Métodos Computacionais I

Linguagem de programação: C

Aplicativo(s) usado(s): CodeBlocks, GDB Online Compiler

Arelienne Siarelly Silva Santos - areliennesantos@fisica.ufmt.br
Carlos Henrique Porto Carvalho - carloscarvalho@fisica.ufmt.br
GitHub: `https://github.com/Carl05-beepboop`
Daniel Caldas Teixeira Silva - danielcaldas@fisica.ufmt.br

Introdução

Esse documento foi construído para que facilite o estudo da disciplina Métodos Computacionais I que tem como principal material os itens citados na bibliografia. Nesse caso, aqui, serão colocados resumos/sínteses dos textos apresentados no livro citado anteriormente, resolução de problemas, argumentos, discussões, questionamentos (e suas respostas) etc. Além disso, todos os materiais utilizados estarão presentes nas referências, para possibilitar o aprofundamento dos demais discentes que estejam interessados em conteúdos completos, não apenas resumos.

Sumário

1 Capítulo

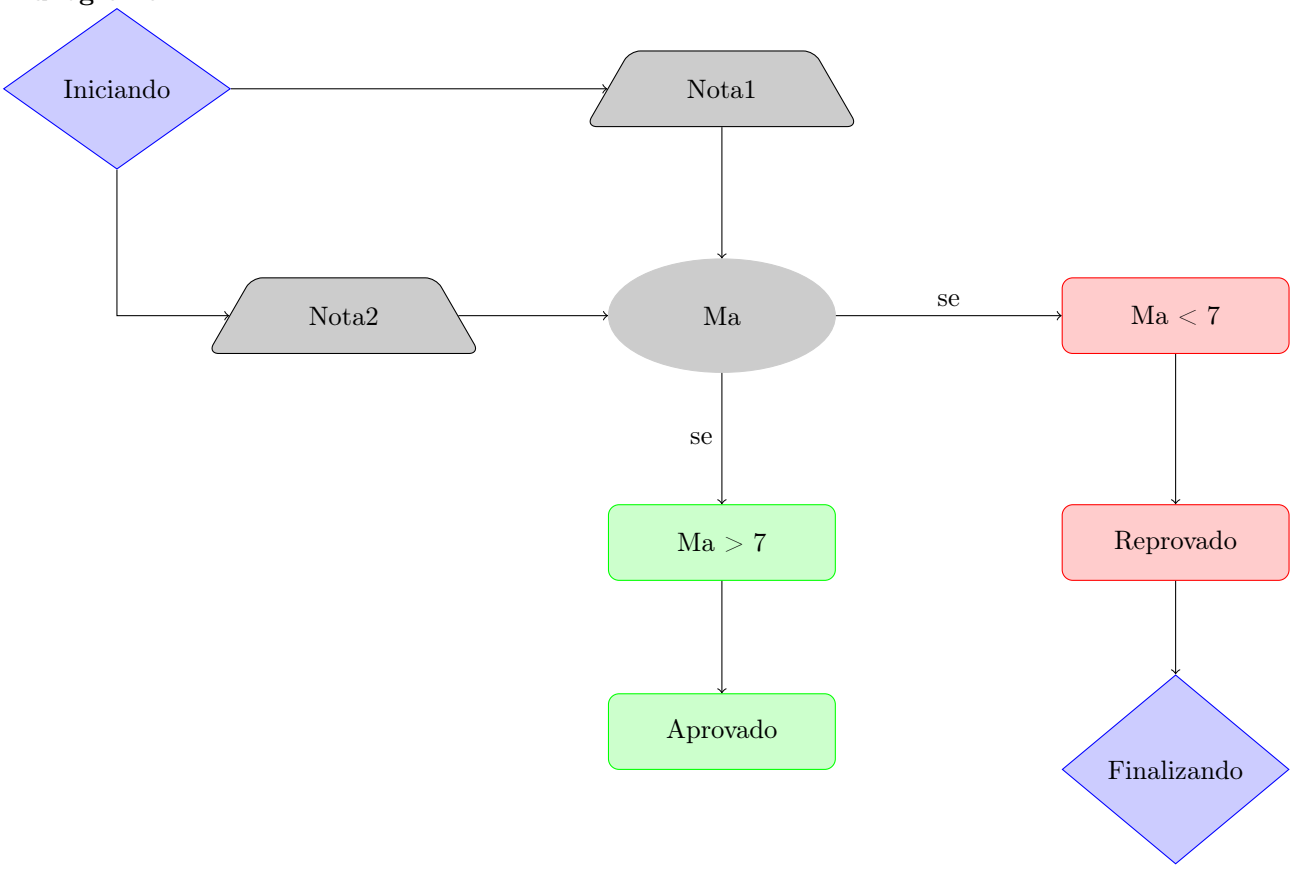
Tópicos tratados: Algoritmos.

Os Algoritmos podem ser, em sua explicação mais simples, um conjunto de passos seguidos capazes de instruir a realização de tarefas utilizando diferentes meios e métodos (e podem existir

em diversos âmbitos em formatos diferentes); conjunto este, pode ser classificado como sequencial (em formato de sequências), finitos (isso é, precisa-se determinar um início e fim), muito bem ordenados e claros (não pode haver duplos sentidos/ambiguidades). Quando focamos na área computacional, os Algoritmos passam a possuírem as seguintes características: definição (devem ser bem definidos, claros e não-ambíguos), finitude (possuem um número finito de passos), efetividade (operações básicas possibilitando sua execução de forma exata e em tempo finito), entradas (devem possuir zero ou mais quantidades que são processados pelos algoritmos durante a execução dos passos) e saídas (devem possuir um ou mais resultados do trabalhado e realizado pelos algoritmos). Além disso, os Algoritmos podem ser representados através de descrição narrativa (em linguagem natural - e.g., português, inglês -), fluxograma (uso de formas geométricas padronizadas para descrever passos) e linguagem algorítmica (a linguagem na qual um computador é capaz de compreender, que consiste em empregos de uma linguagem intermediária entre a natural e a de programação).

1.1 Exemplos

Fluxograma



Linguagem Computacional (C)
observação: não é a algorítmica!

```
#include <stdio.h>

    int main ()
```

```
4
5     {
6
7  /* ----- variáveis ----- */
8      float nota1, nota2, nota3, n, media;
9
10
11 /* ----- entradas ----- */
12
13     printf("Digite sua primeira nota: ");
14        scanf("%f", &nota1);
15
16     printf("Digite sua segunda nota: ");
17        scanf("%f", &nota2);
18
19     printf("Digite sua terceira nota: ");
20        scanf("%f", &nota3);
21
22
23
24 /* ----- definições ----- */
25
26     n = 3;
27     media = (nota1 + nota2 + nota3)/n;
28
29 /* ----- saídas ----- */
30
31     printf("Média = %.2f\n", media);
32
33    if (media >= 7)
34     printf("Aprovado (AP)");
35
36    if (media <= 5)
37     printf("Reprovado (RE)");
38
39    if (media > 5.99 && media < 7)
40     printf("Prova Final (PF)");
41
42  return 0;
43
44    }
45
```

1.2 Exercícios

1. Explique, com suas próprias palavras, o que é algoritmo.

Da forma mais reduzida, pode ser considerado como um conjunto de instruções/passos que possibilitem o acontecimento/a realização de uma determinada tarefa. Como no texto acima, algoritmos possuem características bem definidas e podem se apresentar de formas diferentes.

