INSTITUTO SUPERIOR POLITECNICO DE TECNOLOGIA E CIÊNCIA

DEPARTAMENTO DE ENGENHARIA

ENGENHARIA INFORMATICA

COMPUTAÇÃO GRÁFICA

António Kipanda Cardoso

Edivaldo Maciel de Almeida Andrade

Edmar Cardoso

Fanuel Henriques de Oliveira Quimbango

**ACTIVIDADE DE TRANSLAÇÃO COM O PROCESSING**

LUANDA

2020

RESUMO

Este projecto tem como objectivo criar uma animação baseada em translação para desvio e intersecção de coordenadas no processing, usando técnicas aprendidas na disciplina de Computação Gráfica.

Em uma via, o objeto principal deve desviar dos obstáculos que devem estar no centro, cima ou baixo.

Primeiramente será apresentada uma breve introdução, à seguir, os conceitos do processing e sobre transformações geométricas. No capítulo seguinte é apresentado a implementação do projecto e logo em seguida, a secção dos testes realizados.

**Palavras-chave:** Desenvolvimento, Processing, Computação Gráfica, Translações, Implementação.

SUMÁRIO

[1. CONCEITOS GERAIS 1](#_Toc56034054)

[1.1. PROCESSING 1](#_Toc56034055)

[1.1.1. CARACTERISTICAS 1](#_Toc56034056)

[1.1.2. PORQUÊ O PROCESSING? 1](#_Toc56034057)

[1.2. TRANSFORMAÇÕES LINEARES 2](#_Toc56034058)

[1.2.1. MATRIZES EM COMPUTAÇÃO GRÁFICA 2](#_Toc56034059)

[1.2.2. TRANSFORMAÇÕES EM PONTOS E OBJETOS 3](#_Toc56034060)

[1.2.2.1. TRANSFORMAÇÃO DE TRANSLAÇÃO 3](#_Toc56034061)

[1.3. TRANSLAÇÃO NO PROCESSING 4](#_Toc56034062)

[2. IMPLEMENTAÇÃO 5](#_Toc56034063)

[2.1. DECLARAÇÃO DE VARIÁVEIS GLOBAIS 5](#_Toc56034064)

[2.2. INICIALIZAÇÃO 6](#_Toc56034065)

[2.3. MÉTODO GAME 6](#_Toc56034066)

[2.4. MÉTODO PRINCIPAL (DRAW) 9](#_Toc56034067)

[2.5. MÉTODOS SECUNDÁRIOS 9](#_Toc56034068)

[3. TESTES E RESULTADOS 10](#_Toc56034069)

[REFERÊNCIAS 12](#_Toc56034070)

SUMÁRIO DE QUADROS E FIGURAS

Figura 1: Translação de um triângulo de três unidades...3

Figura 2: Variáveis globais de configuração do jogo.5

Figura 3: Variáveis globais de posicionamento e estado dos objectos.6

Figura 4: Inicialização do jogo.6

Figura 5: Transladando e desenhando o conjunto de objectos do jogo.7

Figura 6: Transladando e desenhando o jogador.7

Figura 7: Verificando as possíveis colisões do jogador com os objectos.8

Figura 8: Somando a pontuação e actualizando a velocidade.8

Figura 9: Tela inicial do jogo.10

Figura 10: Jogo decorrendo.10

Figura 11: Colisão com um obstáculo.11

Figura 12: Vitória.11

# CONCEITOS GERAIS

## PROCESSING

Processing é uma linguagem de programação de código aberto e ambiente de desenvolvimento integrado (IDE), construído para as artes eletrônicas e comunidades de projetos visuais com o objetivo de ensinar noções básicas de programação de computador em um contexto visual e para servir como base para cadernos eletrônicos. O projeto foi iniciado em 2001 por Casey Reas e Ben Fry, ambos ex-membros do Grupo de Computação do MIT Media Lab.

Um dos objetivos do Processing é atuar como uma ferramenta para não-programadores iniciados com a programação, através da satisfação imediata com um retorno visual. A linguagem tem por base as capacidades gráficas da linguagem de programação Java, simplificando características e criar alguns novos.

### CARACTERISTICAS

Processing é considerado um sketchbook, uma alternativa de organização de projetos sem ser o um IDE padrão.

Cada esboço (sketch) de Processing é realmente uma subclasse do Java PApplet classe que implementa a maioria das funcionalidades da Linguagem Processing.

Ao programar em Processing, todas classes adicionais definidas serão tratados como classes internas quando o código é traduzido para Java puro antes de compilar. Isso significa que o uso de variáveis e métodos estáticos em classes é proibido a menos que você diga que deseja o processamento para o código no modo de Java puro.

### PORQUÊ O PROCESSING?

* Open Source
* Versão Beta (ainda) o que o torna ideal
* Multi-plataforma: MacOS, Win, Unix.
* Poder do JAVA
  + Simplicidade
  + Adequa-se ao processo de ensino\*
  + Faz uma boa transição entre linguagens de baixo nível e scripting de muito alto nível;
  + Documentação extensa.
  + Há diversos websites dedicados à programação, especialmente em JAVA e muitos dedicados ao Processing – ver a secção de links;
  + Extensível
* Exporta Executáveis e Applets para a Web
* Comunidade

## TRANSFORMAÇÕES LINEARES

Transformações geométricas são operações que podem ser utilizadas visando a alteração de algumas características como posição, orientação, forma ou tamanho do objeto a ser desenhado.

### MATRIZES EM COMPUTAÇÃO GRÁFICA

Todas as transformações geométricas podem ser representadas na forma de equações. O problema é que manipulações de objetos gráficos normalmente envolvem muitas operações de aritmética simples. As matrizes são muito usadas nessas manipulações porque são mais fáceis de usar e entender do que as equações algébricas, o que explica por que programadores e engenheiros as usam extensivamente.

As matrizes são parecidas com o modelo organizacional da memória dos computadores. Suas representações se relacionam diretamente com estas estruturas de armazenamento, facilitando o trabalho dos programadores e permitindo maior velocidade para aplicações críticas como jogos e aplicações em realidade virtual. É devido a esse fato que os computadores com “facilidades vetoriais” têm sido muito usados junto a aplicações de computação gráfica.

Devido ao padrão de coordenadas usualmente adotado para representação de pontos no plano (x,y) e no espaço tridimensional (x,y,z), pode ser conveniente manipular esses pontos em matrizes quadradas de 2×2 ou 3×3 elementos. Através de matrizes e de sua multiplicação, podemos representar todas as transformações lineares 2D e 3D. Várias transformações podem ser combinadas resultando em uma única matriz denominada matriz de transformação.

### TRANSFORMAÇÕES EM PONTOS E OBJETOS

A habilidade de representar um objeto em várias posições no espaço é fundamental para compreender sua forma. A possibilidade de submetê-lo a diversas transformações é importante em diversas aplicações da computação gráfica [Rogers, 1990]. As operações lineares de rotação e translação de objetos são chamadas operações de corpos rígidos.

