Centro Universitário FEI

PÓS GRADUAÇÃO – ENGENHARIA ELÉTRICA

MESTRADO – IAAA

INTELIGÊNCIA ARTIFICIAL APLICADA

À AUTOMAÇÃO E ROBÓTICA

Programação Científica– PEL 216

Prof. Dr. Reinaldo Augusto da Costa Bianchi

MÉTODO DE MONTE CARLO

"Integração Numérica por Monte Carlo"

FLÁVIO INFANTI

Matrícula: 118.310-2

AULA 06 – 22/08/2019

1 OBJETO

Dar continuidade ao aprendizado da disciplina de Programação Científica e se aprofundar nas técnicas referentes à Programação Orientada a Objeto C++, com a realização de exercícios usando Integração Numérica, consolidando desta forma, os conhecimentos adquiridos durante aula realizada em 22/08/2019, com a implementação de um algoritmo para solucionar a integral de $\int_0^1 \frac{4}{1+x^2}dx$ e a integral de do objeto “toroide”.

2 INTRODUÇÃO

O Método de Monte Carlo foi definido por Halton (1970) como sendo uma técnica para representar a solução de um problema como um parâmetro de uma população hipotética e, que usa uma sequência aleatória de números para construir uma amostra da população da quais estimativas estatísticas desse parâmetro possam ser obtidas.

O primeiro trabalho com método de Monte Carlo foi em 1947 com Jon Von Neuman e Stanislaw Ulam. Conforme colocado em (ULAM J. VON NEUMANN, 1947) e posteriormente em (METROPOLIS, 1949), eles propuseram usar uma simulação computacional em uma parte do projeto Manhattan, na Segunda Guerra Mundial. No projeto de construção da bomba atômica, Ulam e Jon Von Neumann consideraram a possibilidade de utilizar o método, que envolvia a simulação direta de problemas de natureza probabilística relacionados com o coeficiente de difusão do nêutron em certos materiais.

O nome do método se originou pelo uso da aleatoriedade e da natureza repetitiva das atividades realizadas em cassinos em Monte Carlo, Mônaco. O método de Monte Carlo tem sido utilizado há bastante tempo como forma de obter aproximações numéricas de funções complexas. Estes métodos tipicamente envolvem a geração de observações de alguma distribuição de probabilidades e o uso da amostra obtida para aproximar a função de interesse. O método é também referido como simulação estocástica e é um método relativamente simples e fácil de implementar.

A seguir apresentaremos a aproximação de Monte Carlo para o cálculo de integrais unidimensionais e multidimensionais, com limites de integração para simulação de eventos discretos para resolver problemas de cálculo de integrais, onde os métodos de integração numérica são processos que realizam a aproximação de integrais definidas utilizando aproximações matemáticas aplicadas em um intervalo desejado, retornando desta forma, valores muito próximos do que seria obtido no calculo numérico.

3 PROPOSTA DE IMPLEMENTAÇÃO.

Como parte do desenvolvimento foi utilizado um notebook HP Dell i7 2620M que possui 2 núcleos e 4 threads para realização da aula. Nessa máquina foi instalado o LINUX versão UBUNTU 18,04 , bem como todos os programas necessários para realização do exercício solicitado em aula, como :