2. Rotineiramente, usamos algoritmos para as mais diversas tarefas. Cite três algoritmos que podemos encontrar no dia a dia.

Ao acordar, seguimos um conjunto de passos para ir trabalhar/estudar, por exemplo; no trabalho/ambiente de estudo, há algoritmos que você também precisa seguir para concluir suas tarefas; ao finalizar todas as suas atividades do dia, você precisa seguir um algoritmo para retornar para a sua casa, e, ainda, quando chegar precisará seguir mais algoritmos para realizar suas últimas atividades do dia e ir dormir.

3. Em que consiste a característica de efetividade de um algoritmo?

Os algoritmos precisam ter linguagens/operações básicas de modo que isso possibilite sua execução de forma exata e em tempo finito.

4. Suponha que o quarto passo de um determinado algoritmo ordene que a execução retorne ao primeiro. Qual característica não está sendo satisfeita por esse algoritmo?

A definição do algoritmo não está sendo satisfeita. Ou seja, não há uma boa definição, clareza e elementos que não sejam ambíguos (não tenham duplo sentido); fazendo com que também não tenha havido, nesse algoritmo, uma característica de saída (não gerou resultados), como também não foi efetivo etc.

5. Discorra sobre as formas de representação de algoritmos mais comuns, destacando suas vantagens e desvantagens.

I. Descrição narrativa: O fato de usar a língua materna de quem está produzindo o algoritmo pode ser uma facilidade e portanto um ponto positivo, mas pode se tornar negativo porque pode haver ambiguidade de forma frequente nesse tipo de linguagem, mais dificuldade de entendimento e se afastar muito da linguagem de um computador;

II. Fluxograma: Facilmente compreendido, mas na medida em que o número de fluxogramas aumentam, pode dificultar a construção, entendimento e visualização;

III. Linguagem algoritmica: Pode ser facilmente traduzida para uma linguagem de programação, mas que esta linguagem de programação pode ser restrita a um conjunto de instruções no qual o funcionamento depende da arquitetura do hardware.

6. Suponha que você foi premiado com um robô capaz de auxiliá-lo nas tarefas domésticas. Antes que execute determinada atividade, você precisa instruí-lo corretamente através de um algoritmo específico. Sabendo disso, escreva algoritmos, em linguagem natural, para ensiná-lo a realizar cada uma das tarefas abaixo:

Trocar a lâmpada do seu quarto.

```
    /* ALGORITMO */
    Apague a luz;
Pegue uma lampada que possa substituir a antiga e coloque por perto;
Identifique a lampada a ser trocada;
Use as maos para segurar a base do soquete onde a lampada esta inserida;
Segure a base do soquete com uma das maos;
Com a outra segure cuidadosamente a lampada;
Gire levemente no sentido anti horario ate desenroscar a lampada;
Quando desenroscada, coloque no chao;
Pegar a lampada nova;
Colocar a lampada no soquete;
Girar a lampada suavemente no sentido horario ate estar bem enroscada;
    Acenda a luz.

```

Trocar o pneu do seu carro.

```
/* ALGORITMO */
Pegar um pneu reserva do carro;
Pegar um macaco;
Usar o macaco para suspender o carro;
Desenrosque os parafusos;
Tire o pneu antigo;
Coloque o pneu novo e encaixe firmemente;
Pegue os parafusos;
Enrosque os parafusos;
Desca o carro com o uso do macaco;
Retire o macaco.

```

Fazer uma vitamina de banana com açaí.

```
/* ALGORITMO */
Pegue leite de castanha de caju (evitar o leite animal);
Pegue acai ja batido;
Pegue banana;
Pegue um liquitificador;
Pegue uma faca;
Corte a banana em rodelas;
Pegue um copo com capacidade de 500ml;
Despeje o leite no copo ate encher;
Despeje o leite do copo no liquitificador;
Despeje o acai batido no liquitificador;
Pegue uma colher comum de acucar;
Despeje a colher de acucar no liquitificador;
Ligue o liquitificador;
Espere bater durante 30 segundos;
Desligue o liquitificador.

```

Lavar e secar os pratos.

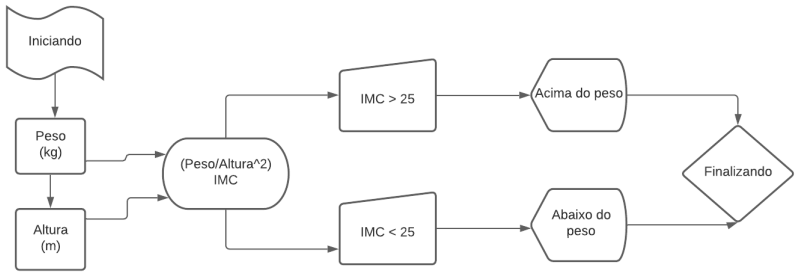
```
/* ALGORITMO */
Va ate a pia;
Pegue o detergente;
Pegue a esponja com o bombril;
Despeje o detergente na esponja;
Pegue um prato;
Esfregue a esponja no prato para limpar a sujeira;
Faca esse processo com todos os pratos;
Ligue a torneira;
Jogue agua em todos os pratos para tirar o sabao;
Desligue a torneira;
Pegue um prato;
Pegue um pano limpo e seco;
Passe o pano no prato ate que ele absorva toda a agua;
Repita o processo com todos os pratos.

```

Calcular quanto você precisar tirar na terceira nota para passar por média em Introdução à Programação.

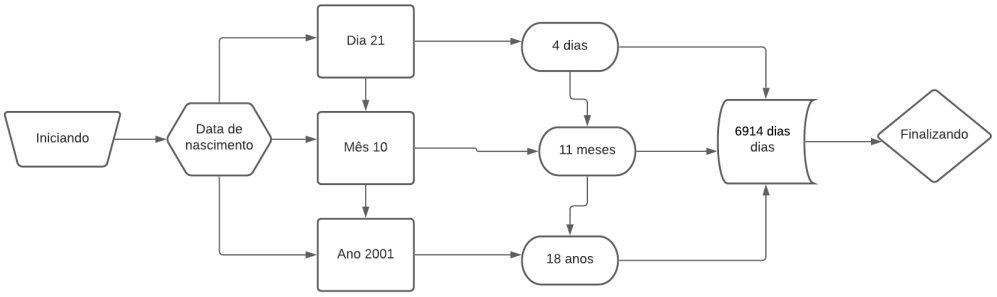
```
/* ALGORITMO */
/* MEDIA 7 E 3 PROVAS */
Somar a media 7 de tres provas (21);
```

```
Dividir a soma por 3;
Perceber que a media da materia e 7;
Pegar a minha primeira nota (7);
Pegar a minha segunda nota (7);
Somar as duas notas (14);
Subtrair 21 - 14;
Concluir que falta 7 pontos para a aprovacao da disciplina.

```

7. Escreva um algoritmo, utilizando fluxograma, que receba como entrada o peso e altura de uma pessoa, calcule seu IMC (Índice de Massa Corpórea) e exiba sua situação, segundo os seguinte critério: Se o $IMC > 25$, a pessoa está acima de seu peso, caso contrário, está abaixo. Onde o $IMC = (Peso)/(Altura^2)$



8. Usando fluxograma, faça um algoritmo que receba como entrada a idade de uma pessoa expressa em anos, meses e dias (Atenção: são 3 entradas) e mostre-a expressa apenas em dias. Considere anos de 365 dias e meses de 30 dias.



