UNIVERSIDADE DE SÃO PAULO
Instituto de Física de São Carlos

Projeto 01

Introdução ao Fortran

Jefter Santiago Mares
n° USP:12559016

22 de agosto de 2022

Conteúdo

São Carlos
2021

Tarefa 1

- A área do tórus é dada por: $A = (2\pi R)(2\pi r)$
- O volume do tórus é dado por: $V = (\pi r^2)(2\pi R)$

Onde R é o raio externo e r é o raio interno.

```
!     Tarefa 01
!     Calcula área e volume de um tórus a partir de raios dados.
      write(*,*) "Digite os valores dos raios (interno, externo):"
      read(*,*) ri, re
      pi = acos(-1e0)

      aArea = 4.e0 * pi ** 2 * re * ri
      aVolume = 2.e0 * pi ** 3 * re * ri

      write(*,*) "Area = ", aArea
      write(*,*) "Volume = ", aVolume
      end
```

Tarefa 2

2.1 Explicação

Sejam $u = (x_1, y_1, z_1)$, $v = (x_2, y_2, z_2)$ e $w = (x_3, y_3, z_3)$. Quero calcular a área lateral e volume do paralelepipedo formado pelos vetores u, v e $r = w - v = (x_3 - x_2, y_3 - y_2, z_3 - z_2)$. Para o cálculo da área temos

$$A_L = 2 \cdot \langle u, r \rangle + \langle v, r \rangle = 2 \langle u + v, r \rangle$$

$$\boxed{A_L = 2 \left[(x_3 - x_2)(x_1 + x_2) + (y_3 - y_2)(y_1 + y_2) + (z_3 + z_2)(z_1 + z_2) \right]}$$

Para o cálculo do volumes usamos o produto misto entre u, v e r, portanto $V = [u, v, r]$, desenvolvendo a operação pelo determinante chegamos à

$$\boxed{V = (x_3 - x_2)(y_1 z_2 - y_2 z_1) + (y_3 - y_2)(x_2 z_1 - x_1 z_2) + (z_3 - z_2)(x_1 y_2 - x_2 y_1)}$$

2.2 Código

```
!     Tarefa 02
!     Dados 3 vetores (u, v, w) calcula o volume do paralelepipedo das arestas
!     definidas por u, v e w - v.
      dimension u(1:3), v(1:3), w(1:3), r(1:3)
      write(*,*)"Digite as coordenadas de cada vetor"
      read(*,*) u(1), u(2), u(3)
      read(*,*) v(1), v(2), v(3)
      read(*,*) w(1), w(2), w(3)
      r(1) = w(1) - v(1)
```

```
      r(2) = w(2) - v(2)
      r(3) = w(3) - v(3)

!     A = 2[(wx - vx)*(ux + vx) + (wy - vy)*(uy + vy) + (wz - vz)*(uz + vz)]
      area =
     +     2 * (((w(1) - v(1)) * (u(1) + v(1))) +
     +     ((w(2) - v(2))  * (u(2) + v(2))) +
     +     ((w(3) - v(3))  * (u(3) + v(3))))
      write(*,*) "Área do paralelepipedo: ", area

      volume =
     +     (w(1) - v(1)) * (u(1)*v(3) - v(2)*u(3)) +
     +     (w(2) - v(2)) * (v(1)*u(3) - u(1)*v(2)) +
     +     (w(3) - v(3)) * (u(1)*v(2) - v(1)*u(2))

      write(*,*) "Volume do paralelepipedo: ", volume
      end
```

Tarefa 3

Foi definido como valor máximo de entrada $N = 1000$, ou seja, a maior quantidade possível de valores em um arquivo deve ser 1000. O input M deverá ser de tal modo que $M \leq N$. Primeiramente implementei a rotina abaixo, que gera um arquivo de nome `in_tarefa3.dat`, que contém os N números aleatórios dos quais M deles serão ordenados.

```
!     Gera o arquivo "in_tarefa3.dat" com N números aleatórios.
    subroutine generate_file(N)
    in = 10 ! Arquivo de entrada
    open(in, file='in_tarefa3.dat')
    r = 0
    do i = 1, N
       call random_number(r)
       write(in, *) r
    end do
    close(in)
    end
```

Na sequencia temos o código principal do programa, que primeiro rebece os inputs necessários, ou seja, as quantidades de números aleatórios e a serem ordenados, **faz a leitura do arquivo** `in_tarefa3.dat` e após isso faz a ordenação dos M primeiros números da lista e escreve-os no arquivo `out_tarefa3.dat`.

```
!     Tarefa 03
!     Cria uma lista de N números aleatórios
!     armazena essa lista num arquivo
!     e ordena os M primeiros números dessa lista.

      real*4 numbers(1:1000)

      in = 10 ! unidade para arquivo de entrada.
```

```
      iout = 20 ! unidade para arquvio de saída

      write(*,*) "Quantidade de números aleatórios (N<=1000):"
      read(*,*) N

      write(*,*) "Quantidade de números a serem ordenados (M<=1000):"
      read(*,*) M

      ! Gera arquivo com N números aleatórios.
      call generate_file(N)

      ! Lê o arquivo com os N números aleatórios
      ! e armazena os M primeiros números no vetor numbers.
      open(in, file="in_tarefa3.dat")
      do i = 1, M
         read(in, *) numbers(i)
      end do
      close(in)

!     Implementação do algoritmo insertion sort.
      do i = 1, M
         j = i
         do while (j .gt. 0 .and. (numbers(j) .lt. numbers(j-1)))
            temp = numbers(j)
            numbers(j) = numbers(j-1)
            numbers(j-1) = temp
            j = j - 1
         end do
      end do

      write(*,*) "Números ordenados."
      open(iout, file="out_tarefa3.dat")
      do i = 1, M
         write(iout,*) numbers(i)
      end do
      write(iout,*) M, "Numeros."
      close(iout)
      end
```

Tarefa 4

Podemos escrever a série

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

como a série de potências

$$\cos x = \sum_{n=0}^{k} (-1)^n \cdot \frac{x^{2n}}{(2n)!}$$

Primeiramente, foi implementado uma função recursiva que realiza o cálculo fatorial presente na fórmula acima.

```
recursive function factorial (n) result (res)
if(n .eq. 1) then
   res = 1
else
   res = n * factorial(n-1)
end if
end function factorial
```

Tarefa 8 ou 9

$$\boxed{V_d = \frac{\pi^{\frac{d}{2}}}{\Gamma\left(\frac{d}{2}+1\right)} R^d}$$

$$\Gamma(1/2) = \sqrt{\pi}, \Gamma(1) = 1, \Gamma(x+1) = x\Gamma(x)$$

```
#define PI 3.141592659
int gamma(int d){
    float x = d/2;
    if(x == 1) return 1;
    if(x == 1/2) return sqrt(PI);
    return x * gamma(x - 1);
}
```