Criação de DVD com Ubuntu 18.04 LTS

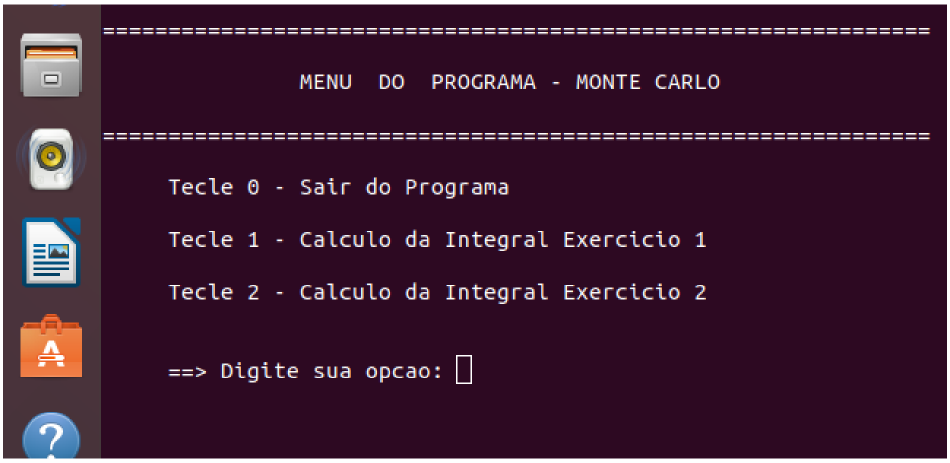
Editor gEdit

Ferramentas como g++, mpi

Code blocks

Portanto, o algoritmo desenvolvido apresenta a seguinte tela de inicialização conforme figura abaixo

Tela 1 - Tela Inicial do Programa



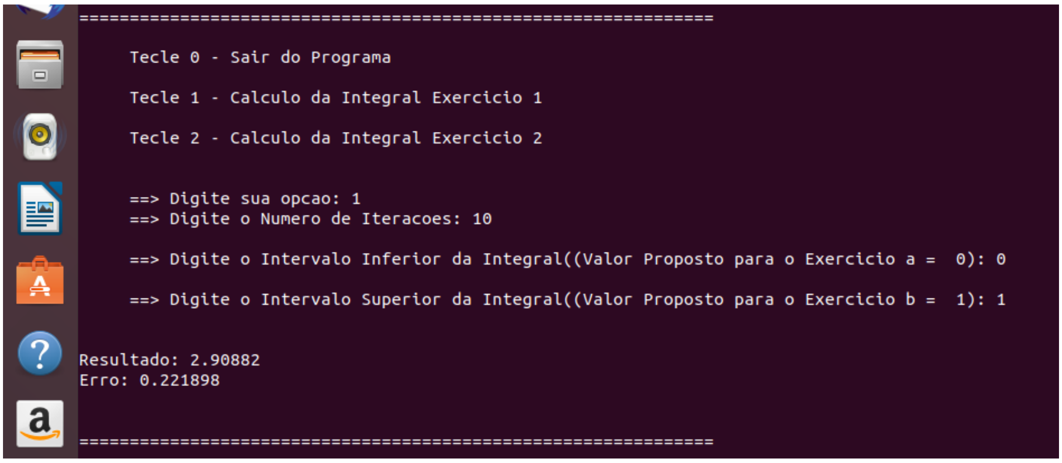
4 RESULTADOS

4.1 EXERCÍCIO 1

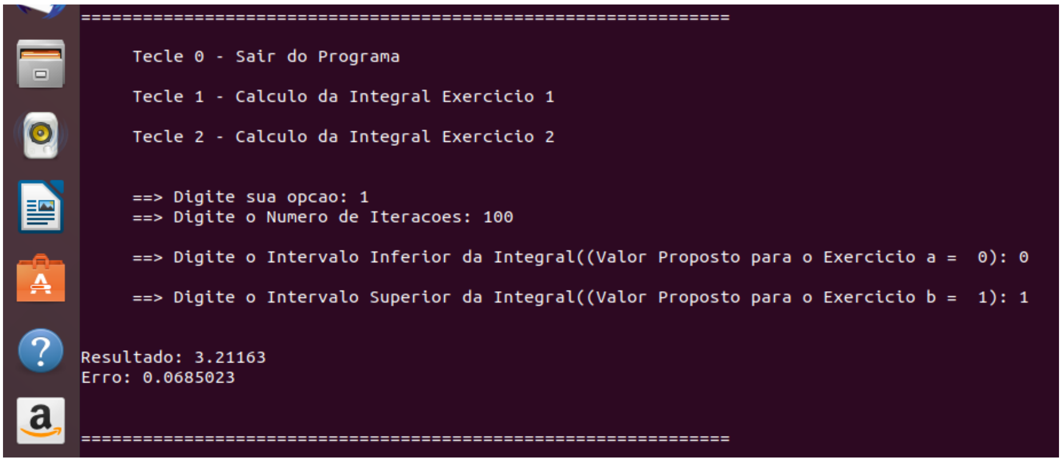
A seguir são apresentados os resultados das simulações realizadas pelo Método de Monte Carlo para solução da Integral $\int_0^1 \frac{4}{1+x^2} dx$.

