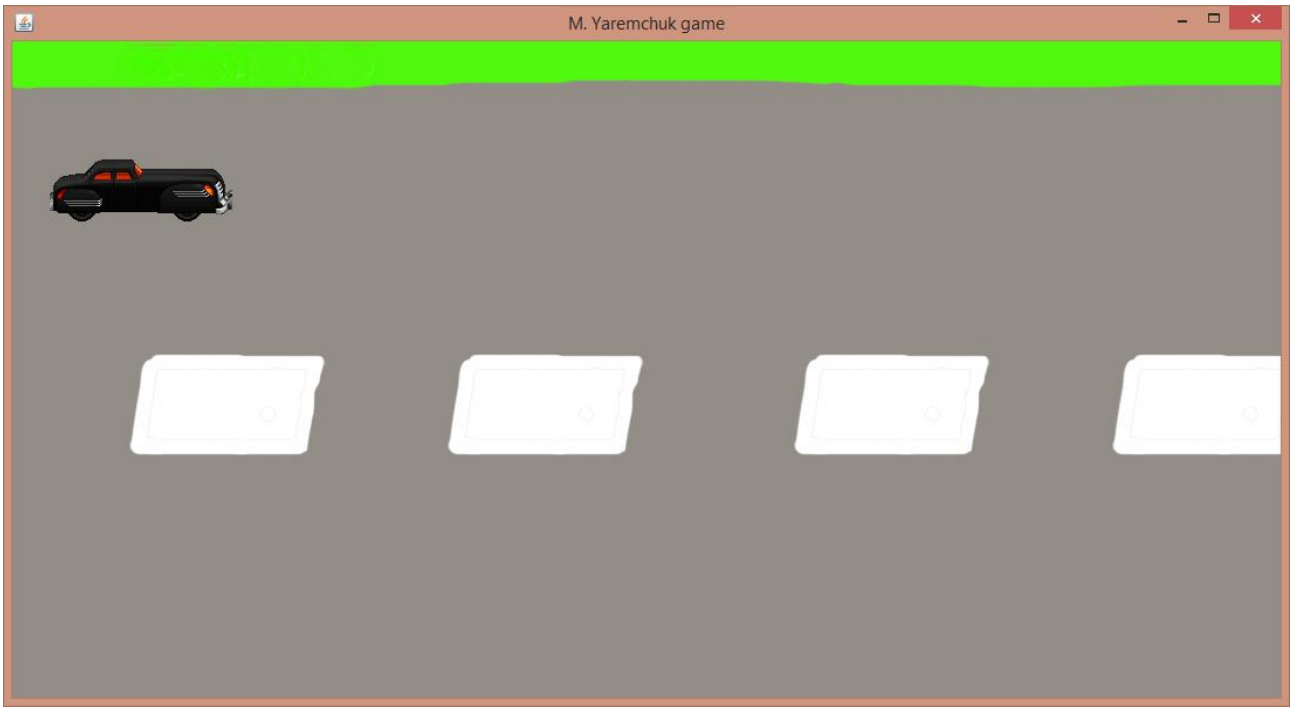
```
package ru.java2e;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JPanel;
```

```
public class Road extends JPanel implements ActionListener{          //клас Road розширяє JPanel
                                                                      //(тобто дорога в нас займатиме всю область нашого головного
                                                                      //фрейму, і, відповідно, це і буде панель, бо щоб щось помістити
                                                                      //на фрейм, треба спочатку його помістити в панель (така особливість))
                                                                      //підключаємо також ActionListener -інтерфейс
                                                                      // функції ActionPerformed для нашого таймера
    Timer mainTimer = new Timer(20, this);          //Створили головний таймер, який буде
                                                    //запускати функцію ActionPerformed
                                                    //кожних 20 мілісекунд в об'єкті this
                                                    //(тобто в  даному об'єкті) [5]
    Image img = new ImageIcon ("res/doroga.png").getImage();          //(Image img – поле img класу
                                                                      //Image). Для того, щоб взяти його з нашого файлу з зображеннями
                                                                      //(папка res), створюємо новий об'єкт класу ImageIcon із шляхом
                                                                      //до нашого файлу і викликаємо метод getImage, який повертає нам зображення.
    Player p = new Player();                        //оголошуємо наший автомобіль

    public Road(){                                  //створюємо конструктор
        mainTimer.start();                          //запускаємо наш таймер
    }
    public void paint(Graphics g){                  //метод викликається автоматично кожного разу,
                                                    //коли нам потрібно перемалювати нашу панель
        g = (Graphics2D) g;                         //виконали приведення типів, бо малювати вміє лише
                                                    //об'єкт Graphics2D
```

Оскільки шар дороги повинен міняти свої координати по x, то замість `g.drawImage(img, 0, 0, null);` **шар буде малюватись по координатах** `g.drawImage(img, p.layer1, 0, null);`**,тому міняємо координату x:**[7]

```
        g.drawImage(img, p.layer1, 0, null);  //тепер малюємо нашу дорогу. Перший
                                              //параметр – це те,що ми малюємо, 2-й і 3-й –
                                              // де ми малюємо (координати), 4-й параметр –
                                              // параметр оновлення зображення (нам його не
                                              //потрібно, тому null)
        g.drawImage(p.img, p.x, p.y, null);       //малюємо наший автомобіль
                                                  //(перший параметр – зображення
                                                  //нашого автомобіля, 2-й і 3-й
                                                  //параметри – координати нашого
                                                  //автомобіля)
                            }
    public void actionPerformed(ActionEvent e){      //кожні 20 мілісекунд
                                                     //наший таймер буде
                                                     //виконувати цю функцію
        p.move();                                //ця функція каже нашому Player їхати

        repaint();                               //також ця функція перемальовуватиме все і
                                                 //викликатиме метод Paint()

        }}
```

Результат етапу 3:



Створився автомобіль, дорога запустилась, анімація дороги працює. Однак ще залишилася проблема – дорога закінчується (тобто з часом малюнок дороги просто пропадає, адже ми ще не зробили зациклення шарів дороги)

*2.4. **Eman 4**(Додавання другого шару дороги та його зациклення з першим шаром дороги + реалізація обробки натиснення на клавіші клавіатури)*

Player.java

```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;


public class Player {

	Image img = new ImageIcon ("res/Player.png").getImage();  //(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об’єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення.[3]

	int v = 50;          //швидкість автомобіля
	int dv = 0;      //прискорення автомобіля (тобто коли натискаємо на газ – швидкість збільшується на величину dv)
	int s = 0;    //повний наший шлях (абсолютна величина) (тобто нам потрібно контролювати скільки ми проїхали)

	int x = 30;   //координати розміщення нашого автомобіля на дорозі
	int y = 100;

	int layer1 = 0;     //початкова координата нашого 1-го шару (картинки) дороги
	int layer2 = 1200;     //малюємо 2-й шар одразу після 1-го, і так, як в нас зображення
	                       //дороги має розширення 1200, то ставимо 1200 і тут.

	public void move() {          //метод руху (переміщення) нашого автомобіля
		s += v;  //плюсуємо наший шлях (абсолютну величину), який ми накопичуємо, щоб знати скільки ми загалом проїхали
		if(layer2 -v <= 0){                //Якщо координати (2-го шару –v) стали
		                                   //менші,або стали дорівнювати нулю
			layer1 = 0;                //тоді повертаємо координати в початкову
			                           //конфігурацію (стан)
			layer2 = 1200;
		} else {                  //в іншому випадку, все робимо як зазвичай

		layer1 -= v;  //зменшуємо координати нашого шару (при переміщенні шлях наший збільшується, а координати шару дороги зменшуються – таким чином ми створюємо ілюзію руху автомобіля)
		layer2 -= v;  //змінюємо наше зображення (тобто рухаємо 2-й шар)
	}}
		public void keyPressed(KeyEvent e){       //метод, який викликається
		                                          //при натисненні клавіші
		                                  //параметр(KeyEvent e) – змінна, яка містить
		                                       // всю інформацію про нашу подію
		JOptionPane.showConfirmDialog (null, “key pressed”);
		}
		public void keyReleased(KeyEvent e){
		JOptionPane.showConfirmDialog (null, “key released”); }}
```

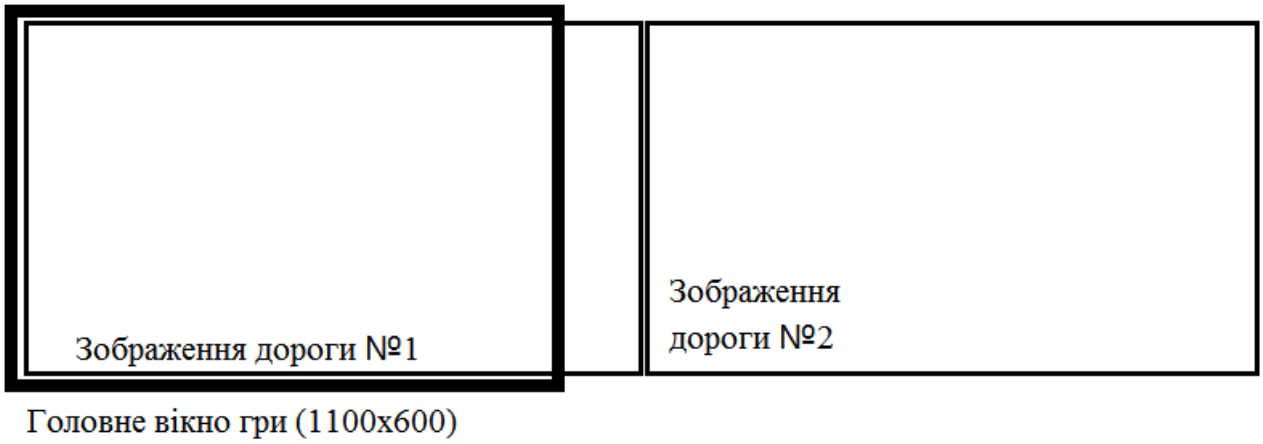
Так, як ми додали 2-й шар дороги і хочемо рухати його, то також нам потрібно його намалювати у методі Paint( ):