Existem diversos tipos de transformações, mas, verémos apenas a transformação de translação, por ser a actividade principal do projecto.

### TRANSFORMAÇÃO DE TRANSLAÇÃO

Transladar significa movimentar o objeto. Transladamos um objeto transladando todos os seus pontos, como mostrado na **Figura 1**. É possível efetuar a translação de pontos no plano (x,y) adicionando quantidades às suas coordenadas. Assim, cada ponto em (x,y) pode ser movido por Tx unidades em relação ao eixo x, e por Ty unidades em relação ao eixo y. Logo, a nova posição do ponto (x,y) passa a ser(x’,y’), que pode ser escrito como:

x’ = x + Tx

y’ = y + Ty

Repare que, se o ponto for representado na forma de um vetor, P=(x,y), a translação de um ponto pode ser obtida pela adição de um vetor de deslocamento à posição atual do ponto: P’ = P + T = [x’ y’] = [x y] + [Tx Ty ].

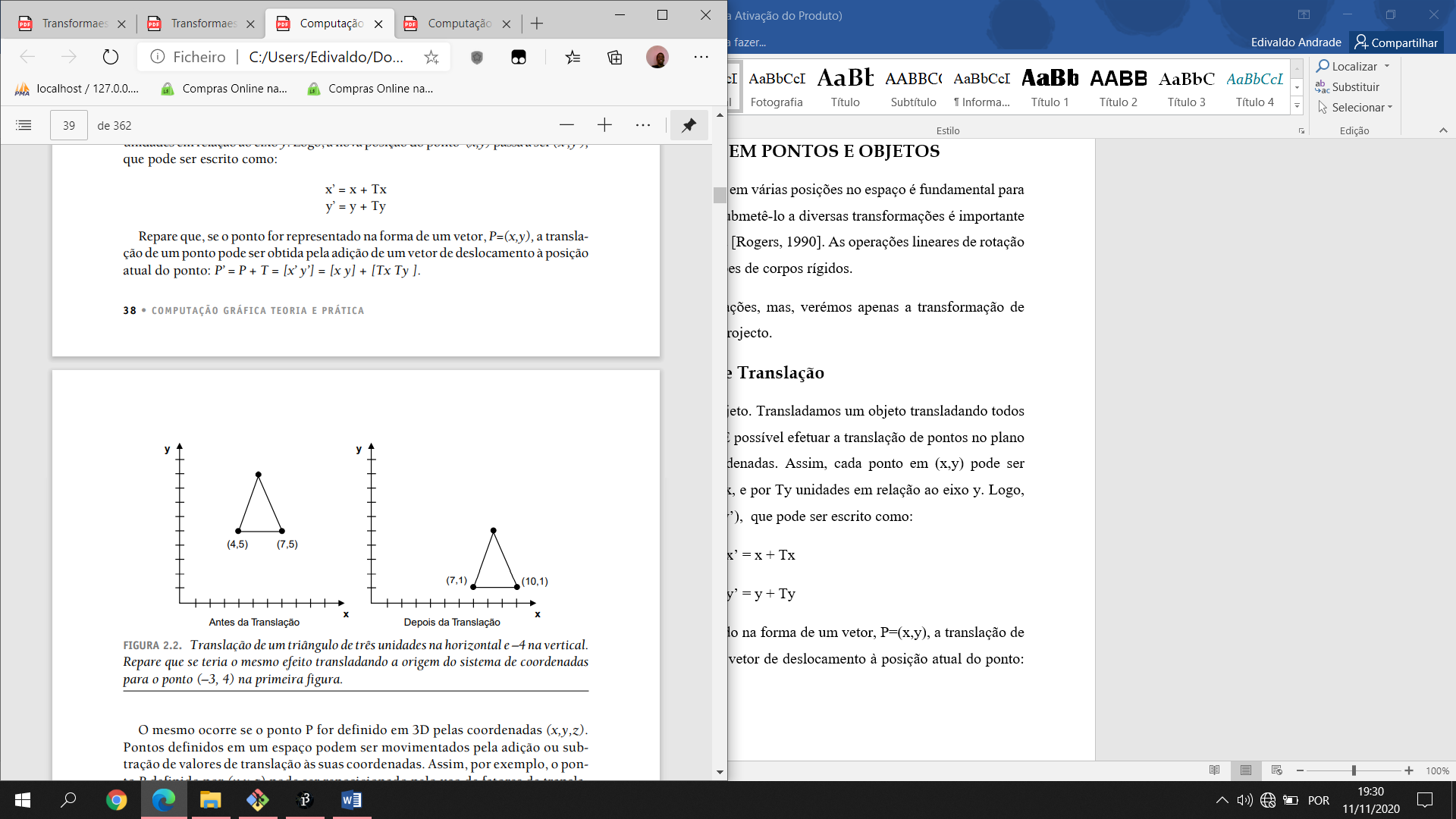


Figura : Translação de um triângulo de três unidades na horizontal e –4 na vertical.

## TRANSLAÇÃO NO PROCESSING

O comando para a translação é translate(TYPE x, TYPE y) e tem como parâmetros as distâncias em cada um dos eixos coordenados. Por exemplo, para mover o ponto de visão ao longo do eixo x de 5 unidades, usa-se:

* translate(5.0, 0.0);

A sequência de comandos para mover uma janela ou objeto, é dada pelas linhas:

* // Definindo o vector object com duas posições no escopo global, a primeira posição representando o eixo x, e a segunda posição o y
* float object[] = {0, 0};
* // Os comandos abaixo podem ser executados na função draw, por causa do loop
* Object[0] = object[0] + 2;
* Object[1] = object[1] + 1;
* translate(object[0], object[1]);
* desenhaObjeto(); // Desenha o objeto deslocado de 2 unidades no eixo x e 1 no eixo y

# IMPLEMENTAÇÃO

Para a implementação prática do jogo foram utilizadas as seguintes estruturas:

## DECLARAÇÃO DE VARIÁVEIS GLOBAIS

Foram definidas um conjunto de variáveis globais, de modo a obter uma simples manipulação das mesmas.

* + **screen\_total –** Tamanho total do ecrã;
  + **screen\_**game – Posição inicial da janela do jogo;
  + **life –** Número inicial de vidas;
  + **score –** Pontuação;
  + **score\_control –** Controle da pontuação para aumento da velocidade do jogo;
  + **velocity –** Velocidade inicial do jogo;
  + **status\_game –** Estado actual do jogo (decorrendo ou não);
  + **win / lost –** Variáveis de controle para recomeço de jogo**;**
  + **position []** – Vector com as 3 coordenadas (posições) possíveis no eixo y.