De acordo com o Programa o usuário deverá escolher a opção 1 para executar os cálculos para solução da integral proposta, observado que também deverão ser fornecidos mais alguns parâmetros como o número de iterações e os limites inferior e superior da integral.

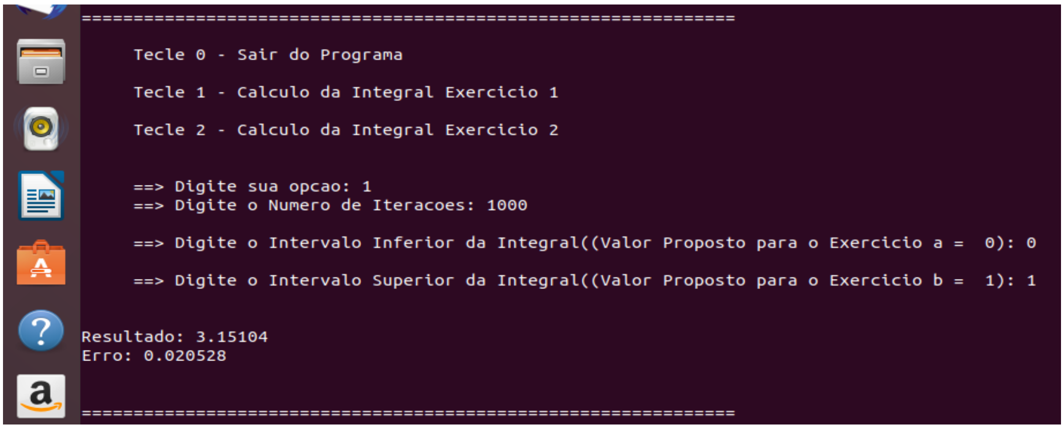
Exercício 1. 1 - Número de Iterações = 10



