

# Desenvolvimento de Simulador de Jogo de Dados Utilizados em Cassinos Americanos

Leonardo Melo<sup>1</sup>

<sup>1</sup> Programa de Pós-Graduação em Computação Aplicada

Departamento de Tecnologia (DTEC)  
Universidade Estadual de Feira de Santana (UEFS)  
Módulo 3, Av. Transnordestina, s/n – Novo Horizonte  
44.036-900 – Feira de Santana – BA – Brasil

leomelocomputacao@gmail.com

**Abstract.** *This article describes the details of the development of a common craps in U.S. casinos. The construction of the application considers the development paradigm structured and uses the programming language c++. The construction this program will assist in the analysis of its operation, in the game's simulation and in the demonstration of the real chances to be successful in this type of gambling.*

**keywords:** *Craps, C++, Structured Development Paradigm, Casino, Console Application, Handling Vectors.*

**Resumo.** *Este artigo descreve os detalhes do desenvolvimento de um jogo de dados comum em cassinos americanos. A construção da aplicação considerou o paradigma de desenvolvimento de software estruturado e utilizou a linguagem c++. A construção deste programa auxiliará na análise do seu funcionamento, na simulação de jogadas e na demonstração das reais chances de se obter sucesso nesse tipo de jogo de azar.*

**Palavras-chave:** *Jogo de Dados, C++, Paradigma de Desenvolvimento Estruturado, Cassino, Aplicação Console, Manipulação de Vetores.*

## 1. Introdução

Em visita aos Estados Unidos, onde os casinos não são ilegais, Álvaro Ento e um grupo de pesquisadores passaram dias jogando e um jogo que chamou a atenção deles foi o Jogo de Dados. Álvaro registrou o comportamento do jogo para posteriores análises de lucros ou prejuízos de apostadores.

Então, foi solicitado a Universidade Estadual de Feira de Santana (UEFS) o desenvolvimento de um programa de computador para auxiliá-los nessas análises. Eles não tem muita familiaridade com interfaces complexas de programas e solicitaram a maior simplicidade possível.

Um dos pesquisadores tem conhecimento básico de programação nas linguagens C e C++. Visando facilitar uma possível manutenção ou melhoria simples do programa proposto foi solicitado o desenvolvimento em uma dessas linguagens.

### 1.1. Funcionamento do Jogo de Dados

O jogo de dados funciona da seguinte maneira: a cada jogada, o jogador aposta um valor e lança dois dados e o croupier (funcionário do cassino) lança dois dados também.

Se o jogador acertar o valor exato dos dois dados ganha 6 vezes o valor apostado, se acerta a soma, ganha 3 vezes o valor apostado.

Se a paridade da soma dos dados do jogador é igual a paridade da soma dos dados do croupier, o jogador ganha de volta metade do valor apostado. Os ganhos não são acumulativos, ou seja, o jogador só pode ganhar por um dos três critérios possíveis (o de maior valor).

### 1.2. Casos de Teste

Com base nos registros dos pesquisadores e nas regras do jogo foi criado um conjunto de casos de teste que contemplam todos os possíveis cenários do jogo. Esses casos de testes são apresentados abaixo:

Digite o valor apostado: 1  
Digite o valor do primeiro dado do jogador: 1  
Digite o valor do segundo dado do jogador: 2  
Digite o valor do primeiro dado do croupier: 2  
Digite o valor do segundo dado do croupier: 1  
O jogador ganhou 6 reais.

Digite o valor apostado: 1  
Digite o valor do primeiro dado do jogador: 1  
Digite o valor do segundo dado do jogador: 6  
Digite o valor do primeiro dado do croupier: 2  
Digite o valor do segundo dado do croupier: 5  
O jogador ganhou 3 reais.

Digite o valor apostado: 1  
Digite o valor do primeiro dado do jogador: 1  
Digite o valor do segundo dado do jogador: 2  
Digite o valor do primeiro dado do croupier: 2  
Digite o valor do segundo dado do croupier: 3  
O jogador ganhou 0,50 reais.

Digite o valor apostado: 1  
Digite o valor do primeiro dado do jogador: 1  
Digite o valor do segundo dado do jogador: 2  
Digite o valor do primeiro dado do croupier: 2  
Digite o valor do segundo dado do croupier: 4  
O jogador perdeu.