2 Capítulo

Tópicos tratados: Introdução à Programação.

Em construção...

2.1 Exemplos

Em construção...

2.2 Exercícios

10. Qual é valor de (x1 + x2) após a execução dos grupos de comandos abaixo:

a. y = 6;

b. z = 8;

c. c = 2;

d. x1 = x1 =((y * z) - z)/c;

e. x2 =(z /2)/y++.

```
#include <stdio.h>

int main ()

    {

   int  y, z, c;
   float x1, x2, resultado;

   y = 6;
   z = 8;
   c = 2;


   x1 = ((y * z) - z)/c;
   x2 = (z /2.0)/y++;
   resultado = (x1 + x2);


    printf ("x1 = %.2f\n", x1);
    printf ("x2 = %.2f\n", x2);
    printf ("(x1 + x2) = %.2f\n", resultado);


 return 0;

    }
```

11. Considerando as atribuições x = 20 e y = 2, calcule o resultado de cada uma das expressões abaixo:

a. (x–+ x * (x % y))

b. (x–+ x * (x % y))

c. (x–+ x * (x % 3))

d. (–x + x * (x % 3))

e. (–x + x * (x % x))

```c
#include <stdio.h>

int main ()
    {

    float a, b, c, d, e;
    int x, y;

   x = 20;
   y = 2;


   a = (x--+ x * (x % y));
   b = (x--+ x * (x % y));
   c = (x--+ x * (x % 3));
   d = (--x + x * (x % 3));
   e = (--x + x * (x % x));


    printf ("RESPOSTAS\n\n");
    printf ("a) %.2f \n", a);
    printf ("b) %.2f \n", b);
    printf ("c) %.2f \n", c);
    printf ("d) %.2f \n", d);
    printf ("e) %.2f \n", e);


    return 0;

    }
```

12/1. Faça um programa em C que solicite ao usuário que digite o ano de seu nascimento, armazene o valor digitado em uma variável e em seguida imprima na saída padrão a sua idade.

```c
#include <stdio.h>

int main()

    {

    int ano, idadena, idadea;


/* para não aniversariantes */
    printf("Digite o ano do seu nascimento: ");
        scanf("%i", &ano);

    idadena = (2020 - ano);
    printf("Caso já tenha feito aniversário, a sua idade é %i anos\n", idadena);

/* para aniversariantes */
   printf("Digite o ano do seu nascimento: ");
        scanf("%i", &ano);

    idadea = (2020 - ano - 1);
    printf("Caso não tenha feito aniversário, a sua idade é %i anos\n", idadea);

return 0;
```

```
    }
```

12/2. Faça um programa em C que solicite ao usuário que digite a data de nascimento (dia/mês/ano) e imprima todas as respostas em dias.

```
#include <stdio.h>

int main ()

    {

    int idadena1, idadea1, dia, mes, ano;
    float idade2;


 /* para não aniversariantes */

    printf("Digite aqui o dia do seu nascimento: ");
        scanf("%i", &dia);
    printf("Digite aqui o mês do seu nascimento: ");
        scanf("%i", &mes);

    printf("Digite aqui o ano do seu nascimento: ");
        scanf("%i", &ano);


    idadena1 = (2020 - ano - 1);
    printf("Caso não tenha feito aniversário, a sua idade em anos é %i\n", idadena1);
    idade2 = (dia + mes*30 + idadena1*365);
    printf("A sua idade em dias é aproximadamente: %.2f\n", idade2);

 /* para aniversariantes */

    printf("Digite aqui o dia do seu nascimento: ");
        scanf("%i", &dia);
    printf("Digite aqui o mês do seu nascimento: ");
        scanf("%i", &mes);

    printf("Digite aqui o ano do seu nascimento: ");
        scanf("%i", &ano);

    idadea1 = (2020 - ano);
    printf("Caso já tenha feito aniversário, a sua idade em anos é %i\n", idadea1);
    idade2 = (dia + mes*30 + idadea1*365);
    printf("A sua idade em dias é aproximadamente: %.2f\n", idade2);


    return 0;

    }
```

3 Capítulo

Tópicos tratados: Estruturas de Controle.

Em construção...

3.1 Exemplos

Em construção...

3.2 Exercícios em grupo

1. Escreva um programa que verifique se um número digitado pelo usuário é menor, igual ou maior que zero.

```
#include <stdio.h>

int main()
{
 //Variavéis
   float x;

 //Entrada de Dados
   printf("Digite um valor:");
   scanf("%f",&x);

 //Saída de Dados

 // x > 0
   if (x > 0)
   printf("%.2f > 0\n0 valor inserido é maior que zero.", x);

 // x < 0
   else
   {if (x < 0)
   printf("%.2f < 0\n0 valor inserido é menor que zero.", x);

 // x = 0
   else
   printf("%.2f = 0\n0 valor inserido é igual a zero.", x);
   }
    return 0;
}
```

2. Dado o algoritmo abaixo, explique o que acontece se o valor lido para a variável x for: 3, 1 e 0. Explique o porquê.

```
#include <stdio.h>

int main() {

    int x;

    scanf(&x);

    if (x) printf("verdadeiro");

    return 0;


```

```
#include <stdio.h>

int main()
```

```
{
   int main() {
   int x;
scanf("%x", &x);
if (x) printf("verdadeiro");
 // o programa lerá 1 e 3 como verdadeiros pq são numeros inteiros, já o 0 ele ignora pq é um
     numero real neutro.

    return 0;

}
```

3. Escreva um programa que informe se um dado ano é ou não bissexto. Obs.: um ano é bissexto se ele for divisível por 400 ou se ele for divisível por 4 e não por 100.

```
#include <stdio.h>

int main ()

    {

        int ano;

            printf ("Insira um ano: ");
                scanf ("%i", &ano);


    if (ano % 400 == 0)

   {

            printf("%i é um ano bissexto.", ano);

   }

   else if (ano % 100 == 0)

   {

            printf("%i não é um ano bissexto.", ano);

   }

   else if (ano % 4 == 0)

   {

            printf("%i é um ano bissexto.", ano);

   }

   else

   {

            printf("%i não  é um ano bissexto.", ano);
```

```
   }


    return 0;

    }
```

4. Escreva um programa que mostre todos os números pares no intervalo de 1 a 40 de forma decrescente, utilizando o comando while. Depois faça o mesmo, mas desta vez, utilizando o comando for.

```
#include <stdio.h>

int main()
{
    printf("mostrar numero pares no intervalo 1 ate 40 de forma decrescente\n");

    int x=40;
    while(x>=2)
   {

       printf("%d \n",x);
       x-=2; // x = x - 2

   }
   printf("___fim___\n");


    return 0;
}
```