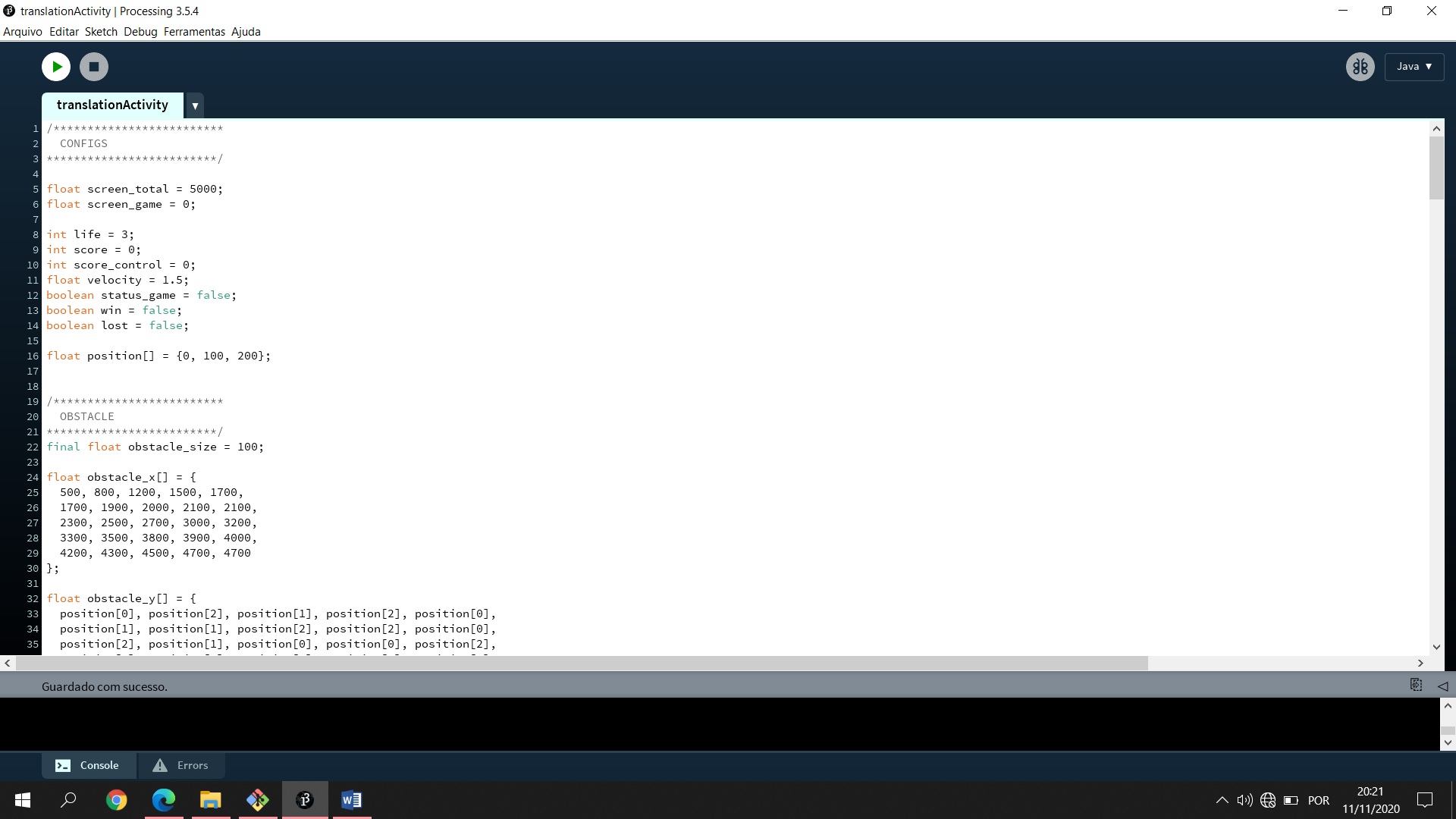


Figura : Variáveis globais de configuração do jogo.

Desenho dos objectos por dados e localizações (mais fácil para actualizar posições e redesenhar bem como testar colisões), como visto na **Figura 3**.

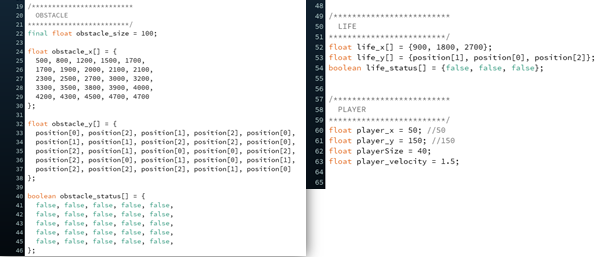


Figura 3: Variáveis globais de posicionamento e estado dos objectos.

## INICIALIZAÇÃO

Na inicialização do jogo, apenas foi definido o tamanho total da janela que recebe o jogo, sendo 800 de largura (x) e 500 de altura (y).