### 1.3. Requisitos para Análise

Para auxiliar na análise das partidas do jogo foi solicitado que o programa a ser desenvolvido permita a entrada de várias jogadas até que seja digitado um valor igual ou menor que 0 para o valor apostado. Quando isso acontecer o programa deverá mostrar um conjunto

de informações. Abaixo um exemplo do conjunto de informações mencionado anteriormente:

```
O jogador jogou 4 vez(es).  
O saldo final foi de 5,5 reais.  
O valor do dado que mais saiu foi 2.  
O maior valor ganho em uma partida foi 6 reais.  
O jogador perdeu 1 vez(es).
```

## 2. Metodologia

O paradigma de programação utilizado foi o Estruturado devido a maior familiaridade dos desenvolvedores envolvidos no projeto do jogo.

Para estar em acordo com o requisito de interface com o usuário simples foi criado um programa para executar em ambiente console, aplicação console (Figura 1).

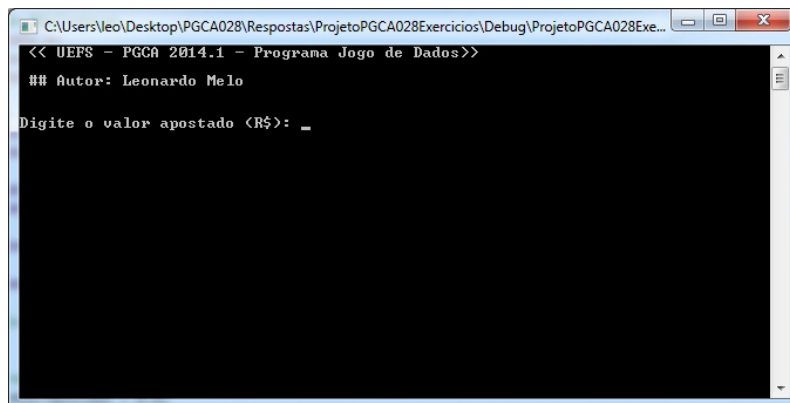


Figura 1. Tela inicial do Jogo de Dados (Aplicação Console).

Optou-se pela utilização do IDE Visual Studio por aumentar a produtividade no desenvolvimento e oferecer suporte a C/C++ (Figura 2).

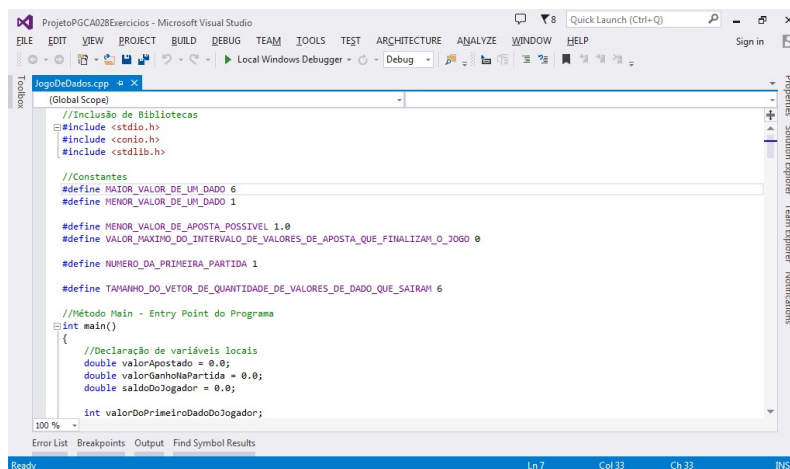


Figura 2. Tela do IDE Visual Studio 2013 exibindo o código fonte do jogo de dados.

Constantes foram criadas para auxiliar na manutenção, facilitar a mudança de parâmetros importantes e evitar a repetição de valores dentro do programa. Essas constantes definem os valores mínimo e máximo de um dado, o menor valor de uma aposta, o tamanho do vetor utilizado, dentro outros (Algoritmo 1).

```
1 #define MAIOR_VALOR_DE_UM_DADO 6
2 #define MENOR_VALOR_DE_UM_DADO 1
3 #define MENOR_VALOR_DE_APOSTA_POSSIVEL 1.0
4 #define VALOR_MAXIMO_DE_FINALIZACAO_DO_JOGO 0
5 #define NUMERO_DA_PRIMEIRA_PARTIDA 1
6 #define TAMANHO_DO_VETOR_DE_VALORES_DE_DADO_QUE_SAIRAM 6
```