5. Um determinado banco abriu uma linha de crédito para os funcionários públicos. Porém, o valor máximo da prestação não poderá ultrapassar 30% do salário deste funcionário. Faça um programa para ajudar este banco. O programa deve permitir o usuário entrar com o salário do funcionário e o valor da prestação e informar se o empréstimo pode ou não ser concedido.

```
#include <stdio.h>

int main()
{
   //Variáveis
    float sal; // Salário
    float pres; // Prestação
    float emp; // Empréstimo

  //Entrada
    printf("Digite o valor do seu salário:");
    scanf("%f", &sal);

    printf("Digite o valor pretendido da prestação:");
    scanf("%f", &pres);

  //Processamento
    emp = (sal * 30) / 100;

  //Saída
```

```
   if (emp >= pres) //Empréstimo Concedido
   printf("O valor máximo da prestação é: %.f\nPortanto, o empréstimo foi CONCEDIDO", emp);

   else // Empréstimo Negado
   printf("O valor máximo da prestação é: %.f\nPortanto, o empréstimo foi NEGADO", emp);

    return 0;
}
```

7. Faça três programas que mostrem de 1 a 10 na tela, utilizando, em cada um, uma estrutura de laço de repetição diferente.

```
#include <stdio.h>

 int main ()
    {

    int num;

 for (num = 1; num <= 10 ; num++)
        printf("%i\n", num);

 return 0;

    }
```

```
#include <stdio.h>

 int main ()
    {
    int num;

 do
 {
    num++;
        printf("%i\n", num);

 }

 while (num <= 9);

 return 0;

    }
```

```
#include <stdio.h>

 int main ()

 {
     int num;

   num = 1;
```

```
while (num <= 10)

{

  printf("%i\n", num);
  num++;


}

return 0;


}
```

8. Escreva um programa que mostre na tela os números múltiplos de 3 no intervalo de 2 a 100.

```
#include <stdio.h>

int main()
{
    printf("mostre os multiplos de 3 no intervalo 2 ate 100");

    int n;

    for(n=1;n<100;n=n+2)
     printf("/n%i",n);


    return 0;
}
```

9. Escreva um programa para ler dois números inteiros M e N e, a seguir, imprimir os números pares existentes no intervalo [M, N].

```
#include <stdio.h>

int main()
{
    //Variáveis
      int num1, num2, cont;

    //Entrada
      printf("Digite um número:");
      scanf("%d", &num1);

      printf("Digite outro número maior que o anteiror:");
      scanf("%d", &num2);

    //Processamento
    for (cont = num1; cont <= num2; cont++)

    {
    if (cont % 2 == 1) continue;
    //Saída
    printf("Número par entre o intervalo %d e %d é : %d\n",num1, num2, cont);
```

```
    }

    return 0;
}
```

11. O supermercado Excelente Preço está precisando ser informatizado. Neste sentido, o dono quer um programa que leia os preços dos produtos até que seja informado o valor zero. No final o programa deve informar o total da compra e perguntar a forma de pagamento. As opções da forma de pagamento são: 1) A vista; 2) No cartão de crédito. Se a opção escolhida for a vista, então o programa informa o valor da compra com um desconto de 5%. Caso a compra seja no cartão de crédito, o programa informa o valor da compra dividido em 4 vezes.

```
#include <stdio.h>

 int main ()
     {

    float preco;
    float total;
    int cont, fpagamento;
    float cartaodecredito, avista, avista2;

    preco++;
    cont++;

 while (preco != 0)

    {

        printf("Entre com o preço: ");
            scanf("%f", &preco);

 if (preco != 0)

    {

    total += preco;

    }

    }

        printf ("O valor total da compra é: %.2f\n", total);

    cartaodecredito = total/4;
    avista = total * 0.05;
    avista2 = total - avista;


        printf ("Escolha um número que represente a sua forma de pagamento:\n1. à vista; \n2. Cart
     ão de Crédito.\n");
            scanf("%i", &fpagamento);

    if (fpagamento <= 1)
        printf ("Você acaba de receber um desconto de cinco porcento e portanto terá que pagar um
     valor de: %.2f \n", avista2);

    else
```

```
        printf ("Você conseguiu um parcelamento de quatro vezes sem juros e terá que pagar quatro
     vezes de: %.2f \n", cartaodecredito);


 return 0;

    }
```

4 Problemas Físicos

4.1 Problemas Cap. 1 (Decaimento Exponencial)

1. Use o mapa iterativo (1.4), com $\Delta t = 1$s, e mostre que uma amostra de Rb82 terá sua radioatividade reduzida à metade depois de decorridos 75s (ou 1min e 15s).

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()

    {


    float N, N2, deltaN;
    float tau;
    float deltat;
    float lnN;
    int t0, t, T0, T;

    #define alfa 0.00924 /* constante do material rb82 */
    deltat = 1;
    tau = 0.0;


    printf("Entre com uma quantidade inicial de núcleos radioativos: ");
        scanf("%f", &N);

    printf("Entre com um tempo inicial: ");
        scanf("%i", &t0);

    printf("Entre com um tempo final: ");
        scanf("%i", &t);


    FILE *fp;

    fp = fopen("Dados teóricos", "w");

    fprintf(fp, "Resultados para plotagem gráfica:\n\nTempo [s] | Núcleos radioativos [N] | ln N(t
     ) \n");
    for (t0 = 0; t0 <= t; t0++)

    {
```

```
    lnN = log (N);
    fprintf(fp, "\n %i  %f  %f\n", t0, N, lnN);
    N = N * (1 - alfa * deltat);

    }

    fclose(fp);

/* ----------- */

    printf("\nPara o cálculo da vida média do material, entre com a mesma quantidade inicial de nú
     cleos radioativos: ");
        scanf("%f", &N2);

    printf("Agora entre com outro tempo inicial: ");
        scanf("%i", &T0);

    printf("Por último, entre com outro tempo final: ");
        scanf("%i", &T);

    FILE *fp2;

    fp2 = fopen("Vida média", "w");

    fprintf(fp2, "Tempo de vida média do material: \n\n Tempo [s] | N(t) | DeltaN \n");

    while (T0 < T)
    {

    T0++;
    deltaN = alfa * deltat * N2;
    fprintf(fp2, "\n %i %f %f \n", T0, N2, deltaN);
    N2 = N2 * (1 - alfa * deltat);
    tau = tau + deltaN * T0;

    }

    tau = tau/1000.0;
    printf("\n\nO resultado da somatória/média ponderada que diz o tempo de vida média do material
      é: %f s", tau);


    fclose(fp2);


    return 0;