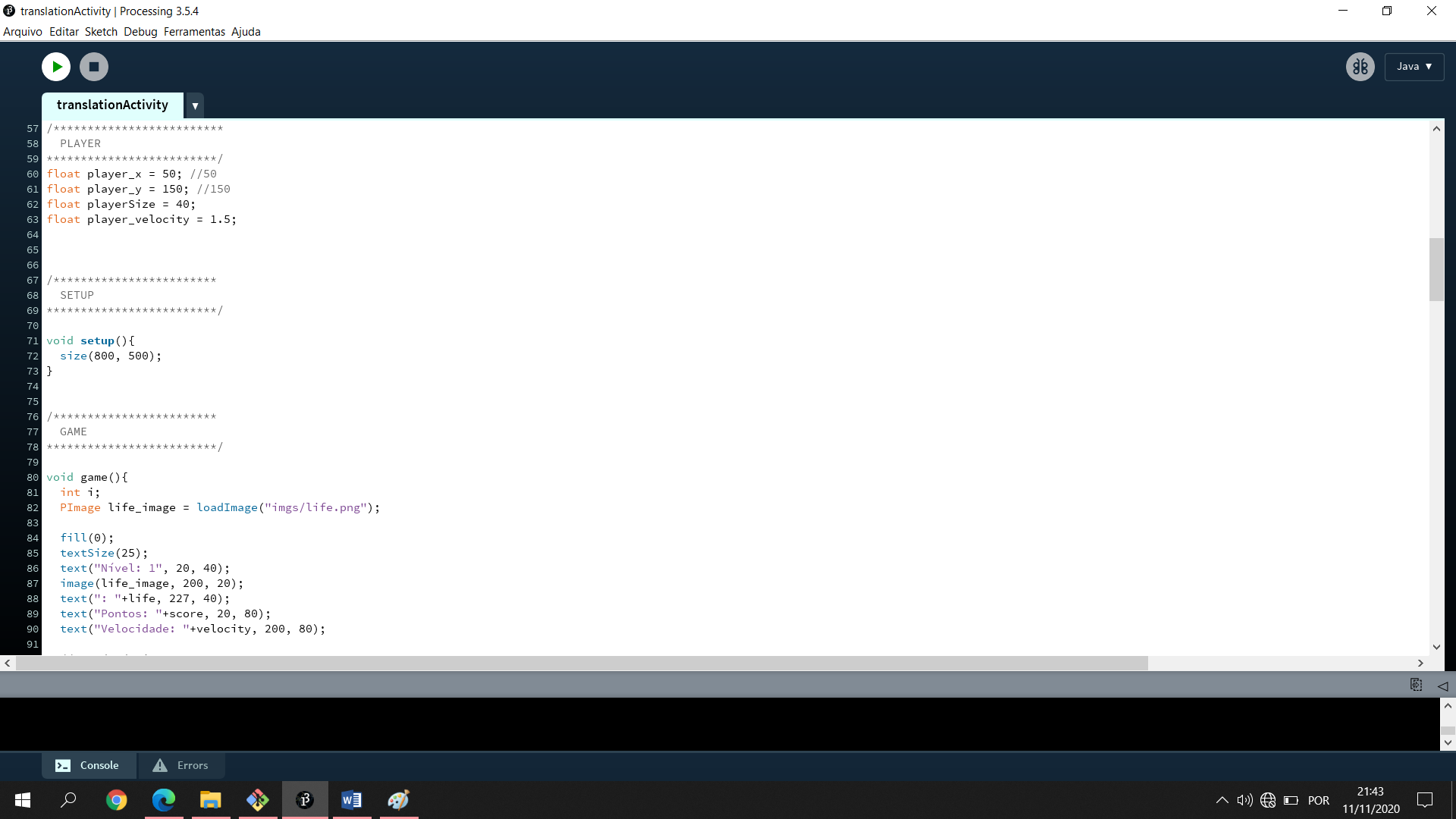


Figura 4: Inicialização do jogo.

## MÉTODO GAME

1. Carrega a imagem de vida na variável;
2. Estatística actual do jogo;
3. Aplicando um fundo na área do jogo;
4. Inicia o movimento de translação da janela (conjunto de objectos) na horizontal (da direita para esquerda), como é possível visualizar na **Figura 5**:
   * Posição da janela -= velocidade;
   * Aplicação da translação na janela;
   * Desenho os objectos:
     1. Obstáculos;
     2. Vidas;

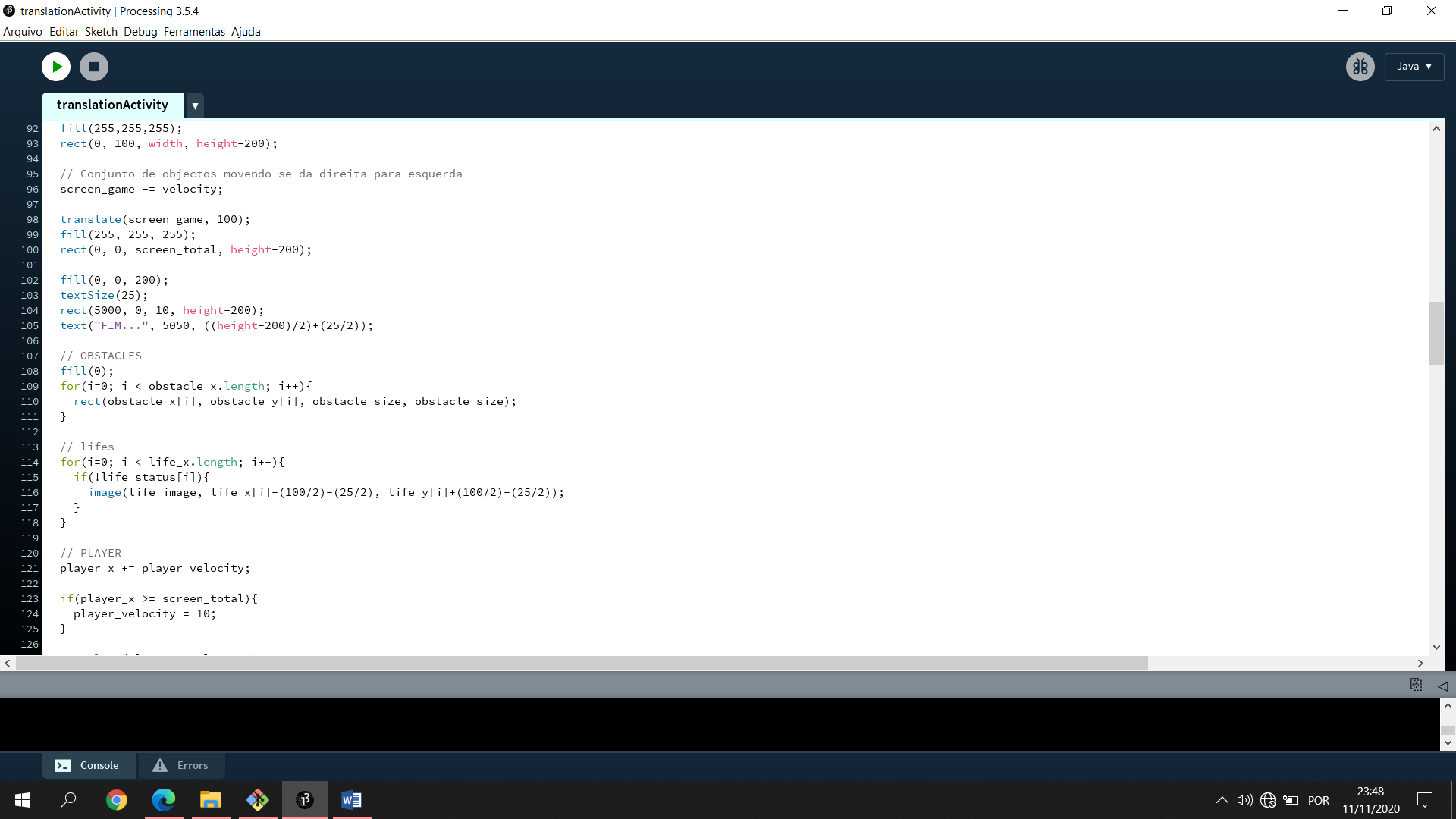


Figura 5: Transladando e desenhando o conjunto de objectos do jogo.

1. Inicia o movimento de translação do jogador na horizontal (da esquerda para direita):
   * Posição do jogador += velocidade do jogador;
   * Aplicação da translação no jogador;
   * Desenha o jogador;