**Algoritmo 1. Constantes utilizadas.**

Todas as variáveis do programa são declaradas no início do programa por questões de organização e centralização e apenas inicializadas quando necessário. O valor apostado em uma partida, o ganho por partida, o saldo final do jogador, os valores dos dados que saíram, a quantidade de partidas são algumas das informações armazenadas durante a execução do sistema (Algoritmo 1).

```
1 double valorApostado = 0.0;
2 double valorGanhoNaPartida = 0.0;
3 double saldoDoJogador = 0.0;
4 int valorDoPrimeiroDadoDoJogador;
5 int valorDoSegundoDadoDoJogador;
6 int valorDoPrimeiroDadoDoCroupier;
7 int valorDoSegundoDadoDoCroupier;
8 bool dadosDeEntradaValidos = true;
9 bool sairDaColetaDeDadosDoJogo = false;
10 int quantidadeDePartidas = 0;
11 int quantidadeDePartidasSemGanhos = 0;
12 double maiorValorGanhoEmUmaPartida =
    MENOR_VALOR_DE_APOSTA_POSSIVEL;
13 int contadoresDeValoresDeDadoQueSairam
14 [TAMANHO_DO_VETOR_DE_VALORES_DE_DADO_QUE_SAIRAM];
15 int maiorValorDeDadoQueSaiu = 0;
```

**Algoritmo 2. Variáveis do programa.**

É usado um vetor com 6 posições para armazenar os contadores do número de vezes que um valor do dado aparece no jogo. Então, como esse vetor armazena contadores é necessário zerá-los antes de incrementá-los (Algoritmo 3).

```
1 for (int i = 0; i <
    TAMANHO_DO_VETOR_DE_VALORES_DE_DADO_QUE_SAIRAM; i++)
2 {
3     contadoresDeValoresDeDadoQueSairam[i] = 0;
4 }
```

**Algoritmo 3. Zerando vetor de contadores.**

O programa é basicamente composto de uma estrutura de repetição e essa estrutura permanece iterando até que seja digitado um valor menor ou igual a 0. Quando a condição de parada é atingida um comando break interrompe essa repetição (Algoritmo 3).

```

1 if (valorApostado <= VALOR_MAXIMO_DE_FINALIZACAO_DO_JOGO)
2 {
3     sairDaColetaDeDadosDoJogo = true;
4     break;
5 }

```

**Algoritmo 4. Condição de parada do programa.**

Dentro da estrutura de repetição principal do jogo existem quatro etapas: a captura da informação da quantidade apostada (Algoritmo 5), a captura dos valores dos dados do jogador (Algoritmos 6 e 7), a captura dos valores dos dados do croupier (Algoritmos 8 e 9) e a apresentação do resultado da jogada (Algoritmo 10).

```

1 do{
2     dadosDeEntradaValidos = true;
3
4     printf_s("\n\nDigite o valor apostado (R$): ");
5     scanf_s("%lf", &valorApostado);
6
7     fflush(stdin);
8
9     if (valorApostado <= VALOR_MAXIMO_DE_FINALIZACAO_DO_JOGO)
10    {
11        sairDaColetaDeDadosDoJogo = true;
12        break;
13    }
14
15    if (valorApostado < MENOR_VALOR_DE_APOSTA_POSSIVEL)
16    {
17        printf_s("\n\nEntrada Invalida!\no valor apostado nao pode
18            ser menor que R$ %.2lf.",
19            MENOR_VALOR_DE_APOSTA_POSSIVEL);
20
21        dadosDeEntradaValidos = false;
22    }
23 } while (!dadosDeEntradaValidos);

```

**Algoritmo 5. Captura da quantidade apostada e validações.**

```

1 do{
2     dadosDeEntradaValidos = true;
3
4     printf_s("\n\nDigite o valor do primeiro dado do jogador: ");
5     scanf_s("%d", &valorDoPrimeiroDadoDoJogador);
6
7     fflush(stdin);
8
9     if (valorDoPrimeiroDadoDoJogador > MAIOR_VALOR_DE_UM_DADO ||
10        valorDoPrimeiroDadoDoJogador < MENOR_VALOR_DE_UM_DADO)
11    {