Exercício 1. 2 - Número de Iterações = 100

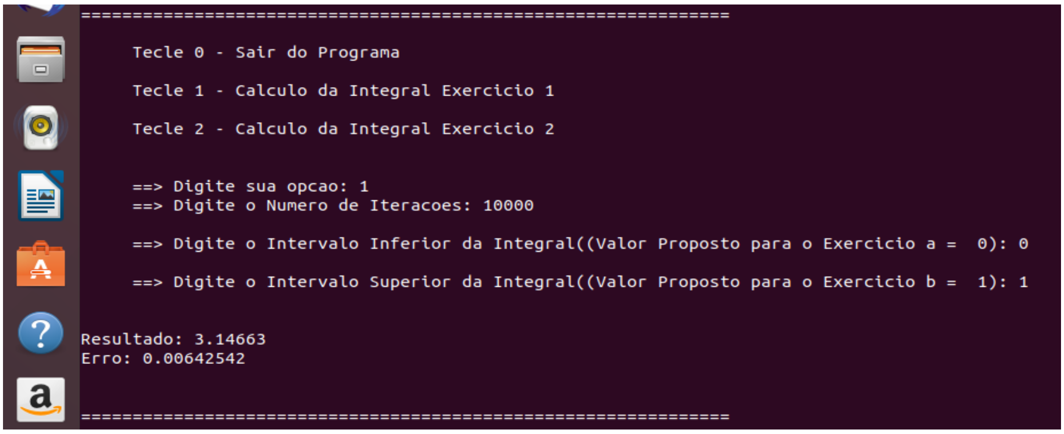


Exercício 1. 3 - Número de Iterações = 1.000



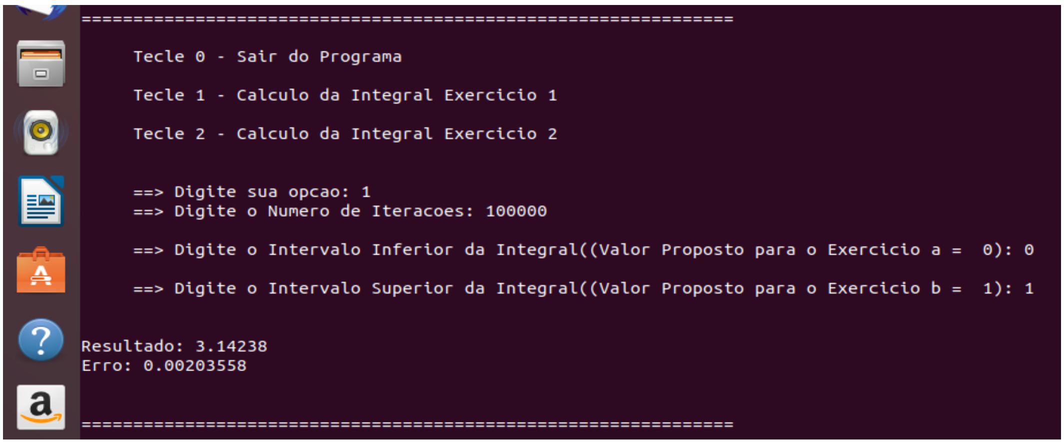
```
===============================================================

     Tecle 0 - Sair do Programa

     Tecle 1 - Calculo da Integral Exercicio 1

     Tecle 2 - Calculo da Integral Exercicio 2


     ==> Digite sua opcao: 1
     ==> Digite o Numero de Iteracoes: 1000

     ==> Digite o Intervalo Inferior da Integral((Valor Proposto para o Exercicio a =  0): 0

     ==> Digite o Intervalo Superior da Integral((Valor Proposto para o Exercicio b =  1): 1


Resultado: 3.15104
Erro: 0.020528


===============================================================
```

Exercício 1. 4 - Número de Iterações = 10.000



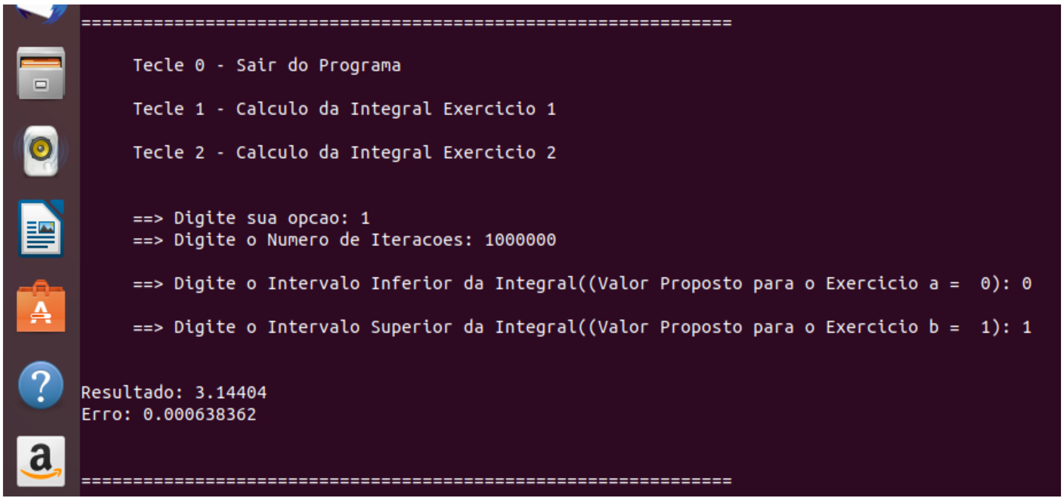
```
===============================================================

     Tecle 0 - Sair do Programa

     Tecle 1 - Calculo da Integral Exercicio 1

     Tecle 2 - Calculo da Integral Exercicio 2


     ==> Digite sua opcao: 1
     ==> Digite o Numero de Iteracoes: 10000

     ==> Digite o Intervalo Inferior da Integral((Valor Proposto para o Exercicio a =  0): 0

     ==> Digite o Intervalo Superior da Integral((Valor Proposto para o Exercicio b =  1): 1


Resultado: 3.14663
Erro: 0.00642542


===============================================================
```

Exercício 1. 5 - Número de Iterações = 100.000



```
===============================================================

     Tecle 0 - Sair do Programa

     Tecle 1 - Calculo da Integral Exercicio 1

     Tecle 2 - Calculo da Integral Exercicio 2


     ==> Digite sua opcao: 1
     ==> Digite o Numero de Iteracoes: 100000

     ==> Digite o Intervalo Inferior da Integral((Valor Proposto para o Exercicio a =  0): 0

     ==> Digite o Intervalo Superior da Integral((Valor Proposto para o Exercicio b =  1): 1


Resultado: 3.14238
Erro: 0.00203558


===============================================================
```