Road.java

```
package ru.java2e;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JPanel;

public class Road extends JPanel implements ActionListener{        //клас Road розширяє JPanel
                                                                   //(тобто дорога в нас займатиме всю область нашого головного
                                                                   //фрейму, і, відповідно, це і буде панель, бо щоб щось помістити
                                                                   //на фрейм, треба спочатку його помістити в панель (така особливість))
                                                                   //підключаємо також ActionListener -інтерфейс
                                                                   // функції ActionPerformed для нашого таймера
    Timer mainTimer = new Timer(20, this);      //Створили головний таймер, який буде  запускати функцію ActionPerformed
                                                //кожних 20 мілісекунд в об'єкті this (тобто в  даному об'єкті)
    Image img = new ImageIcon ("res/doroga.png").getImage();       //(Image img – поле img класу
                                                //Image). Для того, щоб взяти його з нашого файлу з зображеннями
                                                //(папка res), створюємо новий об'єкт класу ImageIcon із шляхом
                                                //до нашого файлу і викликаємо метод getImage, який повертає нам зображення.
    Player p = new Player();                    //оголошуємо наший автомобіль

    public Road(){                              //створюємо конструктор
        mainTimer.start();                      //запускаємо наш таймер
        addKeyListener(new MyKeyAdapter());     //реєструємо наший слухач подій
        setFocusable(true);                     //фокусуємось, щоб всі нажимання
                                                //на клавіші оброблялися (без цього
                                                //ніякі кнопки не реагують на наші дії)

    }
    private class MyKeyAdapter extends KeyAdapter{      //використовуємо
                                                        //клас Адаптер, в якому реалізуємо
                                                        //методи для дій з клавішами клавіатури
        public void keyPressed(KeyEvent e){             //коли клавіша натиснена
            p.keyPressed(e);                            //викликаємо у нашого Player
                                                        //метод keyPressed()
                                                        //параметр e – це event(подія)
        }
        public void keyReleased(KeyEvent e){            //коли відпускаємо клавішу
                                                        //(це також потрібно фіксувати)
            p.keyReleased(e);                           //викликаємо у нашого Player  метод
                                                        //keyReleased();  параметр e – це event(подія)
                                    }  }

    public void paint(Graphics g){              //метод викликається автоматично кожного разу,
                                                //коли нам потрібно перемалювати нашу панель
        g = (Graphics2D) g;                     //виконали приведення типів, бо малювати вміє лише
                                                //об'єкт Graphics2D
        g.drawImage(img, p.layer1, 0, null);    //тепер малюємо нашу дорогу. Перший
                                                //параметр – це те,що ми малюємо, 2-й і 3-й –
                                                // де ми малюємо (координати), 4-й параметр –
                                                // параметр оновлення зображення (нам його не
                                                //потрібно, тому null)
        g.drawImage(img, p.layer2, 0, null);    //тепер малюємо 2-й
                                                //шар дороги
    Тобто для 1-го шару буде drawImage (img,0,0,null), а для 2-го -
drawImage (img,1200,0,null) ,(1200 – тому,що ми ініціалізували 2-й шар як
1200).[2]
        g.drawImage(p.img, p.x, p.y, null);     //малюємо наший автомобіль (перший параметр –
```

```
                                                    //зображення нашого автомобіля, 2-й і 3-й
                                                     //параметри – координати нашого автомобіля)
                                }
public void actionPerformed(ActionEvent e){    //кожні 20 мілісекунд наший таймер буде
                                                     //виконувати цю функцію
        p.move();              //ця функція каже нашому Player їхати

        repaint();            //також ця функція перемальовуватиме все і викликатиме метод Paint()
        }}
```

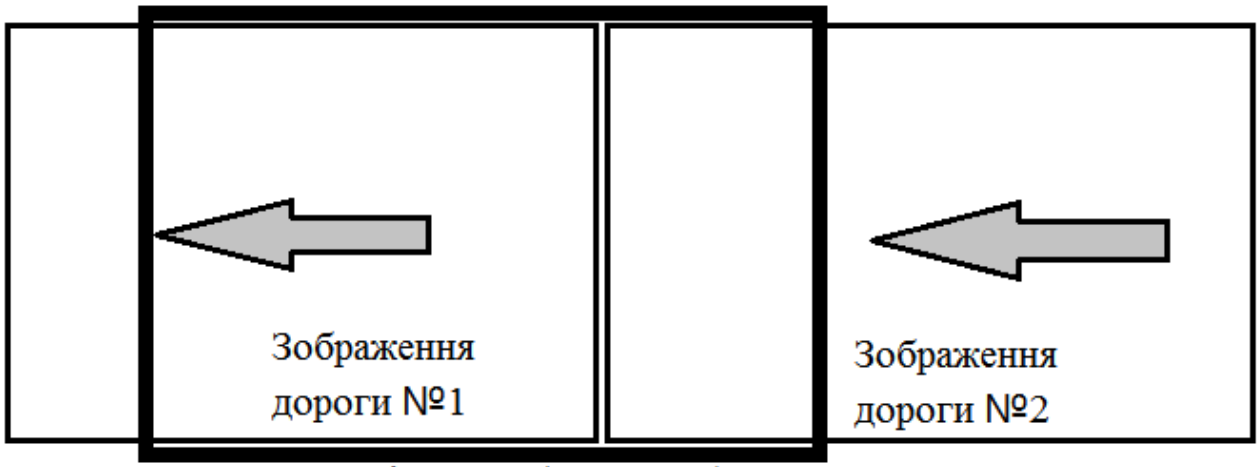
Пояснення зациклення 2-х шарів дороги:



Головне вікно гри (1100x600)

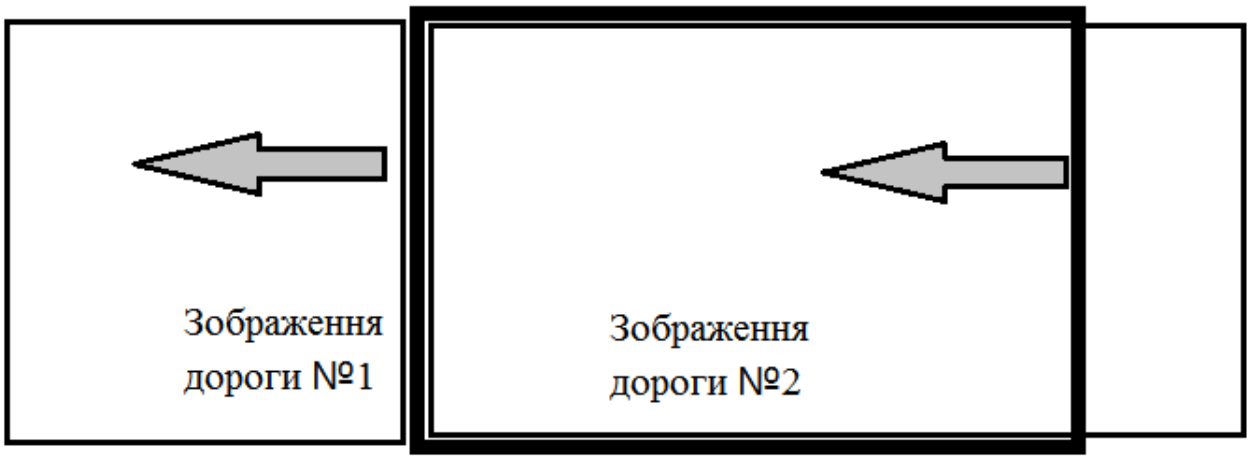
Після першого зображення дороги прямує друге зображення дороги. Тобто в початковий момент часу ми малюємо два зображення, але бачимо лише одне.

В наступний момент часу може виникає ситуація, коли ми бачимо обидва зображення у вікні гри:



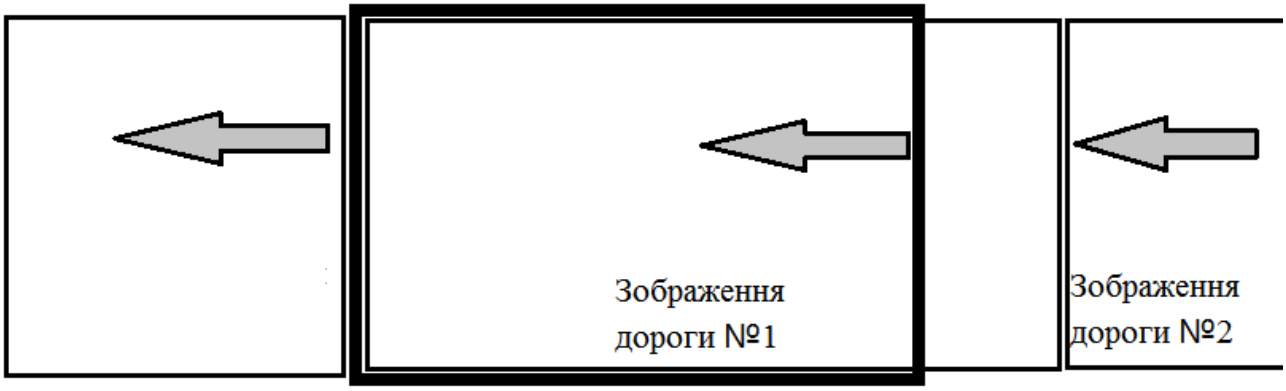
Головне вікно гри (1100x600)

Тепер нам потрібно цей процес зациклити:



Головне вікно гри(1100x600)

В той момент, коли зображення дороги №1 вже вийшло за межі головного вікна нашої гри, координата зображення дороги №2 стало дорівнювати нулю (тобто так як координата зображення дороги №1 дорівнювала на початку). Тобто в даний момент ми поміняємо координати знову на початкові. Ось і все. І в нас тепер зображення,що розміщене в головному вікні на даний момент буде зображенням дороги №1, а наступне за ним зображення буде зображенням дороги №2.