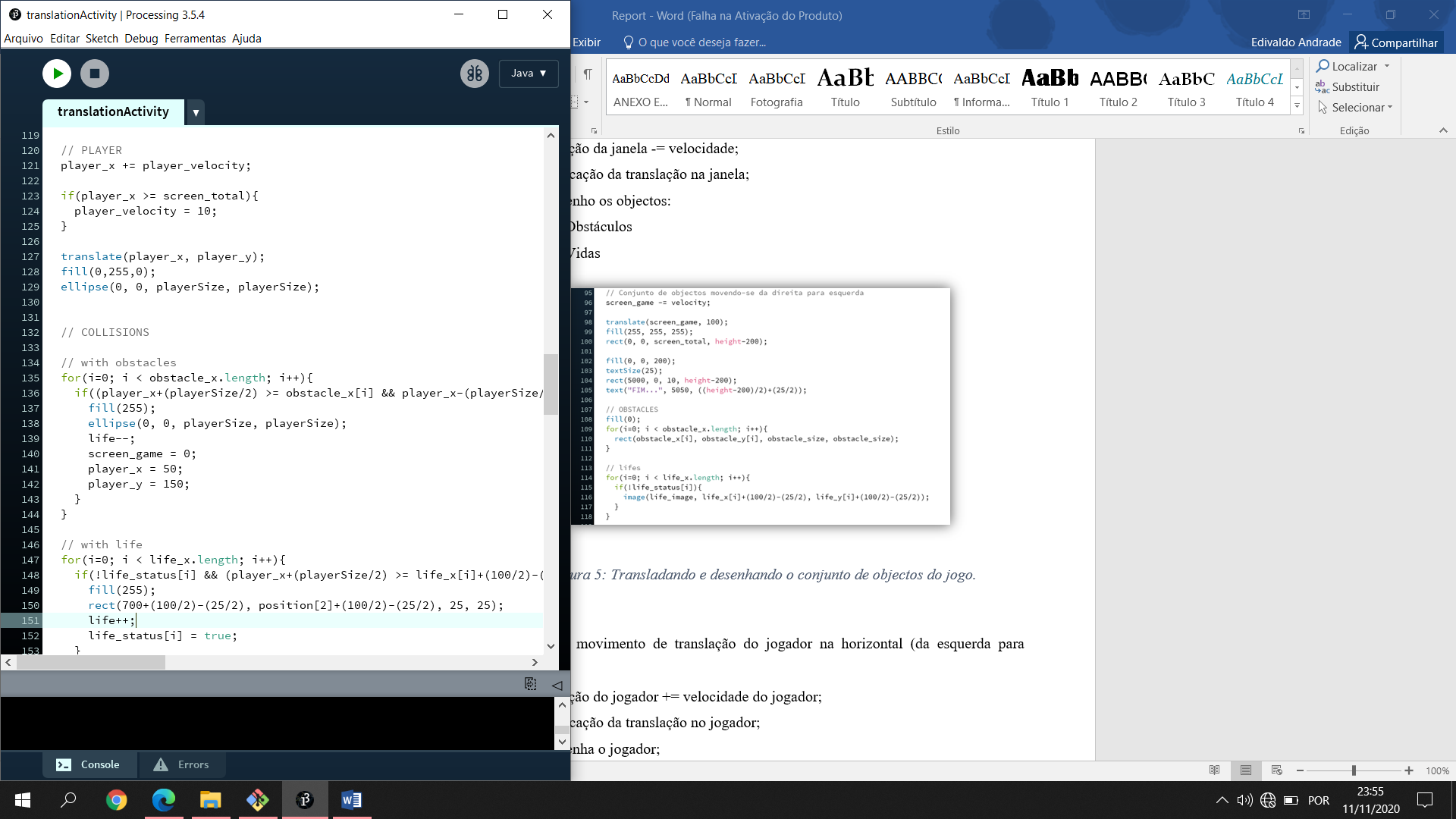


Figura 6: Transladando e desenhando o jogador.

1. Teste de colisão:
   * Do jogador com os obstáculos;
   * Do jogador com as vidas;

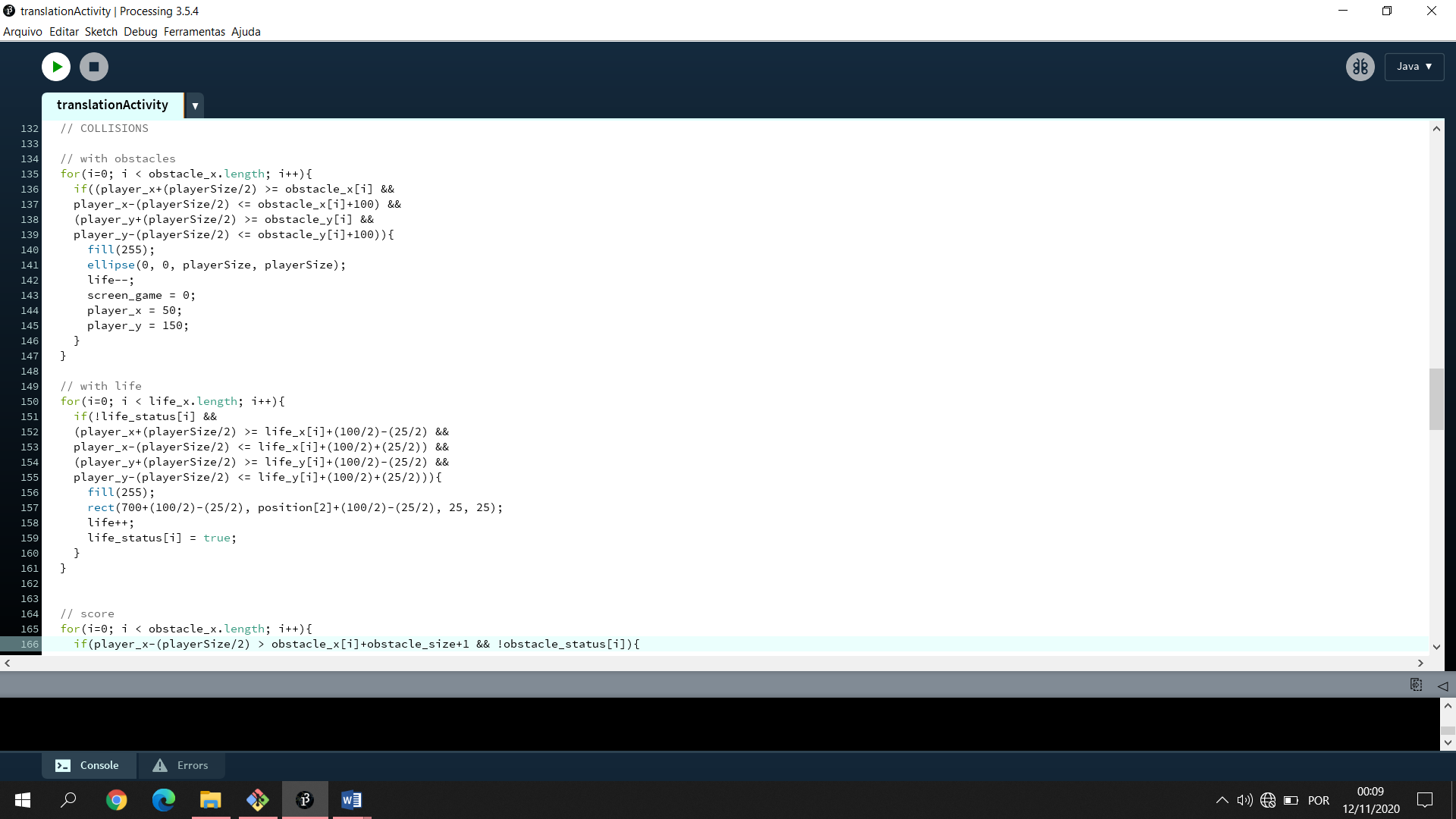


Figura 7: Verificando as possíveis colisões do jogador com os objectos.