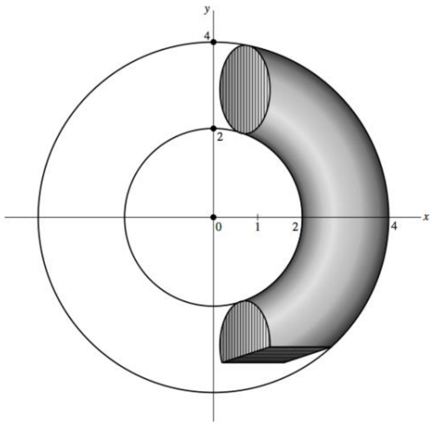
Exercício 1. 6 - Número de Iterações = 1.000.000



Número de Iterações	**Resultado**	**Erro**
10	2,90882	0,221898
100	3,21163	0,068502
1.000	3,15104	0,020528
10.000	3,14663	0,006425
100.000	3,14238	0,002035
1.000.000	3,14404	0,000638

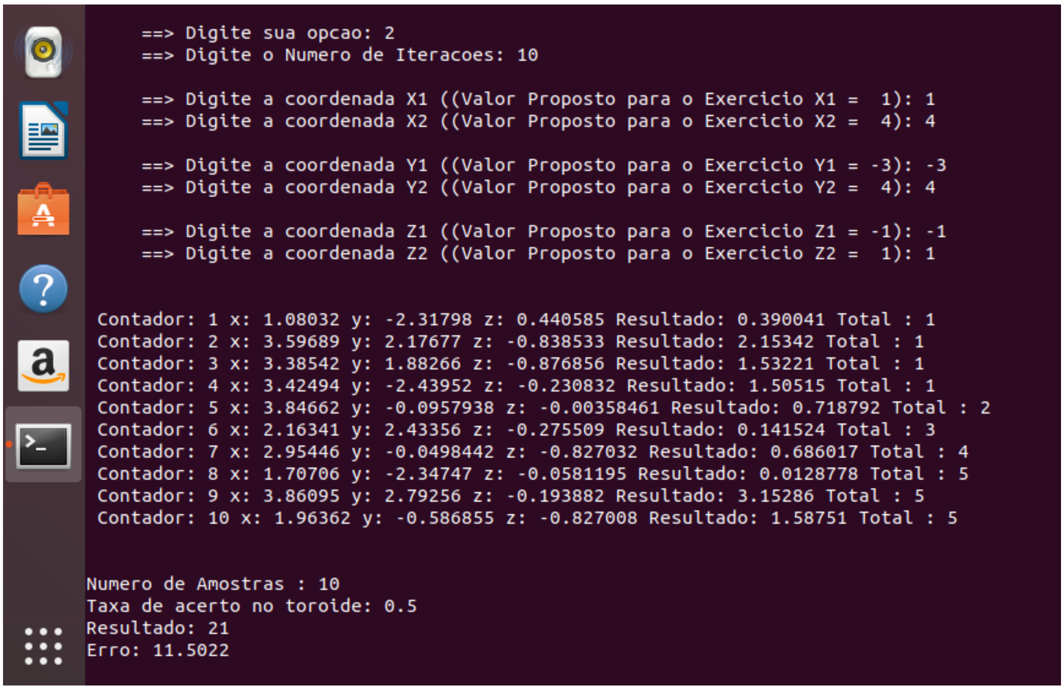
4.2 Exercício 2

A seguir são apresentados os resultados das simulações realizadas pelo Método de Monte Carlo para solução da Integral volume do objeto apresentado na figura abaixo :



De acordo com o Programa o usuário deverá escolher a opção 2 para executar os cálculos para solução da integral proposta, observado que também deverão ser fornecidos mais alguns parâmetros como o número de iterações, as coordenadas x1, x2, y1, y2, z1 e z2 que farão parte integrante dos cálculos da solução.