```

```

11     printf_s("\n\nEntrada Invalida!\nO valor do dado nao pode
        ser menor que %d ou maior que %d.",
        MENOR_VALOR_DE_UM_DADO, MAIOR_VALOR_DE_UM_DADO);
12
13     dadosDeEntradaValidos = false;
14 }
15
16 } while (!dadosDeEntradaValidos);

```

**Algoritmo 6. Captura do valor do primeiro dado do jogador e validações.**

```

1 do{
2     dadosDeEntradaValidos = true;
3
4     printf_s("\n\nDigite o valor do segundo dado do jogador: ");
5     scanf_s("%d", &valorDoSegundoDadoDoJogador);
6
7     fflush(stdin);
8
9     if (valorDoSegundoDadoDoJogador > MAIOR_VALOR_DE_UM_DADO ||
        valorDoSegundoDadoDoJogador < MENOR_VALOR_DE_UM_DADO)
10    {
11        printf_s("\n\nEntrada Invalida!\nO valor do dado nao pode
                ser menor que %d ou maior que %d.",
                MENOR_VALOR_DE_UM_DADO, MAIOR_VALOR_DE_UM_DADO);
12
13        dadosDeEntradaValidos = false;
14    }
15
16 } while (!dadosDeEntradaValidos);

```

**Algoritmo 7. Captura do valor do segundo dado do jogador e validações.**

```

1 do{
2     dadosDeEntradaValidos = true;
3
4     printf_s("\n\nDigite o valor do primeiro dado do croupier: ")
        ;
5     scanf_s("%d", &valorDoPrimeiroDadoDoCroupier);
6
7     fflush(stdin);
8
9     if (valorDoPrimeiroDadoDoCroupier > MAIOR_VALOR_DE_UM_DADO ||
        valorDoPrimeiroDadoDoCroupier < MENOR_VALOR_DE_UM_DADO)
10    {
11        printf_s("\n\nEntrada Invalida!\nO valor do dado nao pode
                ser menor que %d ou maior que %d.",
                MENOR_VALOR_DE_UM_DADO, MAIOR_VALOR_DE_UM_DADO);
12
13        dadosDeEntradaValidos = false;
14    }

```

```
15
16 } while (!dadosDeEntradaValidos);
```

**Algoritmo 8. Captura do valor do primeiro dado do croupier e validações.**

```
1 do{
2     dadosDeEntradaValidos = true;
3
4     printf_s("\n\nDigite o valor do segundo dado do croupier: ");
5     scanf_s("%d", &valorDoSegundoDadoDoCroupier);
6
7     fflush(stdin);
8
9     if (valorDoSegundoDadoDoCroupier > MAIOR_VALOR_DE_UM_DADO ||
10        valorDoSegundoDadoDoCroupier < MENOR_VALOR_DE_UM_DADO)
11     {
12         printf_s("\n\nEntrada Invalida!\no valor do dado nao pode
13             ser menor que %d ou maior que %d.",
14                 MENOR_VALOR_DE_UM_DADO, MAIOR_VALOR_DE_UM_DADO);
15
16         dadosDeEntradaValidos = false;
17     }
18 } while (!dadosDeEntradaValidos);
```