1. Pontuação:
   * Verifica se o jogador passou os obstáculos;
   * Verifica se efectou 10 desvios (100 pontos), para aumento da velocidade de translação.

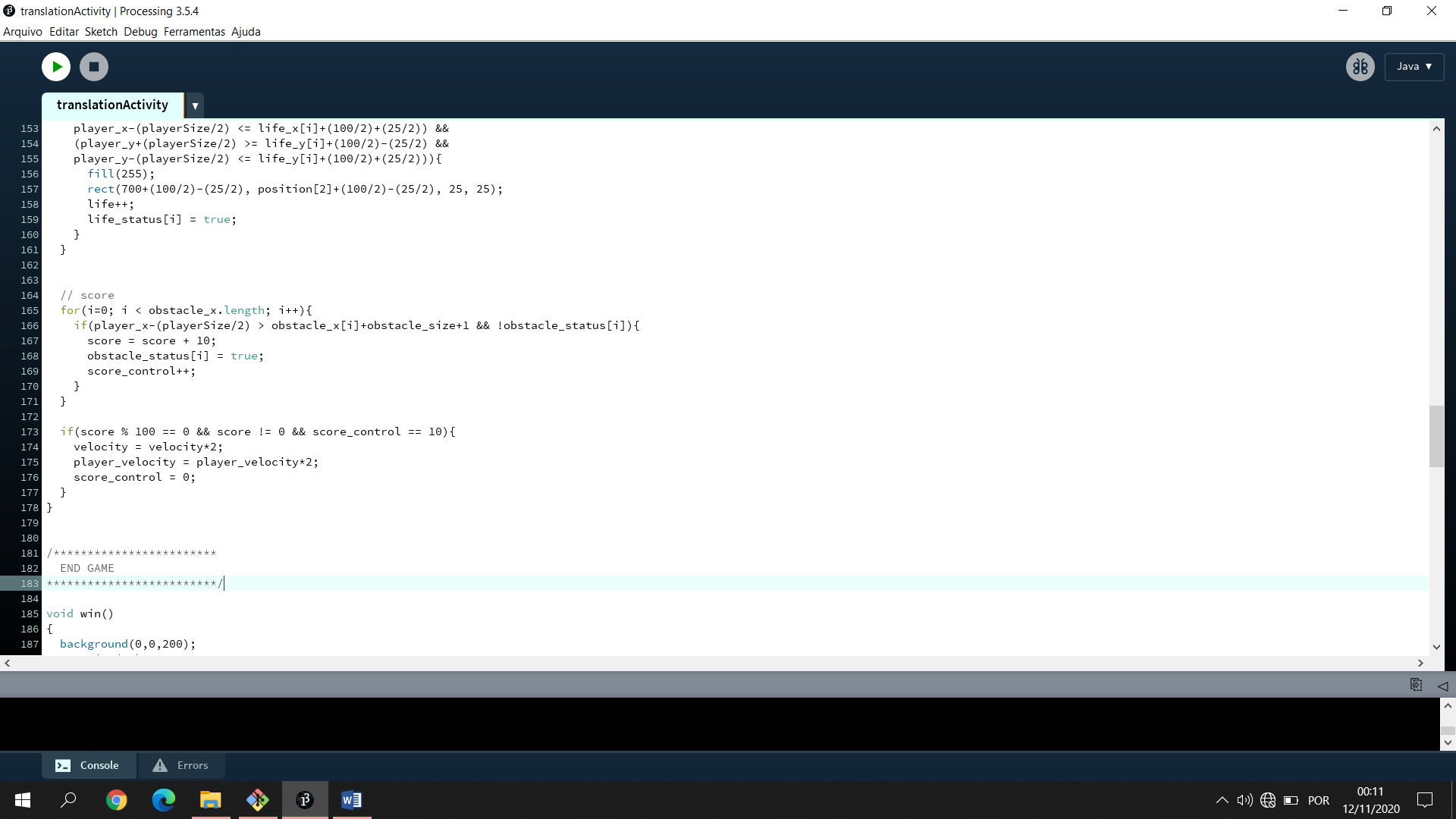


Figura 8: Somando a pontuação e actualizando a velocidade.

## MÉTODO PRINCIPAL (DRAW)

1. Refrescar o ecrã (background);
2. Texto da tela inicial do jogo;
3. Verificação de tecla pressionada para início do jogo;
4. Verifica se o estado do jogo está habilitado para chamada de função que executa o mesmo;
5. Verifica se perdeu (vendo se o número de vidas é **0**);
6. Verifica se ganhou o jogo (olhando para a posição da janela em translação do jogo).

## MÉTODOS SECUNDÁRIOS

* **void win()** – Função chamada em caso de vitória no jogo;
* **void lost()** – Função chamada em caso de derrota no jogo;
* **void reset()** – Restabelece os valores das variáveis globais de configuração, posição e estado do jogo;
* **keyPressed()** – Função responsável dos movimentos do jogador.