    }
```

Tempo [s]	Núcleos radioativos [N]	ln N(t)
0	1000	6.90
5	954.65	6.86
10	911.35	6.81
20	830.56	6.72
30	756.93	6.62
40	689.82	6.53
50	628.67	6.44
60	572.94	6.35
70	522.15	6.25
75	498.47	6.21

2. Use de novo o mapa (1.4), com $\Delta t = 1s$, para o caso do Rb82, e construa um gráfico do núumero de núcleos ainda radioativos em função do tempo. Considere uma amostra com N(0) = 1000 núcleos radioativos iniciais (uma amostra real terá normalmente muito mais, porém o aspecto do gráfico será o mesmo). Para simplificar, assinale no gráfico apenas os valores obtidos a cada 10s.

3. Reconstrua o gráfico do problema anterior, com os mesmos dados numéricos, desta vez adotando escala logarítmica no eixo vertical. Proceda da seguinte maneira: em vez de assinalar no eixo vertical os valores de N(t) calculados para t = 0, 10, 20 . . . , assinale os valores de ln N(t), ou seja o logarítmo Neperiano de cada um dos valores N(t) já calculados.

Gráfico de Decaimento Radioativo

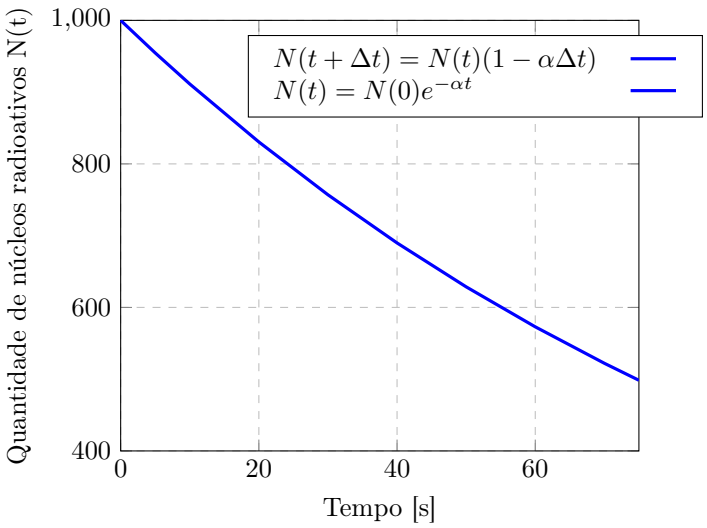
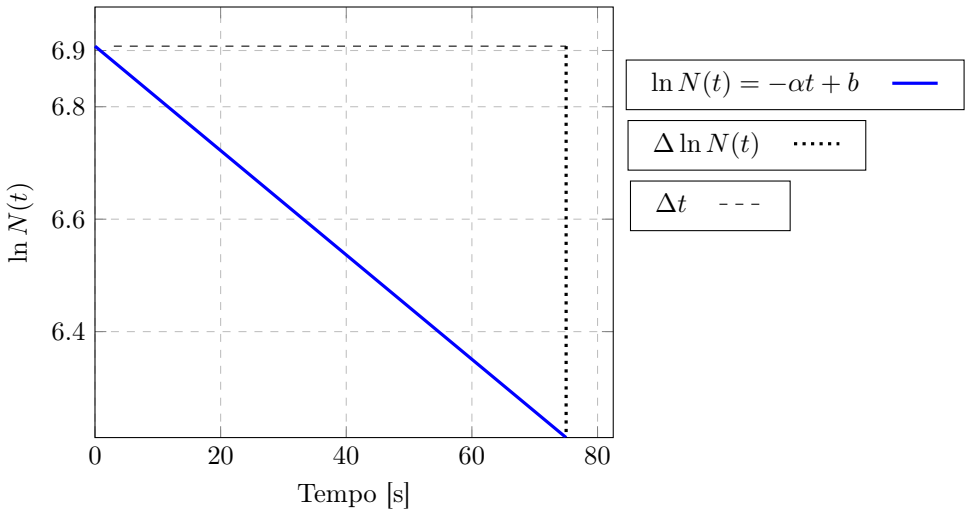


Gráfico do Logarítmo Neperiano do Decaimento Radioativo



4. Meça o coeficiente angular da reta obtida no gráfico anterior, compare com a constante α do Rb82, e mostre que a forma analítica que descreve o decaimento é: $N(t) = N(0)e^{-\alpha t}$

Primeiro, para o cálculo do coeficiente angular precisaremos voltar na tabela (ou voltar no programa caso queira valores mais precisos) e pegar os valores: $\ln N(t0)$, $\ln N(t)$, $t0$, t. Respectivamente: 6.90775, 6.21153, 0, 75. Como se trata de uma reta, uma função de cara $f(x) = ax + b$, sabe-se que o coeficiente angular (a) é determinado, no nosso caso, pela expressão:

$$a = \frac{\ln N(t) - \ln N(t0)}{t - t0} \tag{1}$$

Substituindo os valores citados, teremos que $a = -0.00928295$, negativo (como deveria, já que fizemos um logarítmo de uma função exponencial que decresce) e que, por uma margem de erro pequena, se diferencia da constante α do material Rb82. Uma das razões do erro ter acontecido pode ser visto se pensarmos que a taxa com que a função de decaimento variou em um tempo definido (ou melhor, um $\Delta t = 1 segundo$) pode ser explicada como uma variação com intervalos de tempo muito grandes (já que sabemos, pelo conceito de derivada, que quanto mais o intervalo Δt se aproxima de 0, maior é a precisão do cálculo). Cada vez que diminuimos o valor de Δt, nesse caso, chegamos perto de um valor conhecido: α (é importante lembrar que estamos trabalhando com o módulo dos valores). Então podemos afirmar que:

$$|a| \approx |\alpha| \tag{2}$$

Agora, se pensarmos que o logarítmo natural de N(t) nos deu uma reta, podemos escrever que:

$$\ln f(x) = ax + b \Rightarrow \alpha t + b \tag{3}$$

Onde α = coeficiente angular, t = tempo e b = coeficiente linear. Sabemos que $\alpha = -0.00928295$ e se olharmos no gráfico veremos que a reta corta o eixo y em $(0, 6.90775)$, sendo então o coeficiente linear o valor 6.90775. Temos então:

$$\ln N(t) = \alpha t + b \Rightarrow -0.00928295t + 6.90775 \tag{4}$$

$$\log_e N(t) = -0.00928295t + 6.90775 \tag{5}$$

Aplicando as propriedades logarítmicas conseguimos:

$$e^{-0.00928295t+6.90775} = N(t) \tag{6}$$

ou

$$N(t) = e^{(-0.00928295t)}e^{(6.90775)} \tag{7}$$

O segundo termo dessa expressão nos dá aproximadamente o valor 1000, que é também o nosso valor de $N(0)$, ou simplesmente a condição inicial do nosso problema. Portanto:

$$N(t) = N(0)e^{(-0.00928295t)} \tag{8}$$

concluindo:

$$\boxed{N(t) = N(0)e^{-\alpha t}}$$

5. Calcule numericamente o tempo de vida média τ do Rb82, computando explicitamente a soma ponderada, e compare o resultado com o inverso da constante α.

Observação: os cálculos foram feitos no mesmo programa de decaimento de núcleos radioativos (1.). Sabemos que, para calcular o tempo de vida média de um material (no nosso caso, o Rb82), precisamos usar a expressão

$$\tau = \sum_{i=1}^{n} \frac{\Delta N_i t_i}{N(0)} \tag{9}$$

e desenvolvendo essa expressão conseguimos:

$$\tau = \frac{\Delta N_1 t_1 + \Delta N_2 t_2 + \Delta N_3 t_3 + ...\Delta N_n t_n}{N(0)} \tag{10}$$

Uma observação importante é que a precisão do cálculo vai ser definida pela quantidade de termos $\Delta N_i t_i$; ou seja, quanto mais termos, mais preciso é o cálculo de vida média do material em questão. Como usamos 2000 termos, fica difícil montar uma tabela e mostrar os resultados (mas basta conferir no programa), e $N(0)$ tem um valor fixo 1000.

Colocando os valores na expressão, o resultado obtido é: $108.224716s$. Portanto:

$$\tau = 108.224716s \tag{11}$$

E se fizermos o valor inverso da constante α do material, teremos

$$\frac{1}{\alpha} = \frac{1}{0.00924s} = 108.225108s^{-1} \tag{12}$$

e

$$\frac{1}{\tau} = \frac{1}{108.224716s} = 0.00924003s^{-1} \tag{13}$$

E nesse caso podemos concluir:

$$\boxed{\tau \approx \tfrac{1}{\alpha} \wedge \tfrac{1}{\tau} \approx \alpha}$$

4.2 Problemas Cap. 2 (Crescimento Exponencial

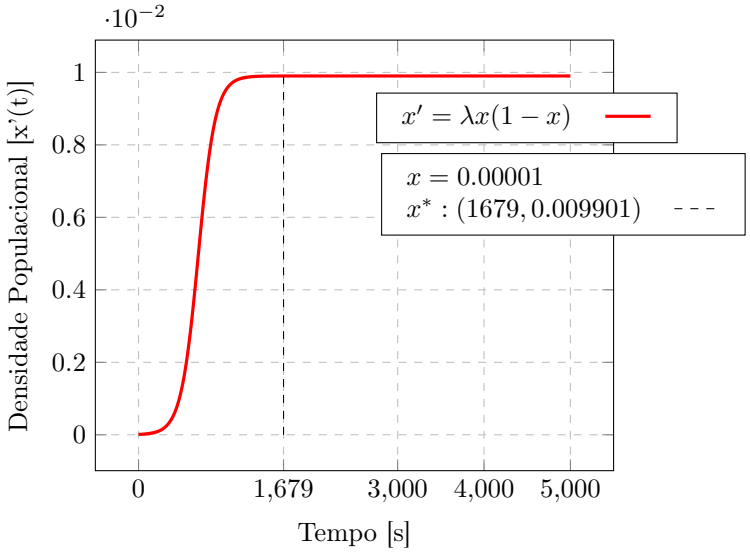
1. Use o mapa iterativo (2.3) com $\Delta t = 1s$ e $\alpha = 0.01s^{-1}$ e mostre que o número de bactérias dobra a cada $70s$.

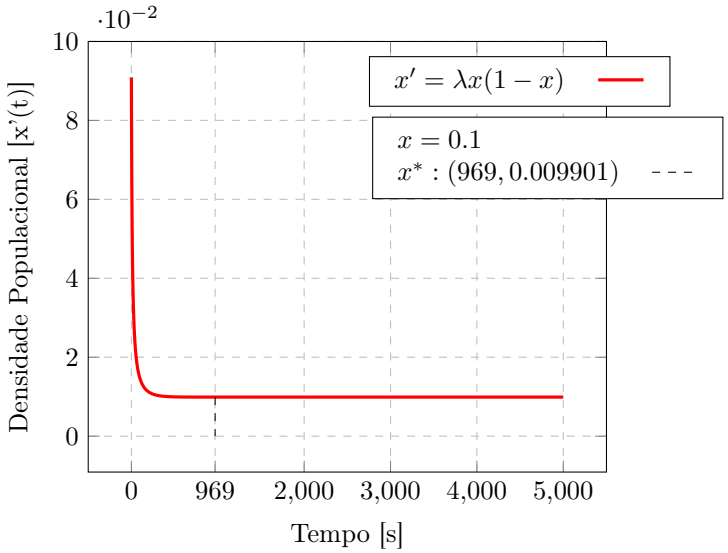
```
#include <stdio.h>