Exercício 2. 1 - Número de Iterações = 10



```
==> Digite sua opcao: 2
==> Digite o Numero de Iteracoes: 10

==> Digite a coordenada X1 ((Valor Proposto para o Exercicio X1 =  1): 1
==> Digite a coordenada X2 ((Valor Proposto para o Exercicio X2 =  4): 4

==> Digite a coordenada Y1 ((Valor Proposto para o Exercicio Y1 = -3): -3
==> Digite a coordenada Y2 ((Valor Proposto para o Exercicio Y2 =  4): 4

==> Digite a coordenada Z1 ((Valor Proposto para o Exercicio Z1 = -1): -1
==> Digite a coordenada Z2 ((Valor Proposto para o Exercicio Z2 =  1): 1


Contador: 1 x: 1.08032 y: -2.31798 z: 0.440585 Resultado: 0.390041 Total : 1
Contador: 2 x: 3.59689 y: 2.17677 z: -0.838533 Resultado: 2.15342 Total : 1
Contador: 3 x: 3.38542 y: 1.88266 z: -0.876856 Resultado: 1.53221 Total : 1
Contador: 4 x: 3.42494 y: -2.43952 z: -0.230832 Resultado: 1.50515 Total : 1
Contador: 5 x: 3.84662 y: -0.0957938 z: -0.00358461 Resultado: 0.718792 Total : 2
Contador: 6 x: 2.16341 y: 2.43356 z: -0.275509 Resultado: 0.141524 Total : 3
Contador: 7 x: 2.95446 y: -0.0498442 z: -0.827032 Resultado: 0.686017 Total : 4
Contador: 8 x: 1.70706 y: -2.34747 z: -0.0581195 Resultado: 0.0128778 Total : 5
Contador: 9 x: 3.86095 y: 2.79256 z: -0.193882 Resultado: 3.15286 Total : 5
Contador: 10 x: 1.96362 y: -0.586855 z: -0.827008 Resultado: 1.58751 Total : 5


Numero de Amostras : 10
Taxa de acerto no toroide: 0.5
Resultado: 21
Erro: 11.5022
```

Exercício 2. 2 - Número de Iterações = 100

```
Numero de Amostras : 100
Taxa de acerto no toroide: 0.63
Resultado: 26.46
Erro: 3.2617

===============================================================
```

Exercício 2. 3 - Número de Iterações = 1.000

```
Numero de Amostras : 1000
Taxa de acerto no toroide: 0.549
Resultado: 23.058
Erro: 1.1101

===============================================================
```

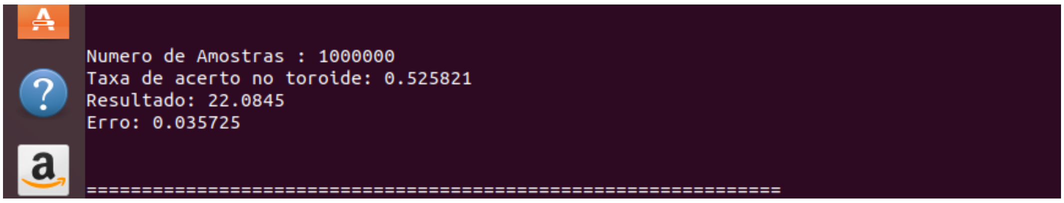
Exercício 2. 4 - Número de Iterações = 10.000

```
Numero de Amostras : 10000
Taxa de acerto no toroide: 0.53
Resultado: 22.26
Erro: 0.356159

===============================================================
```

Exercício 2. 5 - Número de Iterações = 100.000

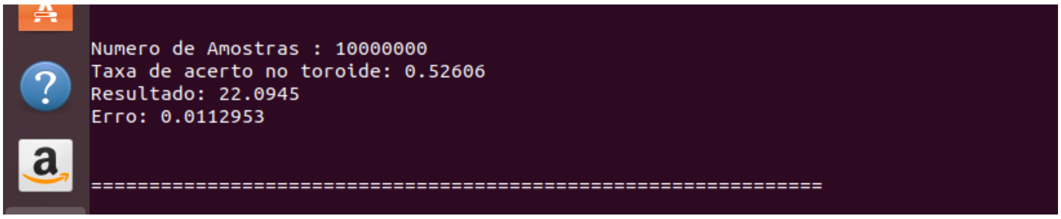
```
Numero de Amostras : 100000
Taxa de acerto no toroide: 0.52624
Resultado: 22.1021
Erro: 0.112938

===============================================================
```