**Algoritmo 9. Captura do valor do segundo dado do croupier e validações.**

```
1 bool ganhouAlgumValor = false;
2
3 if ((valorDoPrimeiroDadoDoJogador ==
4     valorDoPrimeiroDadoDoCroupier && valorDoSegundoDadoDoJogador
5     == valorDoSegundoDadoDoCroupier) || (
6     valorDoPrimeiroDadoDoJogador == valorDoSegundoDadoDoCroupier
7     && valorDoSegundoDadoDoJogador ==
8     valorDoPrimeiroDadoDoCroupier))
9 {
10     valorGanhoNaPartida = valorApostado * 6;
11     ganhouAlgumValor = true;
12
13     saldoDoJogador += valorGanhoNaPartida;
14
15     if (maiorValorGanhoEmUmaPartida < valorGanhoNaPartida)
16     {
17         maiorValorGanhoEmUmaPartida = valorGanhoNaPartida;
18     }
19
20     printf_s("\n\no jogador ganhou R$ %.2lf.",
21             valorGanhoNaPartida);
22 }
23
24 if (!ganhouAlgumValor)
```

```

19 {
20     if ((valorDoPrimeiroDadoDoJogador +
21         valorDoSegundoDadoDoJogador) == (
22         valorDoPrimeiroDadoDoCroupier +
23         valorDoSegundoDadoDoCroupier))
24     {
25         valorGanhoNaPartida = valorApostado * 3;
26         ganhouAlgumValor = true;
27
28         saldoDoJogador += valorGanhoNaPartida;
29
30         if (maiorValorGanhoEmUmaPartida < valorGanhoNaPartida)
31         {
32             maiorValorGanhoEmUmaPartida = valorGanhoNaPartida;
33         }
34
35         printf_s("\n\nO jogador ganhou R$ %.2lf.",
36                 valorGanhoNaPartida);
37     }
38 }
39
40 if (!ganhouAlgumValor)
41 {
42     if (((valorDoPrimeiroDadoDoJogador +
43         valorDoSegundoDadoDoJogador) % 2 == 0) && ((
44         valorDoPrimeiroDadoDoCroupier +
45         valorDoSegundoDadoDoCroupier) % 2 == 0) || ((
46         valorDoPrimeiroDadoDoJogador + valorDoSegundoDadoDoJogador
47         ) % 2 != 0) && ((valorDoPrimeiroDadoDoCroupier +
48         valorDoSegundoDadoDoCroupier) % 2 != 0))
49     {
50         valorGanhoNaPartida = valorApostado / 2;
51         ganhouAlgumValor = true;
52
53         saldoDoJogador += valorGanhoNaPartida;
54
55         if (maiorValorGanhoEmUmaPartida < valorGanhoNaPartida)
56         {
57             maiorValorGanhoEmUmaPartida = valorGanhoNaPartida;
58         }
59
60         printf_s("\n\nO jogador ganhou R$ %.2lf.",
61                 valorGanhoNaPartida);
62     }
63 }
64
65 if (!ganhouAlgumValor)
66 {
67     valorGanhoNaPartida = 0.0;

```



```

57     quantidadeDePartidasSemGanhos++;
58
59
60     printf_s("\n\nO jogador perdeu.");
61 }

```

**Algoritmo 10. Cálculos e apresentação do resultado da jogada.**

Após o encerramento das partidas é apresentado um conjunto de informações, inclusive informações estatísticas, a respeito de todas as partidas (Algoritmo 11).

```

1  printf_s("\n\n\nJogo Encerrado!");
2
3  printf_s("\n\nO jogador jogou %d vez(es).",
4           quantidadeDePartidas);
5  printf_s("\n\nO saldo final foi de %.2lf reais.", saldoDoJogador);
6
7  for (int i = 0; i <
8       TAMANHO_DO_VETOR_DE_VALORES_DE_DADO_QUE_SAIRAM; i++)
9  {
10     if (contadoresDeValoresDeDadoQueSairam[i] >
11         maiorValorDeDadoQueSaiu)
12     {
13         maiorValorDeDadoQueSaiu =
14             contadoresDeValoresDeDadoQueSairam[i];
15     }
16 }
17
18 for (int i = 0; i <
19      TAMANHO_DO_VETOR_DE_VALORES_DE_DADO_QUE_SAIRAM; i++)
20 {
21     if (contadoresDeValoresDeDadoQueSairam[i] ==
22         maiorValorDeDadoQueSaiu)
23     {
24         printf_s("\n\nO valor do dado que mais saiu foi %d, %d vez
25                 (es).", i + 1, contadoresDeValoresDeDadoQueSairam[i]);
26     }
27 }
28
29 printf_s("\n\nO maior valor ganho em uma partida foi %.2lf reais
30         .", maiorValorGanhoEmUmaPartida);
31 printf_s("\n\nO jogador perdeu %d vez(es).",
32         quantidadeDePartidasSemGanhos);