int main()
{
    float n=1;
    int t;
    int tempo[71];
    float x[71];

    FILE *fp;

    fp = fopen("Dados teóricos", "w");

    fprintf(fp, "Resultados para plotagem gráfica:\n\nTempo [s] | Densidade final [x'(t)]\n");
    for (t=1; t<=71; t++){
         if (t==1){
           x[1]=10;
           tempo[1]=0;
           t=2;
         }
        n=n*(1+0.01);
    tempo[t]=t-1;
    x[t]=n;
    }
    for(t=1; t<=71; t++){
        fprintf (fp, "\n    %d              %f", tempo[t], x[t]);
    }

    fclose(fp);

    return 0;

}
```

3. Se considerarmos novamente que as bactérias se duplicam a cada segundo na proporção de uma a cada cem - enquanto houver poucas na lâmina de microscópio, e razoavelmente separadas umas das outros - teremos $\lambda = 1.01s$. Mostre que em vez de explodir, a população se estabiliza numa densidade final $x^* = 0.0099$.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()

    {


    float B;
    float deltat;
    float lnt, lnx;
    float x;
    int t0, t;

    #define alfa 0.01 /* constante de duplicação de bactérias */
    #define lambda 1.01 /* pode ser alterado de acordo com o exercício */

/*  x(t) = B/Bm = limite populacional/densidade */
    deltat = 1;



    printf("Entre com uma quantidade inicial de bactérias: ");
        scanf("%f", &B);

    printf("Entre com uma densidade populacional inicial: ");
        scanf("%f", &x);

    printf("Entre com um tempo inicial: ");
        scanf("%i", &t0);

    printf("Entre com um tempo final: ");
        scanf("%i", &t);


    FILE *fp;

    fp = fopen("Dados teóricos", "w");


    fprintf(fp, "Resultados para plotagem gráfica:\n\nTempo [s] | B(t) | x'(t) | ln(x) | ln(t)\n")
     ;
    for (t0 = 0; t0 <= t; t0++)

    {

    lnt = log(t0);
    x = lambda * x * (1 - x);
    lnx = log(x);
    fprintf(fp, "\n%i  %f  %.10f  %f  %f\n", t0, B, x, lnx, lnt);
    B = B * (1 + alfa * deltat);

    }


    fclose(fp);
```

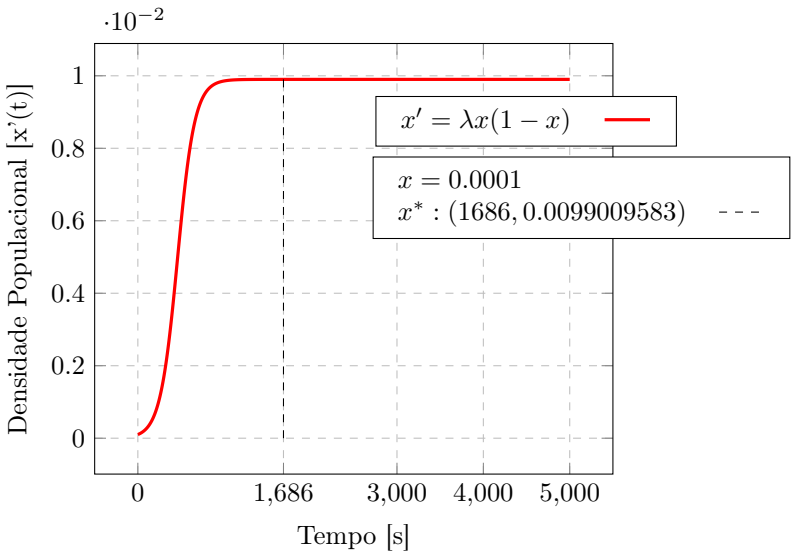
```
return 0;

}
```


$\cdot 10^{-2}$
1
0.8
0.6
0.4
0.2
0
Densidade Populacional [x'(t)]
$x' = \lambda x(1-x)$
$x = 0.00001$
x^* : $(1679, 0.009901)$
0
1,679
3,000
4,000
5,000
Tempo [s]