Exercício 2. 6 - Número de Iterações = 1.000.000



```
Numero de Amostras : 1000000
Taxa de acerto no toroide: 0.525821
Resultado: 22.0845
Erro: 0.035725

=============================================================
```

Exercício 2. 7 - Número de Iterações = 10.000.000



```
Numero de Amostras : 10000000
Taxa de acerto no toroide: 0.52606
Resultado: 22.0945
Erro: 0.0112953

=============================================================
```

Número de Iterações	Taxa de Acerto	Resultado	Erro
10	0,5	21	11,5022
100	0,63	26,46	3,2617
1.000	0,549	23,058	1,1101
10.000	0,530	22,260	0,356159
100.000	0,52624	22,1021	0,112938
1.000.000	0,525821	22,0845	0,035725
10.000.000	0,52606	22,0945	0,011295

5 CONCLUSÕES

Como principal conclusão pode-se afirmar que o método de Monte Carlo vem auxiliar na solução de processos que necessitem utilizar de forma aleatória e repetitiva na obtenção de aproximações numéricas de funções complexas e de difícil solução.

6 REFERÊNCIS BIBLIOGRAFIA

- VETTERLING, W.T.; PRESS W.H.; TEUKOLSKY, S.A.;FLANNERY, B.P. – Numerical Recipes, The Art of Scientific Computing, third edition, Cambridge, 2007.
- CHAPRA, S.C. Applied Numerical Methods –MCGrawHill,2011
- ULAM J. VON NEUMANN, R. D. R. S. Statistical methods in neutron diffusion. LAMS-551, April 9 1947.
- METROPOLIS, S. U. N. The monte carlo method. Journal of the American Statistical Association, 1949.