Головне вікно гри(1100x600)

І в нас тепер знову те зображення, що розміщене в даний момент у межах головного вікна , буде зображенням дороги №1 , а наступне зображення буде зображенням дороги №2. І знову,коли наше зображення дороги №1 повністю вийде за межі головного вікна, ми поміняємо його координати і будемо знову його малювати.

Результат етапу 4:

На цьому етапі я додав другий шар дороги і зациклив його з першим шаром дороги , також реалізував натиснення на клавіші. Отже, тепер наша машина їде по наші дорозі безкінечно та прямолінійно з однаковою швидкістю. При натисненні на будь-яку клавішу виводиться на екран повідомлення «key pressed»:

*2.5. **Eman 5**(Організація правильної обробки натиснення клавіш(тобто щоб система розпізнавала яка саме клавіша натиснена (вверх, вниз, вправо, вліво) і, відповідно, щоб мінялась поведінка нашого автомобіля)*

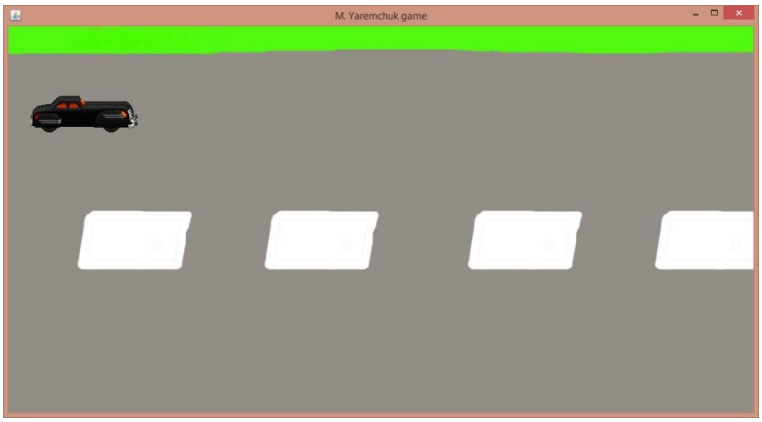
Player.java

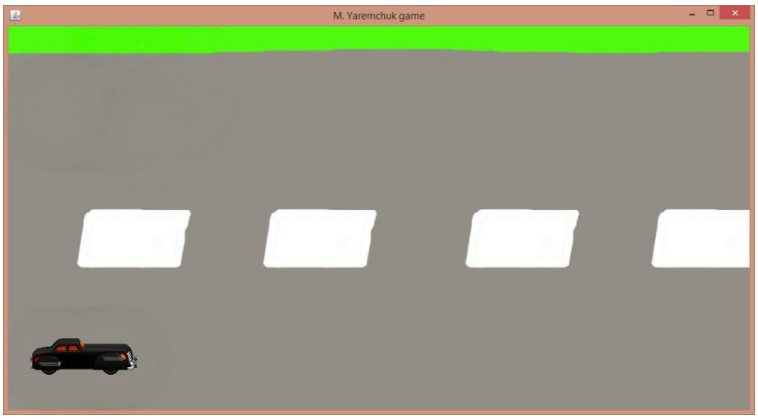
```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;


public class Player {

	public static final int MAX_V = 100;		//константа з максимальною
                                                //швидкістю 100 пікселів за оновлення
	public static final int MAX_TOP = 10;	//максимальна координата по «у»
                                                //нашого авто, вище якої авто не
                                                //підніметься
	public static final int MAX_BOTTOM = 480; //мінімальна координата по «у»
                                                //нашого авто, нижче якої авто не
                                                //опуститься


	Image img = new ImageIcon ("res/Player.png").getImage();  //(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об’єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення.[1]

	int v = 0;	//швидкість автомобіля
	int dv = 0;	//прискорення автомобіля (тобто коли натискаємо на газ – швидкість збільшується на величину dv)
	int s = 0;	//повний наший шлях (абсолютна величина) (тобто нам потрібно контролювати скільки ми проїхали)

	int x = 30;	//координати розміщення нашого автомобіля на дорозі
	int y = 100;
	int dy = 0;			//змінна для зміни координати «у»  нашого автомобіля

	int layer1 = 0;	//початкова координата нашого 1-го шару (картинки) дороги
	int layer2 = 1200;	//малюємо 2-й шар одразу після 1-го, і так, як в нас зображення дороги має розширення 1200, то ставимо 1200 і тут.

	public void move() {	//метод руху (переміщення) нашого автомобіля
		s += v;	//плюсуємо наший шлях (абсолютну величину), який ми накопичуємо, щоб знати скільки ми загалом проїхали
		v += dv;			//плюсуємо швидкість прискорення і дізнаємось
                                        //нову швидкість. (Тобто при натисненні клавіші
                                        //«вправо», набувається значення dv=5 і кожного
                                        //разу в методі move()(тобто кожних 20 мілісекунд)
                                        //(+5) постійно плюсується до швидкості і таким
                                        //чином швидкість постійно набирається) [10]
		if (v <= 0) v = 0;		//якщо ми приплюсували і швидкість стала
                                        //від’ємною, то назад ми не їдемо, а стоїмо на місці
		if (v >= MAX_V) v = MAX_V;		//ставимо максимальну швидкість, щоб
                                                //авто безкінечно не прискорювалось
		y -= dy;			//зміщуємось по координаті «у» (бо вісь
                                        //координат йде зверху вниз)
```

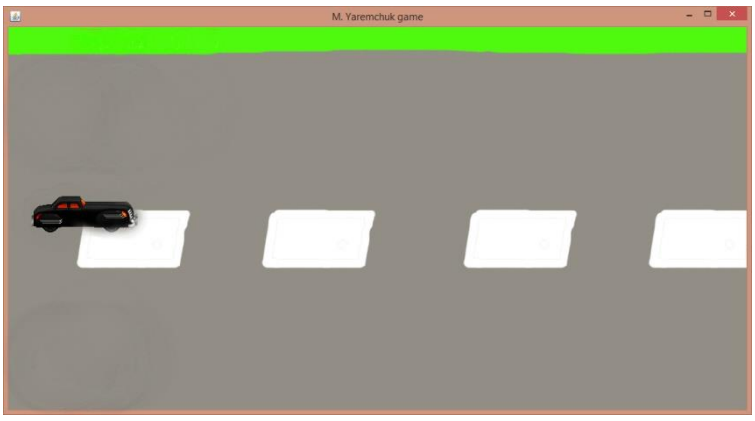
```
            if (y <= MAX_TOP) y = MAX_TOP;          //контролюємо нашу вісь «y»
            if (y >= MAX_BOTTOM) y = MAX_BOTTOM;
            if(layer2 -v <= 0){              //Якщо координати (2-го шару –v) стали
                                                 //менші,або стали дорівнювати нулю
                  layer1 = 0;                //тоді повертаємо координати в початкову
                                                 //конфігурацію (стан)
                  layer2 = 1200;
            } else {   //в іншому випадку, все робимо як зазвичай

            layer1 -= v;  //зменшуємо координати нашого шару (при переміщенні шлях наший
збільшується, а координати шару дороги зменшуються – таким чином ми створюємо ілюзію руху
автомобіля)
            layer2 -= v;  //змінюємо наше зображення (тобто рухаємо 2-й шар)
          }}
            public void keyPressed(KeyEvent e){     //метод, який викликається
                                                        //при натисненні клавіші
                                                //параметр(KeyEvent e) – змінна, яка містить
                                                  // всю інформацію про нашу подію
            int key = e.getKeyCode();          //метод для отримання коду клавіші,
                                                        //яку ми натиснули
            if (key == KeyEvent.VK_RIGHT){    //key – наший код,який ми отримали;
                                              //VK_RIGHT – константа, яка містить
                                              //код клавіші «вправо» (існують такі
                                              //константи, які вже містять в собі
                                              //числові значення клавіш) . Отже,
                                              //якщо key і VK_RIGHT рівні, це
                                            //означає, що ми натиснули клавішу «вправо»
                  dv = 5;                     //при натисненні клавіші «вправо»,
                                              //ми прискорюємось на 5 одиниць
            }
            if (key == KeyEvent.VK_LEFT){     //При натисненні клавіші «вліво», ми
                  dv = -5;                    //тормозимо
            if (key == KeyEvent.VK_UP){       //Якщо натиснена клавіша «вверх», то
                  dy = 15;                    //зміщуємо наше авто по осі «y» на 15
                                                //одиниць вгору
            }
            if (key == KeyEvent.VK_DOWN){     //Якщо натиснена клавіша «вниз», то
                  dy = -15;                   //зміщуємо наше авто по осі «y» на 15
                                              //одиниць вниз
            }}
      public void keyReleased(KeyEvent e){        //коли відпускаємо клавішу
                                                  //(це також потрібно фіксувати)
            int key = e.getKeyCode();           //нам також потрібно отримати код
                                                //клавіші, яка була натиснена
            if (key == KeyEvent.VK_RIGHT || key == KeyEvent.VK_LEFT){
                  dv = 0;                             //якщо була відпущена клавіша
                                                //«вправо», або «вліво», то припиняємо
                                                      //прискорювати автомобіль
            }
            if (key == KeyEvent.VK_UP || key == KeyEvent.VK_DOWN){
                  dy = 0;
                  img = img_c;
            } } }
```

Результат етапу 5:

На цьому етапі я організував правильну обробку натиснення клавіш (тобто тепер система розпізнає яка саме клавіша натиснена (вверх, вниз, вправо, вліво) і, відповідно, міняється поведінка нашого автомобіля (він тепер під нашим керуванням переміщається по екрані). Також я наклав обмеження на рух нашого автомобіля : тепер наший автомобіль може розвивати максимальну швидкість 100 пікселів за оновлення , автомобіль не може їхати назад (інакше це суперечить правилам нашої гри, він повинен їхати вперед) і обмежив максимальне переміщення автомобіля вгору та вниз (щоб автомобіль не виходив за межі дороги).