```

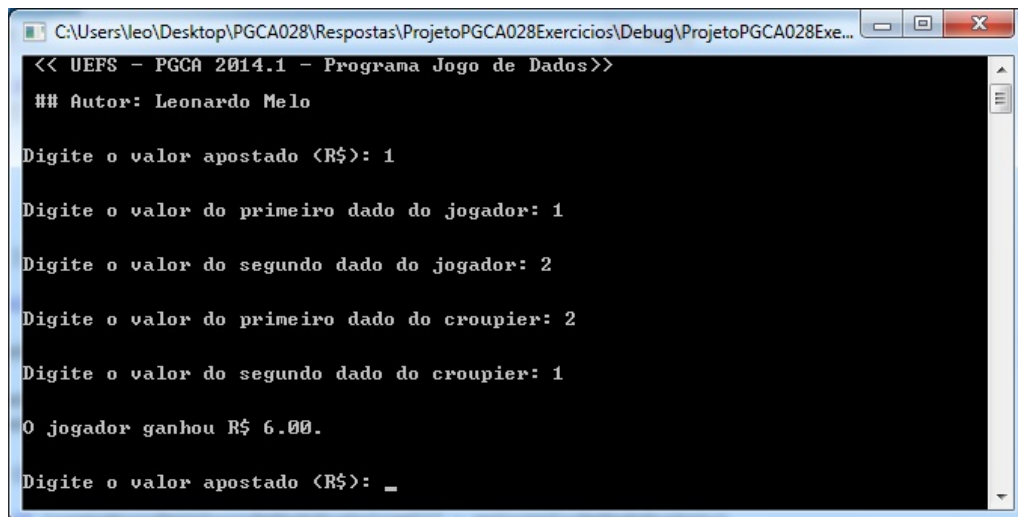
**Algoritmo 11. Apresentação dos dados de encerramento do programa.**

Em vários pontos do código fonte é possível perceber a presença de contadores em pontos estratégicos. Esses contadores são essenciais para a correta apresentação dos dados após o encerramento das partidas.

### 3. Resultados

Nesta seção serão apresentados os resultados dos testes do programa e sua correspondência com os casos de teste demonstrando o correto funcionamento do jogo.

Ao apostar 1 real, o primeiro valor do dado do jogador ser 1, o segundo valor do dado do jogador ser 2, o primeiro valor do dado do croupier ser 2 e o segundo valor do dado do croupier ser 1, o jogador ganha 6 reais (Figura 3).



```
<< UEFS - PGCA 2014.1 - Programa Jogo de Dados>>
## Autor: Leonardo Melo

Digite o valor apostado (R$): 1

Digite o valor do primeiro dado do jogador: 1

Digite o valor do segundo dado do jogador: 2

Digite o valor do primeiro dado do croupier: 2

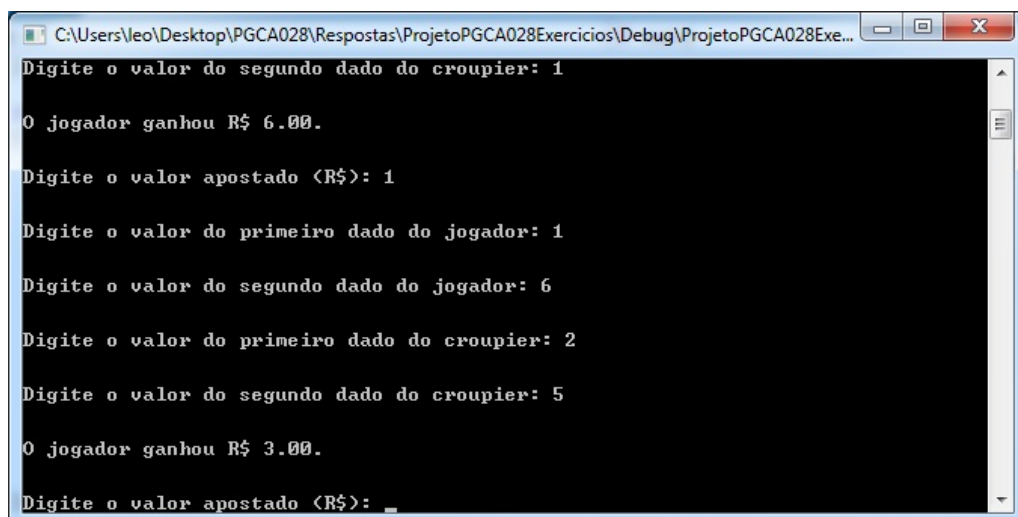
Digite o valor do segundo dado do croupier: 1

O jogador ganhou R$ 6.00.

Digite o valor apostado (R$): _
```

Figura 3. Tela do sistema executando a primeira aposta.