ANEXO – ALGORITMO DO PROGRAMA – “MÉTODO MONTE CARLO”

```
//=== PEL216 - PROGRAMACAO CIENTIFICA
//=== Prof. Dr. Reinaldo Augusto da Costa Bianchi
//=== Aluno: Flavio Infanti nº 118.310-2
//=== AULA 06 - 22/08/2019
//===
//=== MÉTODOS NUMÉRICOS / INTEGRAÇÃO NUMÉRICA / MONTE CARLO

//=============================================================================
//==   BIBLIOTECAS
//=============================================================================

#include <iostream>
#include <random>
#include <chrono>
#include <math.h>
#include <stdio.h>
#include <ctime>

using namespace std;

// Utilizacao do mersenne twister 19937 para gerar numeros pseudo randomicos
float random(float init, float max)
{
		unsigned seed1 = std::chrono::system_clock::now().time_since_epoch().count();
		std::mt19937 generator(seed1);
		double random = (double)generator() / generator.max();
		return init + (double)random * (max - init);
}

// Calcula a Integral do Exercicio 1
float funcaoExercicio1(float x)
{
		return 4 / (1 + pow(x, 2));
}


// Calcula a Integral do Exercicio 2
float funcaoExercicio2(float x, float y, float z)
{
		return pow(z, 2) + pow(sqrt(pow(x, 2) + pow(y, 2)) - 3, 2);
}


 // Verifica se o resultado da funcao esta dentro do toroide
bool isToroide(float resultado)
{
		return resultado <= 1;
}


int main()
{
    while(1)
    {
       // Definicao das variaveis
       int numeroIteracoes = 1;
       int contador = 1;
       float total = 0;
       float totalquad = 0;
       float f2 = 0;
       float f = 0;
       int a = 0;
       int b = 0;
       float x1 = 0;
       float x2 = 0;
       float y1 = 0;
       float y2 = 0;
       float z1 = 0;
       float z2 = 0;
       int opcao_Regra = 0;


       // MENU DO PROGRAMA
          printf ("\n\n===============================================================\n\n");
```

```
printf ("                MENU  DO  PROGRAMA - MONTE CARLO                \n\n");
printf ("===============================================================\n\n");

printf ("     Tecle 0 - Sair do Programa \n\n");
printf ("     Tecle 1 - Calculo da Integral Exercicio 1\n\n");
printf ("     Tecle 2 - Calculo da Integral Exercicio 2\n\n\n");
printf ("     ==> Digite sua opcao: ");
scanf("%i", &opcao_Regra);


switch(opcao_Regra)
{
case 0:
{
      system("clear");
      cout<<"  Programa do Calculo do Toroide Finalizado !.\n\n"<<endl;
      exit(1);
      break;
}

case 1:
{
   printf ("     ==> Digite o Numero de Iteracoes: ");
   scanf("%d", &numeroIteracoes);
   printf ("\n     ==> Digite o Intervalo Inferior da Integral((Valor Proposto para o Exercicio a =  0): ");
   scanf("%d", &a);
   printf ("\n     ==> Digite o Intervalo Superior da Integral((Valor Proposto para o Exercicio b =  1): ");
   scanf("%d", &b);
   cout << endl;
   cout << endl;

   for (; contador <= numeroIteracoes; contador++)
   { // calcula amostras e soma nos acumuladores
      float fx = funcaoExercicio1(random(a, b));
      total += fx;
      totalquad += pow(fx, 2);
   }

   float f = total / numeroIteracoes; //calcula amostra mÃ©dia
   float f2 = totalquad / numeroIteracoes; // calcula amostra quadrada mÃ©dia


   float erro = (b - a) * sqrt((f2 - (float)pow(f, 2)) / numeroIteracoes); //calcula erro estimado
   //mostra na tela valor da integral e erro estimado
   cout << "Resultado: " << (b - a) * f << endl;
   cout << "Erro: " << erro << endl;
   cin.get();
   break;

}

case 2:

   printf ("     ==> Digite o Numero de Iteracoes: ");
   scanf("%d", &numeroIteracoes);

   printf ("\n     ==> Digite a coordenada X1 ((Valor Proposto para o Exercicio X1 =  1): ");
   scanf("%f", &x1);
   printf ("     ==> Digite a coordenada X2 ((Valor Proposto para o Exercicio X2 =  4): ");
   scanf("%f", &x2);

   printf ("\n     ==> Digite a coordenada Y1 ((Valor Proposto para o Exercicio Y1 = -3): ");
   scanf("%f", &y1);
   printf ("     ==> Digite a coordenada Y2 ((Valor Proposto para o Exercicio Y2 =  4): ");
   scanf("%f", &y2);

   printf ("\n     ==> Digite a coordenada Z1 ((Valor Proposto para o Exercicio Z1 = -1): ");
   scanf("%f", &z1);
   printf ("     ==> Digite a coordenada Z2 ((Valor Proposto para o Exercicio Z2 =  1): ");
   scanf("%f", &z2);

   cout << endl;
   cout << endl;

   //Calcula n numeros de amostras com entradas x y e x randomicas
   for (; contador <= numeroIteracoes; contador++)
   {
```

```
            float x = random(x1, x2);
            float y = random(y1, y2);
            float z = random(z1, z2);
            float resultado = funcaoExercicio2(x, y, z);

            // Verifica se o Resultado esta no toroide, Se sim soma 1 no total
            if (isToroide(resultado))
                total += 1;
                f2 += pow(1, 2);

            cout << " Contador: "<< contador << " x: "<< x << " y: "<< y << " z: "<< z << " Resultado: "<< resultado << " Total : "<< total <<
endl;
        }


        f2 = f2 / numeroIteracoes; // calcula media das amostras quadradas
        f = total / numeroIteracoes; // calcula media de acertos dentro do toroide das amostras
        float V = (x2 - x1) * (y2 - y1) * (z2 - z1); //calcula volume de integracao
        float erro = V * sqrt((f2 - (float)pow(f, 2)) / numeroIteracoes); // calcula erro estimado

        //mostra na tela taxa de acerto dentro do toroide, resultado da integracao e erro estimado
        cout << "\n\nNumero de Amostras : "<< numeroIteracoes << endl;
        cout << "Taxa de acerto no toroide: "<< f << endl;
        cout << "Resultado: " << V * (total / numeroIteracoes) << endl;
        cout << "Erro: " << erro << endl;

        cin.get();
      }
  }
}
```