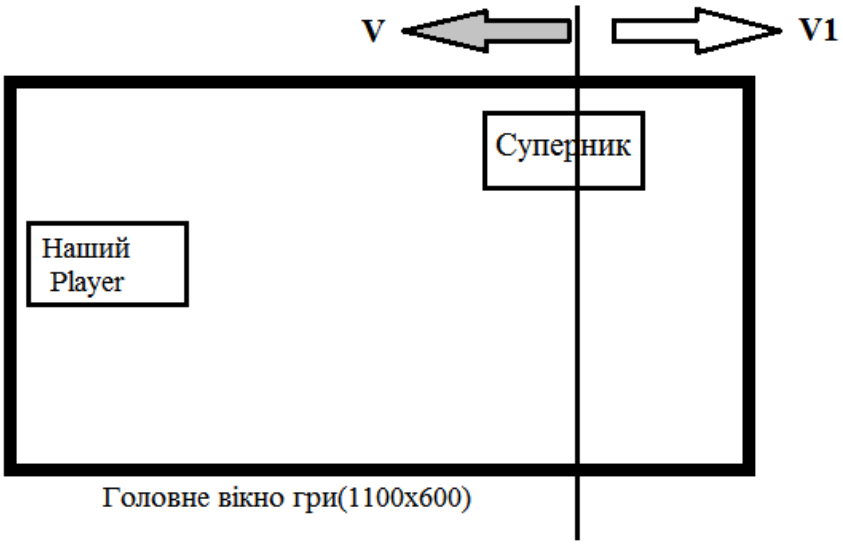
2.6. ***Eman 6***(*Створення суперників*)

Створюємо новий клас : натискаємо правою клавішею миші на папці *src* і вибираємо *New – Class – Name – "Enemy" – Finish* . Клас *Enemy* описуватиме наших суперників.

Enemy.java

```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;
public class Enemy {

	int x;	//координати суперника
	int y;
	int v;	//швидкість суперника
	Image img = new ImageIcon("res/Enemey.png").getImage();	//(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об'єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення.[2]
	Road road;	//нам також необхідне поле «дорога»

	public Rectangle getRect() {		//метод, який повертатиме прямокутник,
						//яким описані наші суперники (метод
						//потрібний, щоб реалізувати зіткнення )
		return new Rectangle(x, y, 138, 62);	//x, y- початкові координати
							//розміщення суперника, 138 – ширина
							//зображення суперника, 62 – висота
							//зображення суперника [4]
	}
	public Enemy(int x, int y, int v, Road road){	//створили конструктор, у
								//параметрах якого x- координата x,
							//y – координата y, v – швидкість, road - дорога
		this.x = x;		//ініціалізуємо
		this.y = y;
		this.v = v;
		this.road = road;
	}

	public void move() {
		x = x - road.p.v + v;	//віднімаємо road.p.v – швидкість шару дороги (це
					//та ж швидкість, з якою рухається наш Player) і
					//додаємо v (власну швидкість суперника)
	}}
```

Пояснення руху суперників:



Вертикальна лінія на нашому зображенні, це відмітка на нашому шарі зображення дороги, для того, щоб показати як він рухається. Шар дороги рухається із швидкістю V (тобто з швидкістю нашого автомобіля). Припустимо, ситуацію коли б наші суперники стояли на місці, то як би змінювалась їхня координата? Було би приблизно так: X = X – V. Тобто вони би статично стояли на дорозі, не рухалися б, і проходили б повз нас. Але нам потрібно, щоб вони їхали, тому нам потрібно ще додати їхній рух в протилежну сторону руху дороги (V1). Тобто будемо мати: X = X – V + V1 (V1 - власна швидкість суперників). Ось так і буде мінятись координата X .

Тепер створюємо наших ворогів. Створювати будемо їх на дорозі:

Road.java

```
package ru.java2e;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JPanel;

public class Road extends JPanel implements ActionListener,Runnable{          //клас Road розширяє JPanel
                                                                              //(тобто дорога в нас займатиме всю область нашого головного
                                                                              //фрейму, і, відповідно, це і буде панель, бо щоб щось помістити
                                                                              //на фрейм, треба спочатку його помістити в панель (така особливість))
                                                                              //підключаємо також ActionListener -інтерфейс
                                                                              // функції ActionPerformed для нашого таймера
                                                                              //додаємо Runnable – це нам потрібно для
                                                                              //генерування ворогів [8]
    Timer mainTimer = new Timer(20, this);      //Створили головний таймер, який буде запускати функцію ActionPerformed
                                                //кожних 20 мілісекунд в об'єкті this (тобто в даному об'єкті)
    Image img = new ImageIcon ("res/doroga.png").getImage();          //(Image img – поле img класу
                                                //Image). Для того, щоб взяти його з нашого файлу з зображеннями
                                                //(папка res), створюємо новий об'єкт класу ImageIcon із шляхом
                                                //до нашого файлу і викликаємо метод getImage, який повертає нам зображення.
    Player p = new Player();                    //оголошуємо наший автомобіль

    Thread enemiesFactory = new Thread(this);   //створюємо поле-потік з
                                                //назвою «фабрика», який буде
```

```
                                                    //працювати паралельно з нашою
                                                  // основною програмою і буде створювати
                                                  //суперників (ми ж хочемо, щоб суперники
                                                  //створювались постійно)
                                              //параметр this означає, що в даному конструкторі
                                            //ми передаємо об'єкт, який реалізує інтерфейс runnable
        List<Enemy> enemies = new ArrayList<Enemy>();      //тут ми створили
                                                                  //колекцію (список), де будуть
                                                                  //зберігатися вороги
        public Road(){                                      //створюємо конструктор
              mainTimer.start();                            //запускаємо наш таймер
              enemiesFactory.start();       //запускаємо потік генерування ворогів
              addKeyListener(new MyKeyAdapter());     //реєструємо наший слухач подій
              setFocusable(true);                           //фокусуємось, щоб всі нажимання
                                                            //на клавіші оброблялися (без цього
                                                            //ніякі кнопки не реагують на наші дії)

        }
        private class MyKeyAdapter extends KeyAdapter{          //використовуємо
                                                                //клас Адаптер, в якому реалізуємо
                                                                //методи для дій з клавішами клавіатури
              public void keyPressed(KeyEvent e){               //коли клавіша натиснена
                    p.keyPressed(e);                            //викликаємо у нашого Player
                                                                //метод keyPressed()
                                                                //параметр e – це event(подія)
              }
              public void keyReleased(KeyEvent e){              //коли відпускаємо клавішу
                                                                //(це також потрібно фіксувати)
                    p.keyReleased(e);                   //викликаємо у нашого Player  метод
                                                        //keyReleased();  параметр e – це event(подія)
                                          }  }

        public void paint(Graphics g){            //метод викликається автоматично кожного разу,
                                                  //коли нам потрібно перемалювати нашу панель
              g = (Graphics2D) g;           //виконали приведення типів, бо малювати вміє лише
                                            //об'єкт Graphics2D
              g.drawImage(img, p.layer1, 0, null);     //тепер малюємо нашу дорогу. Перший
                                                       //параметр – це те,що ми малюємо, 2-й і 3-й –
                                                       // де ми малюємо (координати), 4-й параметр –
                                                       // параметр оновлення зображення (нам його не
                                                       //потрібно, тому null)
              g.drawImage(img, p.layer2, 0, null);            //тепер малюємо 2-й
                                                              //шар дороги
```

Тобто для 1-го шару буде drawImage (img,0,0,null), а для 2-го - drawImage (img,1200,0,null) ,(1200 – тому,що ми ініціалізували 2-й шар як 1200).

```
              g.drawImage(p.img, p.x, p.y, null);  //малюємо наший автомобіль (перший параметр –
                                                   //зображення нашого автомобіля, 2-й і 3-й
                                                   //параметри – координати нашого автомобіля)
```

Щоб paint() малював наших суперників, необхідно пробігтись по колекції і намалювати кожного суперника.

```
Iterator<Enemy> i = enemies.iterator();  //Щоб пробігтись по колекції існує
                                         //Iterator колекції з об'єктами класу Enemy,
                                         //назва його "i", а отримаємо його з нашої
                                         //колекції виводу iterator [6]
```