Ao apostar 1 real, o primeiro valor do dado do jogador ser 1, o segundo valor do dado do jogador ser 6, o primeiro valor do dado do croupier ser 2 e o segundo valor do dado do croupier ser 5, o jogador ganha 3 reais (Figura 4).



```
Digite o valor do segundo dado do croupier: 1

O jogador ganhou R$ 6.00.

Digite o valor apostado (R$): 1

Digite o valor do primeiro dado do jogador: 1

Digite o valor do segundo dado do jogador: 6

Digite o valor do primeiro dado do croupier: 2

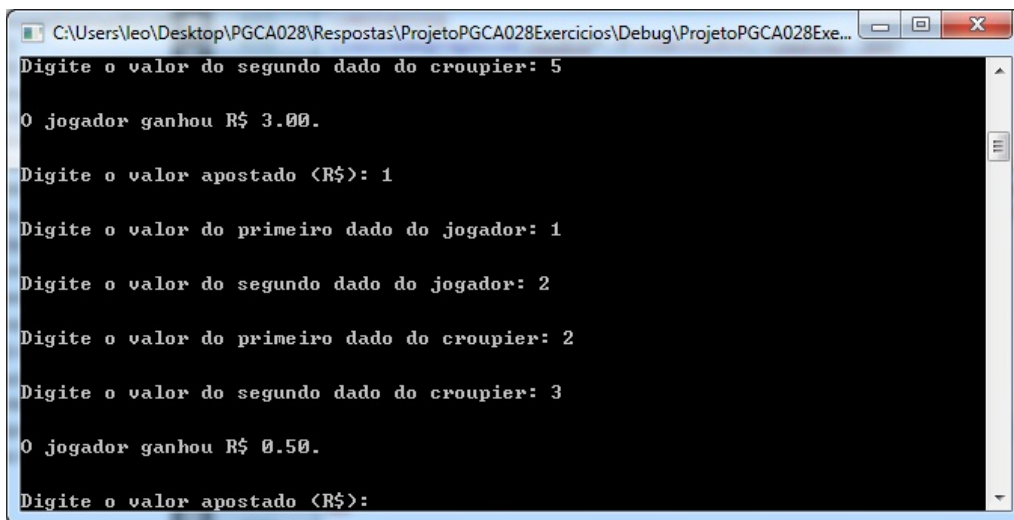
Digite o valor do segundo dado do croupier: 5

O jogador ganhou R$ 3.00.

Digite o valor apostado (R$): _
```

Figura 4. Tela do sistema executando a segunda aposta.

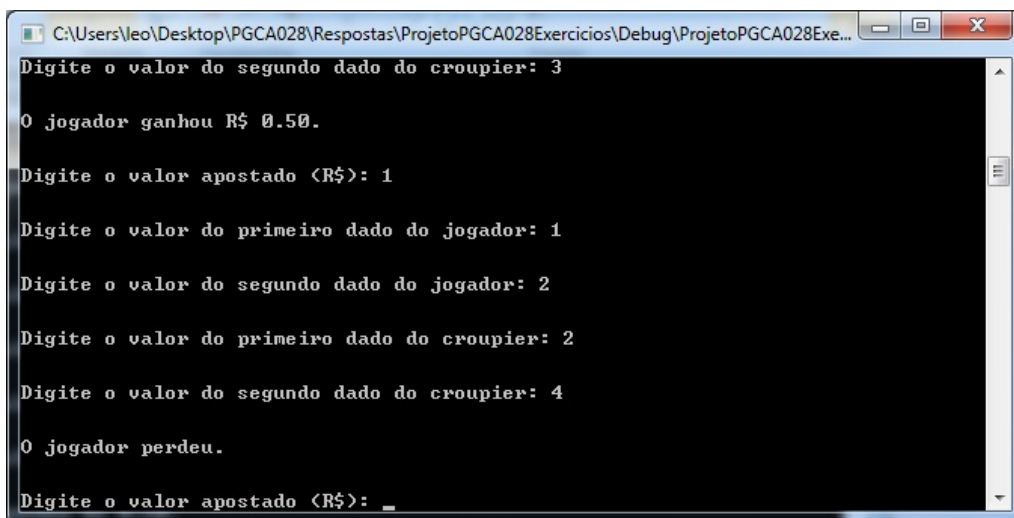
Ao apostar 1 real, o primeiro valor do dado do jogador ser 1, o segundo valor do dado do jogador ser 2, o primeiro valor do dado do croupier ser 2 e o segundo valor do dado do croupier ser 3, o jogador ganha 0,5 reais (Figura 5).



```
C:\Users\leo\Desktop\PGCA028\Respostas\ProjetoPGCA028Exercicios\Debug\ProjetoPGCA028Exe...
Digite o valor do segundo dado do croupier: 5
O jogador ganhou R$ 3.00.
Digite o valor apostado (R$): 1
Digite o valor do primeiro dado do jogador: 1
Digite o valor do segundo dado do jogador: 2
Digite o valor do primeiro dado do croupier: 2
Digite o valor do segundo dado do croupier: 3
O jogador ganhou R$ 0.50.
Digite o valor apostado (R$):
```