$\cdot 10^{-2}$
10
8
6
4
2
0
Densidade Populacional [x'(t)]
$x' = \lambda x(1-x)$
$x = 0.1$
x^* : $(969, 0.009901)$
0
969
2,000
3,000
4,000
5,000
Tempo [s]

4. Começando de uma população inicial correspondente a $x(0) = 0.0001$, na mesma situação do problema anterior, mostre que o sistema se estabiliza em $x^* = 0.00990$ depois de $20min$ ou $1200s$.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main()
{ float x, lam, var;
  int ti, tf, deltat;
#define lam 1.01 //Constante de multiplicação

    x = 0.0001; //Quantidade inicial de bactérias
    ti = 0;
    tf = 5000; //Tempo em segundos.
    deltat = 1;

    FILE *fp;
    fp = fopen("Dados teóricos", "w");

    while (ti<=tf)
    {
        fprintf(fp, "Tempo: %i  Densidade de bactérias: %.10f \n", ti, x);
        x*=lam*(1-x);//X'=X*lambda*(1-x)
        ti+=deltat;

    }

    fclose(fp);

    return 0;
}
```

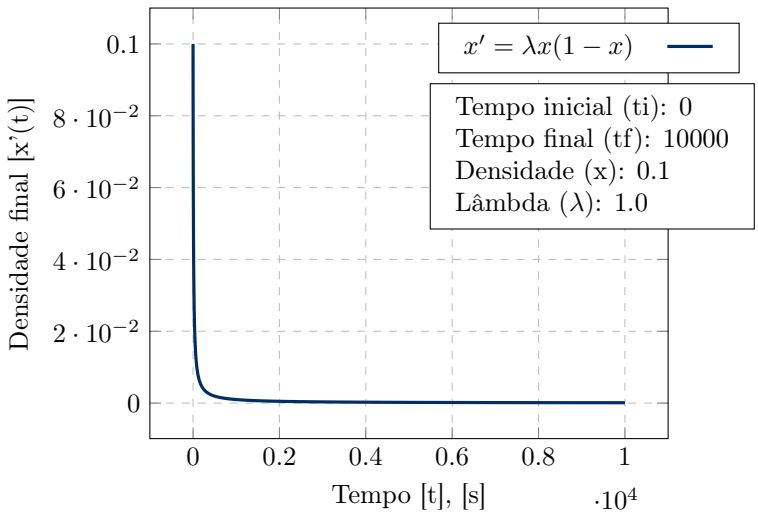


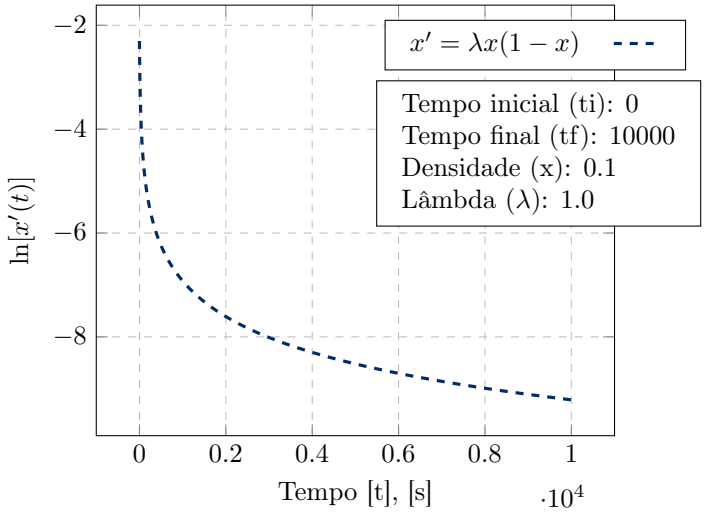
8. Itere umas dez mil vezes o mapa logístico (2.4) na situação crítica $\lambda = \lambda = \lambda_0 =$

1. Comece em $t = 0$ com um valor de densidade inicial arbitrário, por exemplo, $x = 0.1$. Armazene os sucessivos valores de t e x num arquivo de dados com duas colunas, e construa posteriormente o gráfico $x \times t$. Você obterá uma curva que descreve o decaimento. No problema 1-3, verificamos que o gráfico curvo de um decaimento exponencial se transforma num gráfico reto se adotarmos escala logarítmica no eixo vertical. Com os mesmos dados do parágrafo anterior, construa um segundo gráfico $x \times t$, dessa vez com escala logarítmica no eixo vertical. Em vez de uma reta, você obterá uma outra curva, o que mostra não ser exponencial a função x(t). Por esta razão o decaimento crítico é denominado anômalo. Ainda com os mesmos dados, construa um terceiro gráfico, dessa vez com escalas logarítmicas no eixo horizontal e no vertical. Agora, seu gráfico é reto para tempos suficientemente grandes. Determine o coeficiente angular da reta.

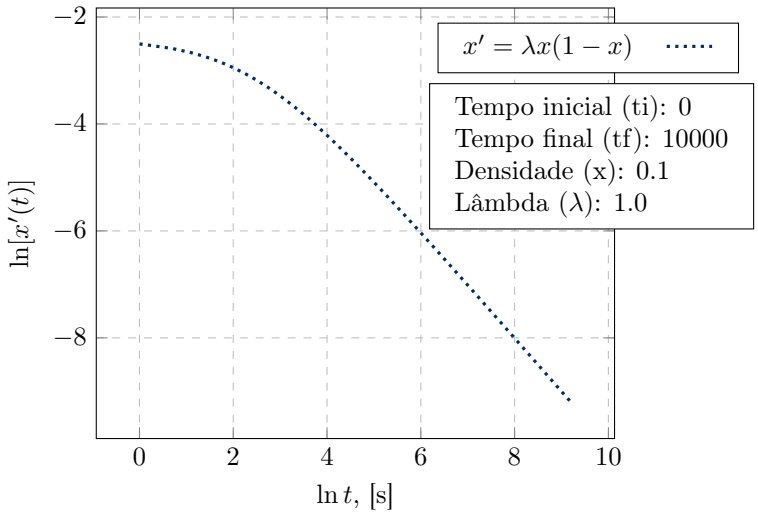
```
#include <stdio.h>
#include <math.h>

int main()
{
  //Entrada
   float ti, tf;
   float x, lamb;
   FILE *fp;
   fp = fopen("MapLog.dat","w");

   printf("Digite o tempo inicial:");
   scanf("%f", &ti);

   printf("Digite o tempo final:");
   scanf("%f", &tf);

   printf("Digite o valor da densidade inicial:");
   scanf("%f", &x);

   printf("Digite o valor de lambda:");
   scanf("%f", &lamb);