Тепер рухаємось по всіх елементах

```
              while(i.hasNext()) {              //поки існує наступний елемент (тобто
                                                //поки не досягли кінця)
                    Enemy e = i.next();     //створимо новий екземпляр класу Enemy і
                                            //візьмемо поточний елемент колекції
                    if(e.x >= 2400 || e.x <= -2400){    //якщо координати по осі «x»
                                                        //більші за розміри екрану (коли
```

```
                                                    //суперник виходить за межі екрану)
                        i.remove();           //видаляємо його
                    }else{
                        e.move();             //рухаємо наших суперників
                    g.drawImage(e.img, e.x, e.y, null);        //в іншому випадку
                                                                //малюємо його
                }}}
        public void actionPerformed(ActionEvent e){    //кожні 20 мілісекунд наший таймер буде
                                                        //виконувати цю функцію
                p.move();       //ця функція каже нашому Player їхати
                testCollisionWithEnemies();     //перевіряємо зіткнення з ворогами
                repaint();          //також ця функція перемальовуватиме все і викликатиме метод Paint()
                }

    private void testCollisionWithEnemies() {     //метод перевіряє наявність
                                                    //зіткнення
        Iterator<Enemy> i = enemies.iterator();   //Щоб пробігтись по колекції існує
                                                //Iterator колекції з об'єктами класу Enemy,
                                                //назва його "i", а отримаємо його з нашої
                                                    //колекції виводу iterator
                while (i.hasNext()){                //поки існує наступний елемент (тобто
                                                    //поки не досягли кінця)
                    Enemy e = i.next();       //візьмемо поточний елемент колекції
                    if(p.getRect().intersects(e.getRect())){  //перевіряємо чи
                                                //прямокутник із нашим авто (p.getRect())
                                            //зіштовхнувся із прямокутником ворога (e.gerRect)
                    JOptionPane.showMessageDialog(null, "Ви програли!!!");
                        System.exit(1); //ви програли і виходимо з гри (1 – код виходу)
    }}}
    @Override
    public void run() {                 //run – це точка входу в наший 2-й потік, він буде
                                        //створювати ворогів безкінечно
        while(true) {                       //не постійно, а з випадковою затримкою
            Random rand = new Random();     //створюємо об'єкт класу Random для
                                            //отримання випадкових чисел
            try {                   //тепер будемо затримуватись на випадковий період часу
                Thread.sleep(rand.nextInt(2000));      //дана конструкція
                                                    //дозволяє затримати потік на
                                                //випадкову кількість мілісекунд від 0 до
                                                //2000 (тобто час буде від 0 до 2 сек)
                enemies.add(new Enemy(1200, rand.nextInt(600),
                                                        //створюємо ворогів
                                                //  rand.nextInt(600) – беремо
                                                //випадкову позицію на дорозі по осі «у»
                                                //(від 0 до 600, бо ширина в нас 600)
                            rand.nextInt(60),this));       //rand.nextInt(60) –
                                                //швидкість суперника (макс. шв. = 100,
                                                //тому можемо спокійно взяти 60) буде
                                                //генеруватись від 0 до 60
                                                //this – дорога – це і є наший даний об'єкт
            } catch (InterruptedException e) {  //система автоматично виводить
                        e.printStackTrace();                //такий виняток [6]
    }}}
```

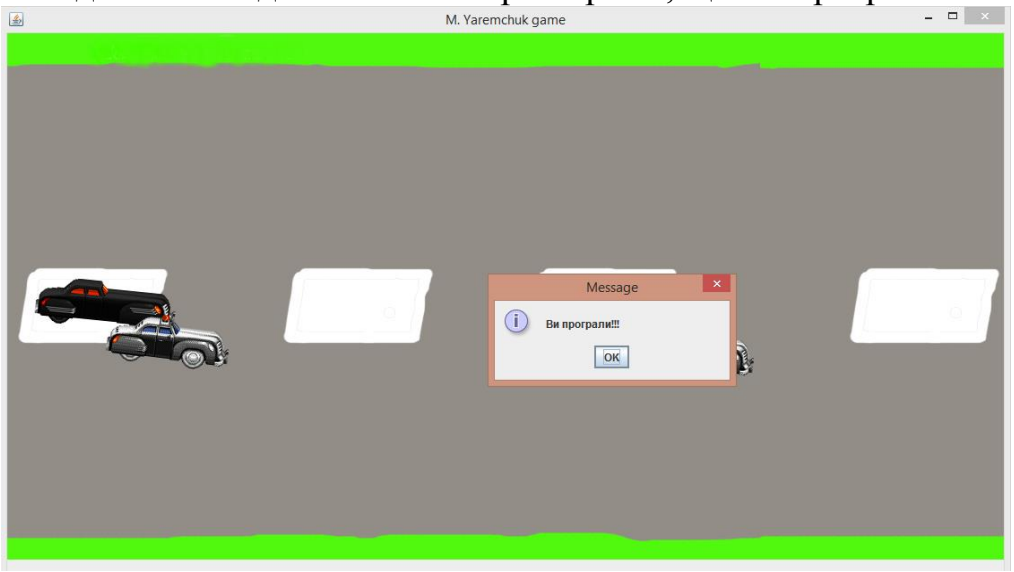
Player.java

```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;


public class Player {

	public static final int MAX_V = 100;                    //константа з максимальною
                                                            //швидкістю 100 пікселів за оновлення
	public static final int MAX_TOP = 10;                   //максимальна координата по «у»
                                                            //нашого авто, вище якої авто не підніметься
	public static final int MAX_BOTTOM = 480;               //мінімальна координата по «у» нашого
                                                            //авто, нижче якої авто не опуститься

	Image img = new ImageIcon ("res/Player.png").getImage();  //(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями (папка res), створюємо новий об'єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод getImage, який повертає нам зображення. [1]

	public Rectangle getRect() {                            //метод, який повертатиме прямокутник,
                                                            //яким описані наші суперники (метод
                                                            //потрібний, щоб реалізувати зіткнення )
		return new Rectangle(x, y, 160, 56);                //x, y- початкові координати
                                                            //розміщення суперника, 160 – ширина
                                                            //зображення нашого авто, 56 – висота
                                                            //зображення нашого авто [6]
	}
	int v = 0;   //швидкість автомобіля
	int dv = 0;  //прискорення автомобіля (тобто коли натискаємо на газ – швидкість збільшується на величину dv)
	int s = 0;   //повний наш шлях (абсолютна величина) (тобто нам потрібно контролювати скільки ми проїхали)

	int x = 30;    //координати розміщення нашого автомобіля на дорозі
	int y = 100;
	int dy = 0;             //змінна для зміни координати «у»  нашого автомобіля

	int layer1 = 0;     //початкова координата нашого 1-го шару (картинки) дороги
	int layer2 = 1200;    //малюємо 2-й шар одразу після 1-го, і так, як в нас зображення дороги має розширення 1200, то ставимо 1200 і тут.

	public void move() {  //метод руху (переміщення) нашого автомобіля
		s += v;  //плюсуємо наший шлях (абсолютну величину), який ми накопичуємо, щоб знати скільки ми загалом проїхали
		v += dv;                                            //плюсуємо швидкість прискорення і дізнаємось
                                                            //нову швидкість. (Тобто при натисненні клавіші
                                                            //«вправо», набувається значення dv=5 і кожного
                                                            //разу в методі move()(тобто кожних 20 мілісекунд)
                                                            //(+5) постійно плюсується до швидкості і таким
                                                            //чином швидкість постійно набирається)
		if (v <= 0) v = 0;                                  //якщо ми приплюсували і швидкість стала
                                                            //від'ємною, то назад ми не їдемо, а стоїмо на місці
		if (v >= MAX_V) v = MAX_V;                          //ставимо максимальну швидкість, щоб
                                                            //авто безкінечно не прискорювалось
		y -= dy;                                            //зміщуємось по координаті «у» (бо вісь
                                                            //координат йде зверху вниз)
		if (y <= MAX_TOP) y = MAX_TOP;                      //контролюємо нашу вісь «у»
		if (y >= MAX_BOTTOM) y = MAX_BOTTOM;
		if(layer2 -v <= 0){                                 //Якщо координати (2-го шару –v) стали
                                                            //менші,або стали дорівнювати нулю
			layer1 = 0;                                     //тоді повертаємо координати в початкову
```

```
                                                    //конфігурацію (стан)
                layer2 = 1200;
            } else {   //в іншому випадку, все робимо як зазвичай

              layer1 -= v;  //зменшуємо координати нашого шару (при переміщенні шлях наший
збільшується, а координати шару дороги зменшуються – таким чином ми створюємо ілюзію руху
автомобіля)
              layer2 -= v;  //змінюємо наше зображення (тобто рухаємо 2-й шар)
           }}
              public void keyPressed(KeyEvent e){      //метод, який викликається
                                                          //при натисненні клавіші
                                                  //параметр(KeyEvent e) – змінна, яка містить
                                                    // всю інформацію про нашу подію
              int key = e.getKeyCode();                  //метод для отримання коду клавіші,
                                                          //яку ми натиснули
              if (key == KeyEvent.VK_RIGHT)               //key – наший код,який ми отримали;
                                                          //VK_RIGHT – константа, яка містить
                                                          //код клавіші «вправо» (існують такі
                                                          //константи, які вже містять в собі
                                                           //числові значення клавіш) . Отже,
                                                           //якщо key і VK_RIGHT рівні, це
                                                           //означає, що ми натиснули клавішу «вправо»
                    dv = 5;                              //при натисненні клавіші «вправо»,
                                                         //ми прискорюємось на 5 одиниць
              }
              if (key == KeyEvent.VK_LEFT){               //При натисненні клавіші «вліво», ми
                    dv = -5;                              //тормозимо
              if (key == KeyEvent.VK_UP){                 //Якщо натиснена клавіша «вверх», то
                    dy = 15;                              //зміщуємо наше авто по осі «у» на 15
                                                          //одиниць вгору
              }
              if (key == KeyEvent.VK_DOWN){                //Якщо натиснена клавіша «вниз», то
                    dy = -15;                             //зміщуємо наше авто по осі «у» на 15
                                                           //одиниць вниз
           }}
       public void keyReleased(KeyEvent e){                  //коли відпускаємо клавішу
                                                             //(це також потрібно фіксувати)
              int key = e.getKeyCode();                       //нам також потрібно отримати код
                                                              //клавіші, яка була натиснена
              if (key == KeyEvent.VK_RIGHT || key == KeyEvent.VK_LEFT){
                    dv = 0;                                    //якщо була відпущена клавіша
                                                               //«вправо», або «вліво», то припиняємо
                                                               //прискорювати автомобіль
              }
              if (key == KeyEvent.VK_UP || key == KeyEvent.VK_DOWN){
                    dy = 0;
                    img = img_c;
           } } }
```