# TESTES E RESULTADOS

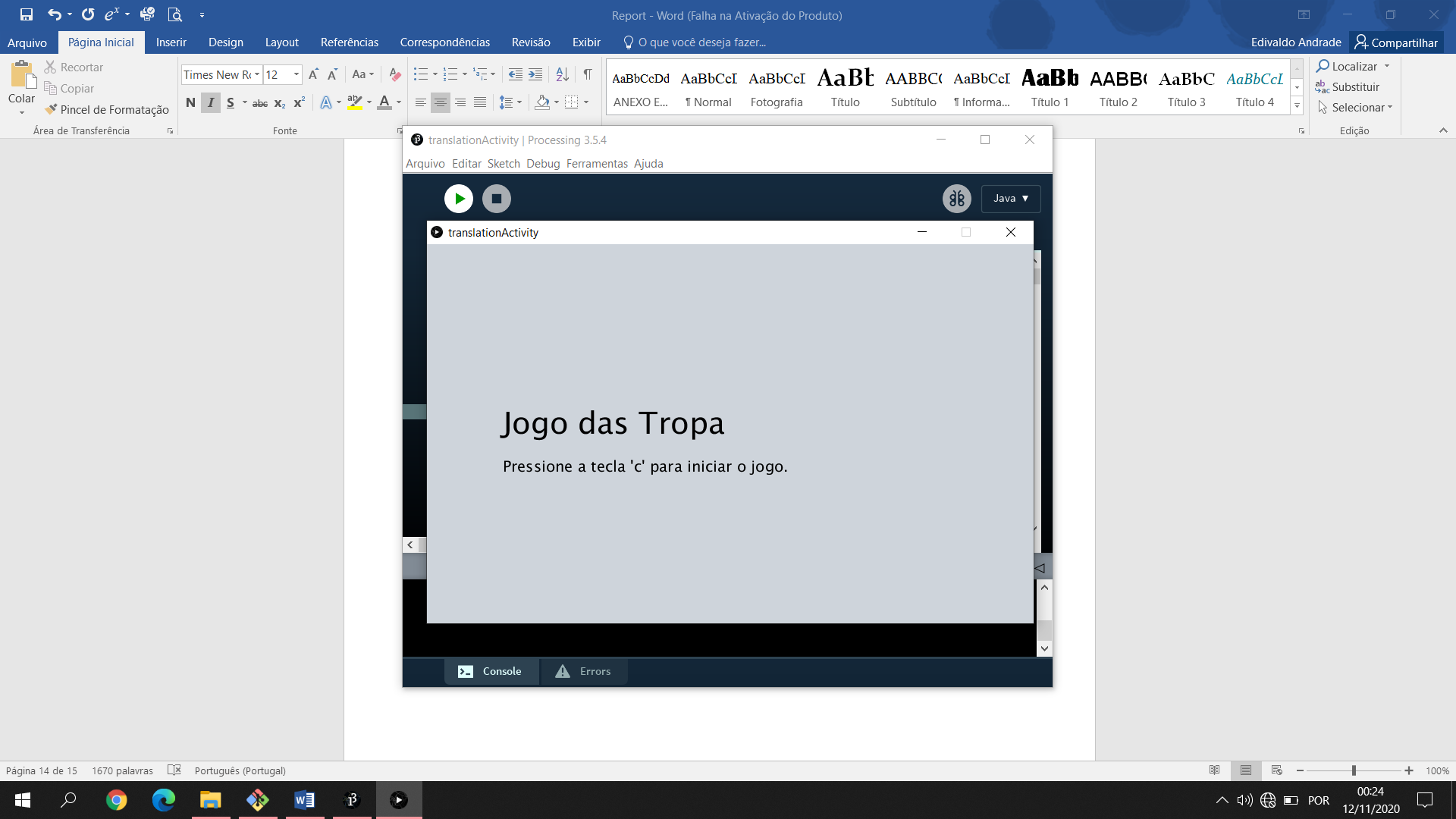


Figura 9: Tela inicial do jogo.

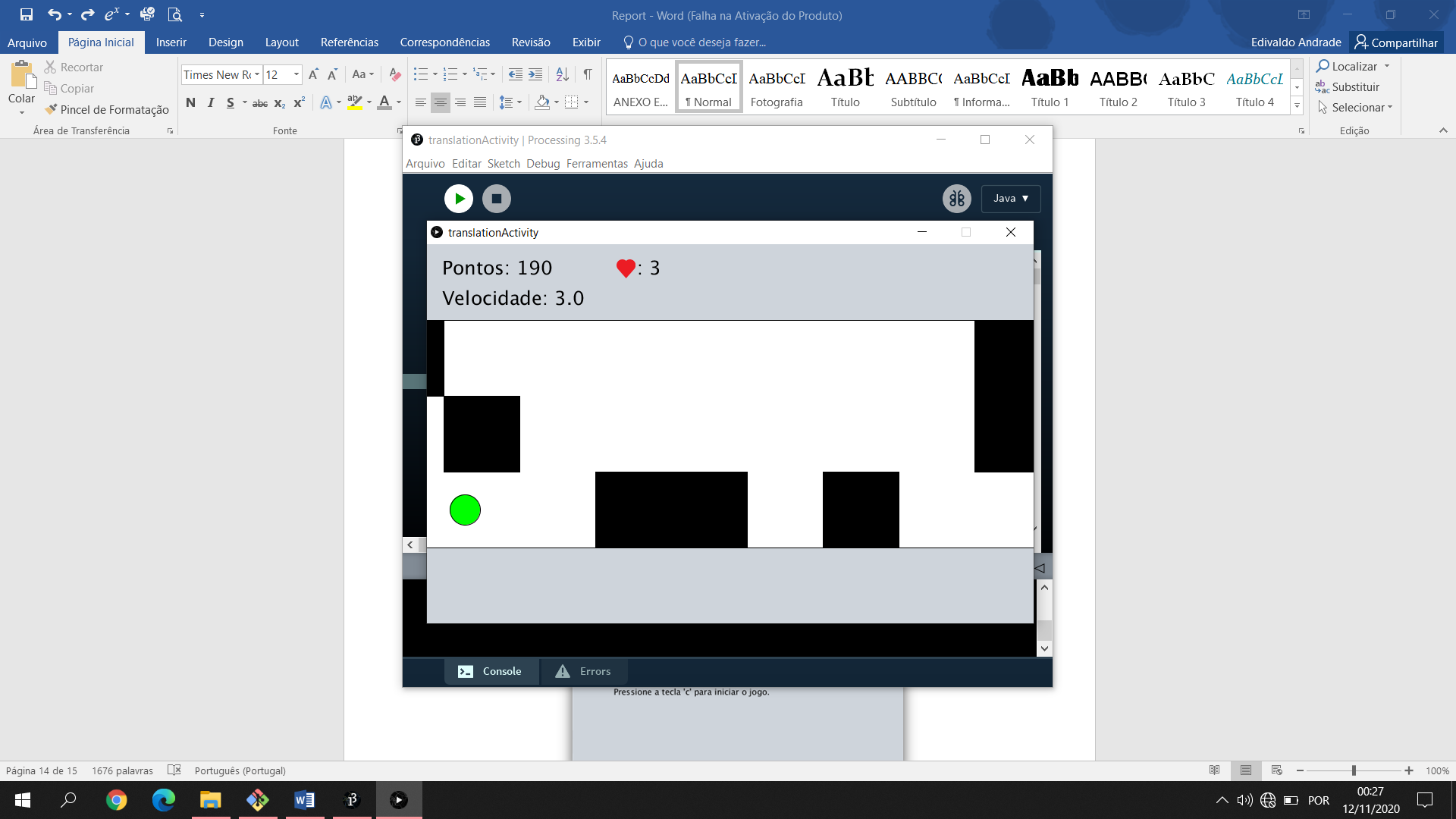


Figura 10: Jogo decorrendo.