**Figura 5. Tela do sistema executando a terceira aposta.**

Ao apostar 1 real, o primeiro valor do dado do jogador ser 1, o segundo valor do dado do jogador ser 2, o primeiro valor do dado do croupier ser 2 e o segundo valor do dado do croupier ser 4, o jogador perde todo dinheiro apostado na rodada (Figura 6).



```
C:\Users\leo\Desktop\PGCA028\Respostas\ProjetoPGCA028Exercicios\Debug\ProjetoPGCA028Exe...
Digite o valor do segundo dado do croupier: 3
O jogador ganhou R$ 0.50.
Digite o valor apostado (R$): 1
Digite o valor do primeiro dado do jogador: 1
Digite o valor do segundo dado do jogador: 2
Digite o valor do primeiro dado do croupier: 2
Digite o valor do segundo dado do croupier: 4
O jogador perdeu.
Digite o valor apostado (R$): _
```

**Figura 6. Tela do sistema executando a quarta aposta.**

Ao encerrar esse grupo de testes os dados de encerramento tem que informar que houve 4 jogadas, o maior valor ganho em uma partida foi 6 reais , o jogador perdeu 1 vez, o número de dado que mais saiu foi o 2 (7 vezes) e que o saldo do jogador é de 5,5 reais (Figura 7).

#### 4. Discussões

Apesar do programa ser implementado em conformidade com os casos de teste, uma discussão surgiu sobre um conceito importante e que pode ser motivo de uma futura manutenção, o saldo final.

**Figura 7. Tela do sistema mostrando os dados de encerramento do jogo.**

Pela definição mais simples, para se obter o saldo basta subtrair o total apostado do total ganho. Então, se partirmos dessa definição devemos considerar que quando o jogador realiza a primeira aposta ele fica com saldo negativo (menos x reais no bolso).

Uma outra visão (aparentemente mais lógica) é que o jogador sempre começa com um valor inicial a ser apostado e mesmo que aposte toda quantia na primeira jogada ele não fica com saldo negativo e sim zerado.

A primeira visão foi implementada por se adequar as casos de teste. Mas, acredita-se que a segunda definição é a correta pois não é possível dever ao cassino.

## **5. Conclusão**

Os resultados do teste demonstram que o funcionamento do programa está correto, atingindo assim o objetivo proposto. O programa desenvolvido ajudará os pesquisadores em suas análises e demonstrações dessas análises. Um proposta de trabalho futuro seria o desenvolvimento do mesmo programa no paradigma orientado a objetos para uma comparação entres as duas abordagens.

## **Referências**

- Hirama, K. (2012). *Engenharia de Software: Qualidade e Produtividade com Tecnologia*. Pearson Prentice Hall.
- Medina, M. and Fertig, C. (2005). *Algoritmos e Programação*. Novatec.
- Schlomer, N. (2009). Bibtex: How to cite a website. [http://nschloe.blogspot.com.br/2009/06/bibtex-how-to-cite-website\\_21.html](http://nschloe.blogspot.com.br/2009/06/bibtex-how-to-cite-website_21.html).
- Silva, R. (2012). Mostrando código c++ no latex. <http://latexbr.blogspot.com.br/2012/07/dica-mostrando-codigo-c-java-etc-no.html>.
- Vasconcelos, P. (2011). Introdução ao latex beamer. <http://www.ncc.up.pt/~pbv/aulas/comtec/beamer.pdf>.
- Victorine, V. M. (2006). *Treinamento em Linguagem C++*. Pearson Prentice Hall.

Victorine, V. M. (2008). *Treinamento em Linguagem C*. Pearson Prentice Hall.

Werner (2013). How to change listing caption. <http://tex.stackexchange.com/questions/64839/how-to-change-listing-caption>.