Результат етапу 6:

На цьому етапі я створив суперників, які розміщені на нашій дорозі і рухаються із своєю швидкістю. Всі вони генеруються випадково і швидкість кожного суперника теж генерується випадковою. На даний момент ми можемо керувати нашим автомобілем, об’їзджати суперників і при зіткненні з ними виводиться повідомлення на екран про те, що ми програли.



*2.7. **Eman 7**(Накладення відповідного зображення нашого автомобіля при повороті ліворуч та праворуч, створення спідометра , додавання умови перемоги)*

Player.java

```
package ru.java2e;
import java.awt.Image;
import javax.swing.ImageIcon;


public class Player {

	public static final int MAX_V = 100;                    //константа з максимальною
                                                            //швидкістю 100 пікселів за оновлення
	public static final int MAX_TOP = 10;                   //максимальна координата по «у»
                                                            //нашого авто, вище якої авто не підніметься
	public static final int MAX_BOTTOM = 480;               //мінімальна координата по «у» нашого
                                                            //авто, нижче якої авто не опуститься

	Image img_c = new ImageIcon ("res/Player.png").getImage();

                                                            //центральне (головне ) зображення нашого авто
//(Image img – поле img класу Image). Для того, щоб взяти його з нашого файлу з зображеннями
(папка res), створюємо новий об’єкт класу ImageIcon із шляхом до нашого файлу і викликаємо метод
getImage, який повертає нам зображення.

	Image img_l = new ImageIcon("res/Player_Left.png").getImage();
                                                            //зображення нашого автомобіля
                                                            //при повороті ліворуч та праворуч
	Image img_r = new ImageIcon("res/Player_Right.png").getImage();
	Image img = img_c;    //поточне зображення нашого автомобіля
	public Rectangle getRect() {                            //метод, який повертатиме прямокутник,
                                                            //яким описані наші суперники (метод
                                                            //потрібний, щоб реалізувати зіткнення )
		return new Rectangle(x, y, 160, 56);                //x, y- початкові координати
                                                            //розміщення суперника, 160 – ширина
                                                            //зображення нашого авто, 56 – висота
                                                            //зображення нашого авто
	}
	int v = 0;   //швидкість автомобіля
	int dv = 0;  //прискорення автомобіля (тобто коли натискаємо на газ – швидкість
збільшується на величину dv)
	int s = 0;   //повний наш шлях (абсолютна величина) (тобто нам потрібно контролювати
скільки ми проїхали)

	int x = 30;    //координати розміщення нашого автомобіля на дорозі
	int y = 100;
	int dy = 0;             //змінна для зміни координати «у»  нашого автомобіля

	int layer1 = 0;     //початкова координата нашого 1-го шару (картинки) дороги
	int layer2 = 1200;   //малюємо 2-й шар одразу після 1-го, і так, як в нас зображення
дороги має розширення 1200, то ставимо 1200 і тут.

	public void move() {  //метод руху (переміщення) нашого автомобіля
		s += v;  //плюсуємо наший шлях (абсолютну величину), який ми накопичуємо, щоб
знати скільки ми загалом проїхали
		v += dv;                                            //плюсуємо швидкість прискорення і дізнаємось
                                                            //нову швидкість. (Тобто при натисненні клавіші
                                                            //«вправо», набувається значення dv=5 і кожного
```

```
                                                    //разу в методі move()(тобто кожних 20 мілісекунд)
                                                    //(+5) постійно плюсується до швидкості і таким
                                                    //чином швидкість постійно набирається)
            if (v <= 0) v = 0;                      //якщо ми приплюсували і швидкість стала
                                                    //від'ємною, то назад ми не їдемо, а стоїмо на місці
            if (v >= MAX_V) v = MAX_V;                      //ставимо максимальну швидкість, щоб
                                                            //авто безкінечно не прискорювалось
            y -= dy;                                            //зміщуємось по координаті «y» (бо вісь
                                                                //координат йде зверху вниз)
            if (y <= MAX_TOP) y = MAX_TOP;          //контролюємо нашу вісь «y»
            if (y >= MAX_BOTTOM) y = MAX_BOTTOM;
            if(layer2 -v <= 0){                 //Якщо координати (2-го шару –v) стали
                                                    //менші,або стали дорівнювати нулю
                  layer1 = 0;                   //тоді повертаємо координати в початкову
                                                    //конфігурацію (стан)
                  layer2 = 1200;
            } else {   //в іншому випадку, все робимо як зазвичай

            layer1 -= v;  //зменшуємо координати нашого шару (при переміщенні шлях наший
збільшується, а координати шару дороги зменшуються – таким чином ми створюємо ілюзію руху
автомобіля)
            layer2 -= v;  //змінюємо наше зображення (тобто рухаємо 2-й шар)
          }}
            public void keyPressed(KeyEvent e){     //метод, який викликається
                                                        //при натисненні клавіші
                                                //параметр(KeyEvent e) – змінна, яка містить
                                                  // всю інформацію про нашу подію
            int key = e.getKeyCode();                   //метод для отримання коду клавіші,
                                                        //яку ми натиснули
            if (key == KeyEvent.VK_RIGHT)               //key – наший код,який ми отримали;
                                                        //VK_RIGHT – константа, яка містить
                                                        //код клавіші «вправо» (існують такі
                                                        //константи, які вже містять в собі
                                                        //числові значення клавіш) . Отже,
                                                        //якщо key і VK_RIGHT рівні, це
                                                        //означає, що ми натиснули клавішу «вправо»
                  dv = 5;                               //при натисненні клавіші «вправо»,
                                                        //ми прискорюємось на 5 одиниць
            }
            if (key == KeyEvent.VK_LEFT){               //При натисненні клавіші «вліво», ми
                  dv = -5;                              //тормозимо
            if (key == KeyEvent.VK_UP){                 //Якщо натиснена клавіша «вверх», то
                  dy = 15;                              //зміщуємо наше авто по осі «y» на 15
                                                        //одиниць вгору
                  img = img_l;                  //завантажуємо зображення автомобіля,
                                                //який повертає ліворуч
            }
            if (key == KeyEvent.VK_DOWN){               //Якщо натиснена клавіша «вниз», то
                  dy = -15;                             //зміщуємо наше авто по осі «y» на 15
                                                        //одиниць вниз
                  img = img_r;
          }}
        public void keyReleased(KeyEvent e){            //коли відпускаємо клавішу
                                                        //(це також потрібно фіксувати)
            int key = e.getKeyCode();                   //нам також потрібно отримати код
                                                        //клавіші, яка була натиснена
            if (key == KeyEvent.VK_RIGHT || key == KeyEvent.VK_LEFT){
                  dv = 0;                               //якщо була відпущена клавіша
                                                        //«вправо», або «вліво», то припиняємо
                                                        //прискорювати автомобіль
            }
            if (key == KeyEvent.VK_UP || key == KeyEvent.VK_DOWN){
                  dy = 0;
                  img = img_c;      //завантажуємо центральне зображення автомобіля
          } } }
```

Створюємо тепер спідометр. Йдемо на дорогу, де ми це все і малюватимемо:

Road.java

```
package ru.java2e;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JPanel;

public class Road extends JPanel implements ActionListener,Runnable{          //клас Road розширяє JPanel
                                                                              //(тобто дорога в нас займатиме всю область нашого головного
                                                                              //фрейму, і, відповідно, це і буде панель, бо щоб щось помістити
                                                                              //на фрейм, треба спочатку його помістити в панель (така особливість))
                                                                              //підключаємо також ActionListener -інтерфейс
                                                                              // функції ActionPerformed для нашого таймера
                                                                              //додаємо Runnable – це нам потрібно для генерування ворогів
    Timer mainTimer = new Timer(20, this);      //Створили головний таймер, який буде  запускати функцію ActionPerformed
                                                //кожних 20 мілісекунд в об'єкті this (тобто в  даному об'єкті)
    Image img = new ImageIcon ("res/doroga.png").getImage();          //(Image img – поле img класу
                                                //Image). Для того, щоб взяти його з нашого файлу з зображеннями
                                                //(папка res), створюємо новий об'єкт класу ImageIcon із шляхом
                                                //до нашого файлу і викликаємо метод getImage, який повертає нам зображення.
    Player p = new Player();                    //оголошуємо наший автомобіль

    Thread enemiesFactory = new Thread(this);                 //створюємо поле-потік з
                                                              //назвою «фабрика», який буде
                                                              //працювати паралельно з нашою
                                                              // основною програмою і буде створювати
                                                              //суперників (ми ж хочемо, щоб суперники
                                                              //створювались постійно)
                                                              //параметр this означає, що в даному конструкторі
                                                              //ми передаємо об'єкт, який реалізує інтерфейс runnable
    List<Enemy> enemies = new ArrayList<Enemy>();     //тут ми створили
                                                      //колекцію (список), де будуть
                                                      //зберігатися вороги  [5]
    public Road(){                                    //створюємо конструктор
          mainTimer.start();                          //запускаємо наш таймер
          enemiesFactory.start();                     //запускаємо потік генерування ворогів
          addKeyListener(new MyKeyAdapter());         //реєструємо наший слухач подій
          setFocusable(true);                         //фокусуємось, щоб всі нажимання
                                                      //на клавіші оброблялися (без цього
                                                      //ніякі кнопки не реагують на наші дії)

    }
    private class MyKeyAdapter extends KeyAdapter{          //використовуємо
                                                            //клас Адаптер, в якому реалізуємо
                                                            //методи для дій з клавішами клавіатури
          public void keyPressed(KeyEvent e){               //коли клавіша натиснена
                p.keyPressed(e);                            //викликаємо у нашого Player
                                                            //метод keyPressed()
                                                            //параметр e – це event(подія)

          }
          public void keyReleased(KeyEvent e){              //коли відпускаємо клавішу
                                                            //(це також потрібно фіксувати)
                p.keyReleased(e);                           //викликаємо у нашого Player  метод
                                                            //keyReleased();  параметр e – це event(подія)
                                      }  }

    public void paint(Graphics g){                  //метод викликається автоматично кожного разу,
                                                    //коли нам потрібно перемалювати нашу панель
          g = (Graphics2D) g;                       //виконали приведення типів, бо малювати вміє лише
```

```
                                                    //об’єкт Graphics2D
            g.drawImage(img, p.layer1, 0, null);     //тепер малюємо нашу дорогу. Перший
                                                    //параметр – це те,що ми малюємо, 2-й і 3-й –
                                                    // де ми малюємо (координати), 4-й параметр –
                                                    // параметр оновлення зображення (нам його не
                                                    //потрібно, тому null)
            g.drawImage(img, p.layer2, 0, null);            //тепер малюємо 2-й
                                                            //шар дороги
```

Тобто для 1-го шару буде drawImage (img,0,0,null), а для 2-го - drawImage (img,1200,0,null) ,(1200 – тому,що ми ініціалізували 2-й шар як 1200).

```
            g.drawImage(p.img, p.x, p.y, null);  //малюємо наший автомобіль (перший параметр –
                                                //зображення нашого автомобіля, 2-й і 3-й
                                                 //параметри – координати нашого автомобіля)
            double v = (200/Player.MAX_V) * p.v;   //ділимо максимальну
                                                //швидкість (200км/год) на нашу максимальну
                                                //швидкість 100 (пікселів за обновлення) і
                                                //отримуємо скільки в нас кілометрів за годину
                                            //в одному пікселі і множимо це на кількість пікселів (p.v)
            g.setColor(Color.WHITE);
            Font font = new Font("Arial", Font.ITALIC, 20);
                                                                    //20 – розмір шрифту
            g.setFont(font);                                //встановлюємо наший шрифт
            g.drawString("Швидкість: " + v + " км/год", 100, 30); //100, 30 –
                                                            //розміщення даного запису на екрані
```

Щоб paint() малював наших суперників, необхідно пробігтись по колекції і намалювати кожного суперника.

```
Iterator<Enemy> i = enemies.iterator();             //Щоб пробігтись по колекції існує
                                                    //Iterator колекції з об’єктами класу Enemy,
                                                    //назва його “і”, а отримаємо його з нашої
                                                    //колекції виводу iterator
```

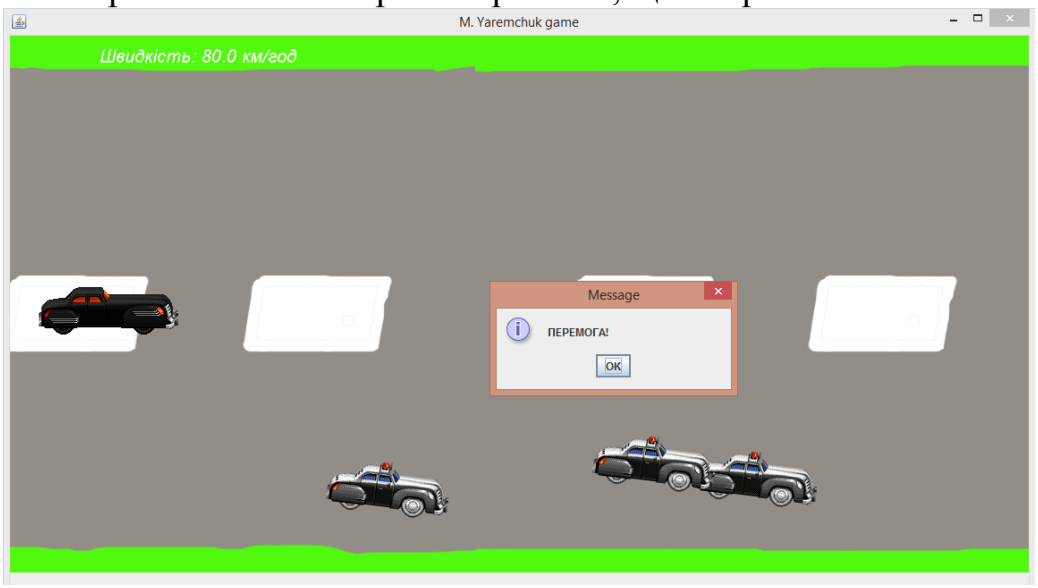
Тепер рухаємось по всіх елементах

```
            while(i.hasNext()) {                            //поки існує наступний елемент (тобто
                                                            //поки не досягли кінця)
                    Enemy e = i.next();                 //створимо новий екземпляр класу Enemy і
                                                        //візьмемо поточний елемент колекції
                    if(e.x >= 2400 || e.x <= -2400){        //якщо координати по осі «х»
                                                            //більші за розміри екрану (коли
                                                            //суперник виходить за межі екрану)
                            i.remove();                     //видаляємо його
                    }else{
                            e.move();                       //рухаємо наших суперників
                    g.drawImage(e.img, e.x, e.y, null);         //в іншому випадку
                                                                //малюємо його
            }}}
      public void actionPerformed(ActionEvent e){    //кожні 20 мілісекунд наший таймер буде
                                                     //виконувати цю функцію
            p.move();       //ця функція каже нашому Player їхати
            repaint();          //також ця функція перемальовуватиме все і викликатиме метод Paint()
            testCollisionWithEnemies();                     //перевіряємо зіткнення з ворогами
            testWin();                          //метод перевіряє, чи ми ще не перемогли
            }

      private void testWin() {
            if (p.s > 20000) {              //загальна відстань,яку вже проїхав Player
                  JOptionPane.showMessageDialog(null, "ПЕРЕМОГА!");
                  System.exit(0);
                                        }
                        }
private void testCollisionWithEnemies() {    //метод перевіряє наявність зіткнення
      Iterator<Enemy> i = enemies.iterator();           //Щоб пробігтись по колекції існує
                                                        //Iterator колекції з об’єктами класу Enemy,
                                                        //назва його “і”, а отримаємо його з нашої
```

```
                                                  //колекції виводу iterator
            while (i.hasNext()){                      //поки існує наступний елемент (тобто
                                                      //поки не досягли кінця)
                  Enemy e = i.next();               //візьмемо поточний елемент колекції
                  if(p.getRect().intersects(e.getRect())){      //перевіряємо чи
                                                  //прямокутник із нашим авто (p.getRect())
                                                  //зіштовхнувся із прямокутником ворога (e.gerRect)
                  JOptionPane.showMessageDialog(null, "Ви програли!!!");
                        System.exit(1);           //ви програли і виходимо з гри (1 – код виходу)
}}}
@Override
public void run() {                               //run – це точка входу в наший 2-й потік, він буде
                                                  //створювати ворогів безкінечно
      while(true) {                               //не постійно, а з випадковою затримкою
            Random rand = new Random();           //створюємо об’єкт класу Random для
                                                  //отримання випадкових чисел
            try {                           //тепер будемо затримуватись на випадковий період часу
                  Thread.sleep(rand.nextInt(2000));       //дана конструкція
                                                  //дозволяє затримати потік на
                                                  //випадкову кількість мілісекунд від 0 до
                                                  //2000 (тобто час буде від 0 до 2 сек)
                  enemies.add(new Enemy(1200, rand.nextInt(600),
                                                  //створюємо ворогів
                                                  //  rand.nextInt(600) – беремо
                                                  //випадкову позицію на дорозі по осі «у»
                                                  //(від 0 до 600, бо ширина в нас 600)
                              rand.nextInt(60),this));      //rand.nextInt(60) –
                                                  //швидкість суперника (макс. шв. = 100,
                                                  //тому можемо спокійно взяти 60) буде
                                                  //генеруватись від 0 до 60
                                                  //this – дорога – це і є наший даний об’єкт
            } catch (InterruptedException e) {          //система автоматично виводить
                        e.printStackTrace();          //такий виняток
}}}
```

Результат етапу 7:

На цьому етапі я встановив відповідні зображення нашого автомобіля при повороті ліворуч та праворуч, створив спідометр , додав умови перемоги (тобто який проміжок нам потрібно проїхати, щоб перемогти.