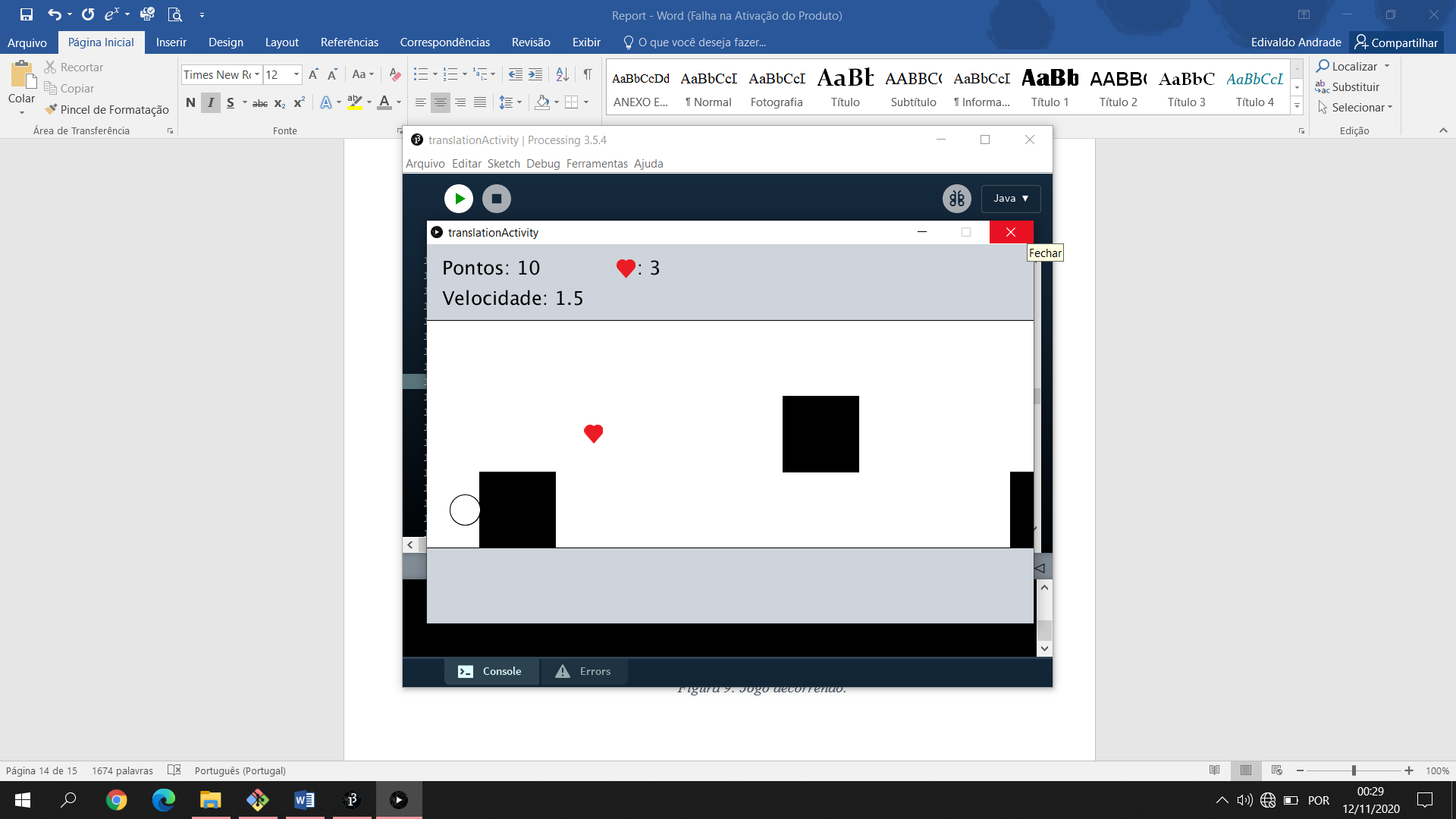


Figura 11: Colisão com um obstáculo.

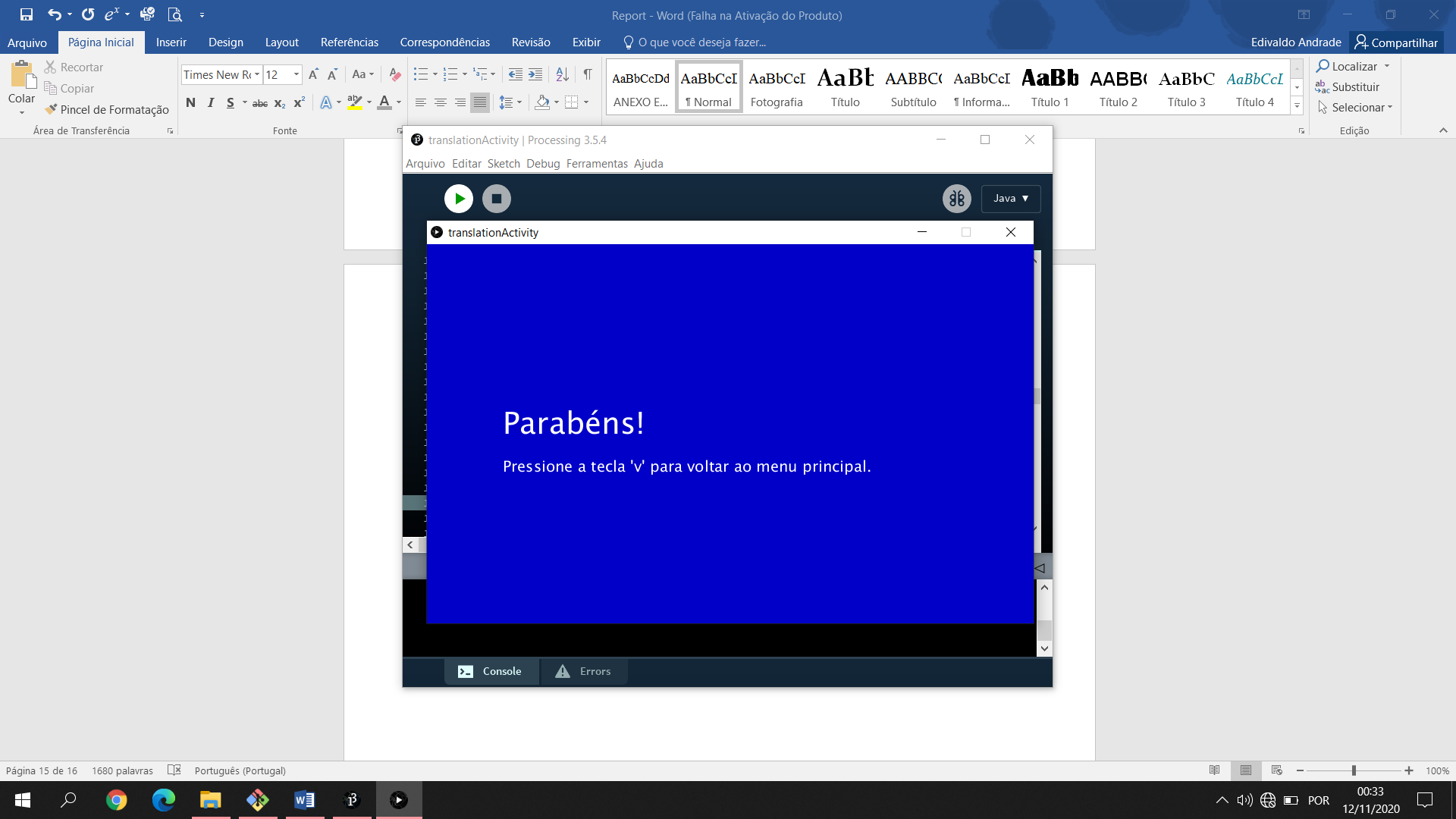


Figura 12: Vitória.

# REFERÊNCIAS

* AGOSTINHO, Solander. TRANSFORMAÇÕES GEOMÉTRICAS. Luanda: ISPTEC, 2020.
* Conci, Aura; AZEVEDO, Eduardo. TEORIA DA COMPUTAÇÃO GRÁFICA. RJ: Editora Campus.