Додав зображення нашого автомобіля при повороті ліворуч:

При повороті праворуч:

*2.8. **Етап 8**(Збереження нашої гри, збірка проекту)*

Тепер потрібно побудувати наший додаток так, щоб його можна було запустити, не компілюючи, не відкриваючи його в середовищі, а просто двома натисненнями на ярлику можна було б перенести на будь-якому комп’ютері.

Збірка проекту: правою кнопкою на нашому проекті – Export – JAR file – Next – вказуємо шлях збереження – Next – Next – Finish. Все, наша гра повністю готова. Успішно побудована та збережена. Успішно запускається. [8]

Висновок:

На завершення, для того, щоб остаточно відповісти на питання «Що таке JAVA?», торкнемося такої важливої сфери JAVA-програмування, як програмування JAVA-ігор. Ви можете спостерігати ці ігри у ваших мобільних телефонах. Мова JAVA найкраще підходить для створення таких продуктів.

Створення 2D ігор є надзвичайно корисним сьогодні. Вони можуть бути різних жанрів, залежно від вподобань користувачів. Одні ігри заставлять нас логічно мислити, інші вимагають швидкої реакції, деякі – необхідність прийняття рішень, однак є й такі, які допомагають нам розслабитись, цікаво провести час після важкого робочого дня, чи інших труднощів у житті, тобто створені для розваги та відпочинку. Саме до такого жанру (розваги) належить моя гра.

Я вважаю, що в Україні потрібно більше розвивати створення 2D ігор, адже ця галузь позитивно впливає на суспільство (розвиток,самовдосконалення, навчання, розвага) і не викликає залежності. Створення таких проектів не вимагає великих капіталовкладень і довготривалих термінів розробки. Саме завдяки розвитку даної сфери, Україна зможе наповнити свій бюджет, оскільки такі ігри дорого не коштують і їх може придбати кожен бажаючий, незалежно від матеріального становища та віку. Отже, потрібно більше заохочувати молодь розробляти 2D ігри, більше давати їм можливостей у розвитку цієї сфери.

У першому розділі курсової роботи були вивчені основні засоби, що використовуються для розробки ігор, зокрема, мова програмування високого рівня Java і середовище розробки Eclipse.

У другій частині курсової роботи описана поетапна розробка моєї гри з висновками про зроблені дії на кожному етапі.

Усі результати роботи протестовано на реальних пристроях з даною ОС та отримано задовільний результат роботи.

Результатом моєї курсової роботи є готова Java-гра, у якій Ви керуєте автомобілем і головне завдання полягає в тому, щоб проїхати по дорозі певний проміжок, не спричинивши зіткнення з іншими автомобілями.

Список використаної літератури:

1. Bruce Eckel thinking in java 4th edition

2. Matt Weisfeld the object-oriented thought process

3. Oracle Official Website

4. Крис Джамса Библиотека программиста Java .- Jamsa Press, 1996, ООО "Попурри", 1996

5. Нейл Бартлетт, Алекс Лесли, Стив Симкин Программирование на Java. Путеводитель .- The Coriolis Group,Inc.,1996, Издательство НИПФ "ДиаСофт Лтд.",1996

6. Кен Арнольд, Джеймс Гослинг Язык программирования Java .- Addison-Wesley Longman,U.S.A.,1996, Издательство "Питер-Пресс", 1997

7. Java Was Strongly Influenced by Objective-C

8. Нотон П. JAVA:Справ.руководство :Пер.с англ./Под ред.А.Тихонова.-М.:БИНОМ:Восточ.Кн.Компания,1996:Восточ.Кн.Компания.-447с..-(Club Computer)

9. Патрик Нотон, Герберт Шилдт Полный справочник по Java .- McGraw-Hill,1997, Издательство "Диалектика",1997

10. Дэвид Флэнэген Java in a Nutshell .- O'Reilly & Associates, Inc., 1997, Издательская группа BHV, Киев, 1998

Додаток А. Код програми

Main.java

```
package ru.java2e;

import javax.swing.*;

public class Main {

	public static void main(String[] args) {

		JFrame f = new JFrame ("M. Yaremchuk game");
		f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		f.setSize(1100, 600);
		f.add(new Road());
		f.setVisible(true);
	}
}
```

Road.java

```
package ru.java2e;

import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Random;

import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.Timer;
public class Road extends JPanel implements ActionListener, Runnable {
	Timer mainTimer = new Timer(20, this);
	Image img = new ImageIcon(getClass()
			.getClassLoader().getResource("res/doroga.png")).getImage();

	Player p = new Player();
	Thread enemiesFactory = new Thread(this);
	List<Enemy> enemies = new ArrayList<Enemy>();
	public Road(){
		mainTimer.start();
		enemiesFactory.start();
		addKeyListener(new MyKeyAdapter());
		setFocusable(true);
```

```
}
private class MyKeyAdapter extends KeyAdapter{
	public void keyPressed(KeyEvent e){
		p.keyPressed(e);
	}
	public void keyReleased(KeyEvent e){
		p.keyReleased(e);
	    }
    }

public void paint(Graphics g){
	g = (Graphics2D) g;
	g.drawImage(img, p.layer1, 0, null);
	g.drawImage(img, p.layer2, 0, null);
	g.drawImage(p.img, p.x, p.y, null);

	double v = (200/Player.MAX_V) * p.v;
	g.setColor(Color.WHITE);
	Font font = new Font("Arial", Font.ITALIC, 20);
	g.setFont(font);
	g.drawString("Швидкість: " + v + " км/год", 100, 30);

	Iterator<Enemy> i = enemies.iterator();
	while(i.hasNext()) {
		Enemy e = i.next();
		if(e.x >= 2400 || e.x <= -2400){
			i.remove();
		}else{
			e.move();
		g.drawImage(e.img, e.x, e.y, null);
	}
	 }
      }
public void actionPerformed(ActionEvent e){
	p.move();
	repaint();
	testCollisionWithEnemies();
	testWin();
}
private void testWin() {
	if (p.s > 20000) {
		JOptionPane.showMessageDialog(null, "ПЕРЕМОГА!");
		System.exit(0);
	    }
     }

private void testCollisionWithEnemies() {
	Iterator<Enemy> i = enemies.iterator();
	while (i.hasNext()){
		Enemy e = i.next();
		if(p.getRect().intersects(e.getRect())){
			JOptionPane.showMessageDialog(null, "Ви програли!!!");
			System.exit(1);
		    }
```

```
                }
            }
        @Override
        public void run() {
            while(true) {
                Random rand = new Random();
                try {
                    Thread.sleep(rand.nextInt(2000));
                    enemies.add(new Enemy(1200, rand.nextInt(600),
                            rand.nextInt(60),this));
                } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
            }
        }
```

Player.java

```
package ru.java2e;

import java.awt.Image;
import java.awt.Rectangle;
import java.awt.event.KeyEvent;

import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

public class Player {

    public static final int MAX_V = 100;
    public static final int MAX_TOP = 10;
    public static final int MAX_BOTTOM = 480;

    Image img_c = new ImageIcon(getClass()
            .getClassLoader().getResource("res/Player.png")).getImage();
    Image img_l = new ImageIcon(getClass()
            .getClassLoader().getResource("res/Player_Left.png")).getImage();
    Image img_r = new ImageIcon(getClass()
            .getClassLoader().getResource("res/Player_Right.png")).getImage();

    Image img = img_c;

    public Rectangle getRect() {
        return new Rectangle(x, y, 160, 56);
    }

    int v = 0;
    int dv = 0;
    int s = 0;

    int x = 30;
    int y = 100;
    int dy = 0;
```

```
int layer1 = 0;
int layer2 = 1200;

public void move() {
        s += v;
        v += dv;
        if (v <= 0) v = 0;
        if (v >= MAX_V) v = MAX_V;
        y -= dy;
        if (y <= MAX_TOP) y = MAX_TOP;
        if (y >= MAX_BOTTOM) y = MAX_BOTTOM;
        if(layer2 -v <= 0){
                layer1 = 0;
                layer2 = 1200;
        } else {
        layer1 -= v;
        layer2 -= v;
        }
  }

public void keyPressed(KeyEvent e){
        int key = e.getKeyCode();
        if (key == KeyEvent.VK_RIGHT){
                dv = 5;
        }
        if (key == KeyEvent.VK_LEFT){
                dv = -5;
        }
        if (key == KeyEvent.VK_UP){
                dy = 15;
                img = img_l;
        }
        if (key == KeyEvent.VK_DOWN){
                dy = -15;
                img = img_r;
             }
        }
public void keyReleased(KeyEvent e){
        int key = e.getKeyCode();
        if (key == KeyEvent.VK_RIGHT || key == KeyEvent.VK_LEFT){
                dv = 0;
        }
        if (key == KeyEvent.VK_UP || key == KeyEvent.VK_DOWN){
                dy = 0;
                img = img_c;
            }
          }
  }
```

Enemy.java

```
package ru.java2e;

import java.awt.Image;
import java.awt.Rectangle;
```

```
import javax.swing.ImageIcon;

public class Enemy {

	int x;
	int y;
	int v;
	Image img = new ImageIcon(getClass()
				.getClassLoader().getResource("res/Enemey.png")).getImage();
	Road road;

	public Rectangle getRect() {
		return new Rectangle(x, y, 138, 62);
	}

	public Enemy(int x, int y, int v, Road road){
		this.x = x;
		this.y = y;
		this.v = v;
		this.road = road;
	}

	public void move() {
		x = x - road.p.v + v;
	}
}
```