  //Processamento
  for(ti=0; ti<=tf ; ti++)
  {
    log (x);
    log (ti);
    fprintf(fp,"%.f  %.10f  %.10f  %.10f\n", ti, x, log(x), log(ti));
     x = x * lamb * (1 - x);
  }
  return 0;
}
```

Após a construção do programa, será construido três gráficos, todos contendo as mesmas condições iniciais.





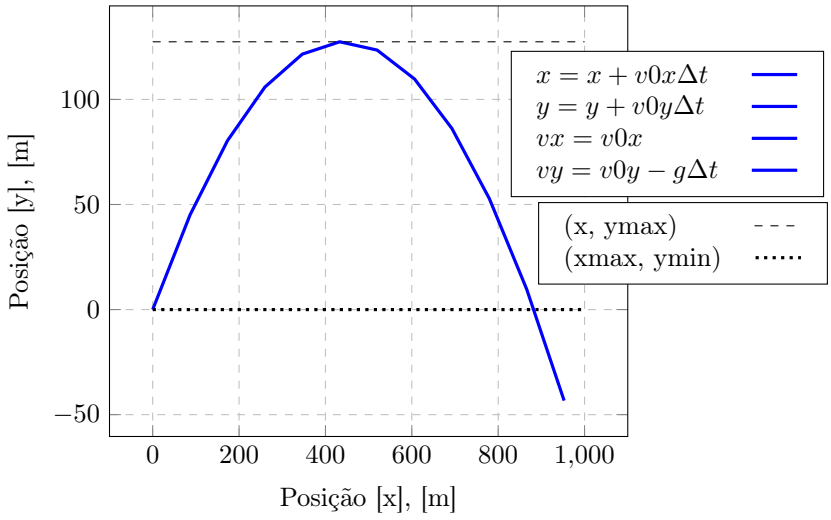
Observação importante: O exercício em questão afirma que ao colocar os dois componentes do primeiro gráfico (densidade e tempo) em escala logarítmica, ele ficaria reto. Porém, como observado abaixo, o terceiro gráfico não está relilíneo. Nós, do grupo, não conseguimos achar onde estava o erro pois ao nosso ver o programa foi corretamente programado.


−2
−4
−6
−8
0
2
4
6
8
10
ln[x′(t)]
ln t, [s]
x′ = λx(1 − x)
Tempo inicial (ti): 0
Tempo final (tf): 10000
Densidade (x): 0.1
Lâmbda (λ): 1.0

4.3 Problemas Cap. 3 (Lançamento Oblíquo)

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()

    {


    float x0, x, y0, y;
    float v0x, vx, v0y, vy;
    float ang, gr, rad;
    int dt, t0, t;

    #define g 9.80665


        printf("Entre com uma posição inicial em x: ");
            scanf("%f", &x0);

        printf("Entre com a mesma posição inicial em y: ");
            scanf("%f", &y0);

        printf("Entre com uma velocidade inicial em x: ");
            scanf("%f", &v0x);

        printf("Entre com a mesma velocidade inicial em y: ");
            scanf("%f", &v0y);

        printf("Entre com um tempo inicial: ");
            scanf("%i", &t0);

        printf("Entre com um tempo final: ");
            scanf("%i", &t);

        printf("Insira um ângulo de lançamento em graus: ");
            scanf("%f", &gr);



    rad = 0.017453 * gr;

        printf("\n0 ângulo de lançamento de %.f graus em radianos é: %f rad\n", gr, rad);

    ang = rad;


    FILE *lo;


    lo = fopen("Dados teóricos", "w");

    fprintf(lo, "Resultados para plotagem gráfica: \n\n");
    fprintf(lo, "\n\nTempo [s] | Velocidade, x [m/s] | Velocidade, y [m/s] | Posição, x [m] | Posi
     ção, y [m]\n\n");



    /* decomposição vetorial */
```

```
    v0x = v0x * cos(ang);
    v0y = v0y * sin(ang);
    /* --------------------- */



    for (t0 = 0; t0 <= t; t0++)

    {

    x = x0 + v0x * t0;
    y = y0 + v0y * t0 - g * (t0 * t0)/2;

    vx = v0x;
    vy = v0y - g * t0;

    /* outra possibilidade de resolução
    diferencial do problema (e preferencialmente
    adotando deltat, dt = 1s */

//    x = x + v0x * dt;
//    y = y + v0y * dt;
//    v = v0x;
//    v = v0y - g * dt;

    /* ------------------------------- */

    fprintf(lo, "\n%i  %.3f  %.3f  %.3f  %.3f", t0, vx, vy, x, y);

    if (y < 0)
    {
       break;
    }

    }

    fclose(lo);

    return 0;

    }
```


100
50
0
−50
0
200
400
600
800
1,000
Posição [x], [m]
Posição [y], [m]
$x = x + v0x\Delta t$
$y = y + v0y\Delta t$
$vx = v0x$
$vy = v0y - g\Delta t$
(x, ymax)
(xmax, ymin)

4.4 Problemas Cap. 4 (Movimento Harmônico Simples - Pêndulo)

1. Verifique que o erro cometido na aproximação (4.7) é de 1% para $\theta = 0.25rad$ (ou 14°), e que o erro é menor ainda para ângulos menores.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
int main()
{
#define pi 3.14159265
    float h; // h = Operação para determinar o valor do ângulo. Que é o nosso teta.
    int x; // ângulos em graus.
    float ep, vap, ve;


        while(x<=30)
        {
          h=(x*pi)/180;  //Transformação para radianos.
          sin(h);
          vap = h;
          ve = sin(h);
          ep = fabs(vap - ve)/ve * 100;
            printf("seno (%d graus) = %f     Valor de teta: %f      Erro percentual: %f%%\n", x,
     sin(h), h, ep);
        x++;
        }
    return 0;
}
```

	$\sin\theta$	θ	$\%erro = \frac{\lvert\theta - \sin\theta\rvert}{\sin\theta} \times 100$
1°	$\sin 0.017453$	0.017453	0.005080%
5°	$\sin 0.087266$	0.087266	0.127040%
10°	$\sin 0.174533$	0.174533	0.509502%
14°	$\sin 0.244346$	0.244346	1.002057%
18°	$\sin 0.314159$	0.314159	1.664074%
22°	$\sin 0.383972$	0.383972	2.500185%
26°	$\sin 0.453786$	0.453786	3.516300%
30°	$\sin 0.523599$	0.523599	4.7197585%

Observação: Se adotarmos $\theta = 0.25rad$ e aplicarmos na expressão de erro percentual, teremos:

$$\%erro = \frac{|\theta - \sin\theta|}{\sin\theta} \times 100 \tag{14}$$

$$\frac{|0.25 - 0.247403959254523|}{0.247403959254523} \times 100 \approx 1.04\% \tag{15}$$

Que ainda é um dado factível.
observação: uma outra forma de resolução se encontra na seguinte imagem (de uma tabela): `https://prnt.sc/vrllae`

PROGRAMA GERAL:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
    {
        /* I) forma não diferencial de resolução */    float teta, tetaA, costetaA, arccostetaA,
     Te, Pe;
                                                       float teta2, tetaA2;
        /* II) forma diferencial de resolução */       float tetaD, teta0, teta1;
                                                       float tetaD2, teta02, teta12;

        /* I) */                                       float dt, t0, t;
                                                       float dt2, t02, t2;
        /* II) */                                      float DT, T0, T, Ti, Tf;
                                                       float DT2, T02, T2, Ti2, Tf2;

        #define g 9.8
        #define pi 3.14
        float l=2.0;
        float v0=1.0;
        float vmin= 0.25 * sqrt(g * l);
        float vc= 2 * sqrt(g * l);
/* -------------------------------- */
        FILE *out;
    out = fopen("I) Dados teóricos", "w");

    fprintf(out, "I) Resultados para plotagem gráfica dentro das aproximações: \n\n");
    fprintf(out, "\n\nTempo, t [s] | Posição, teta(t) [rad]\n\n");
```

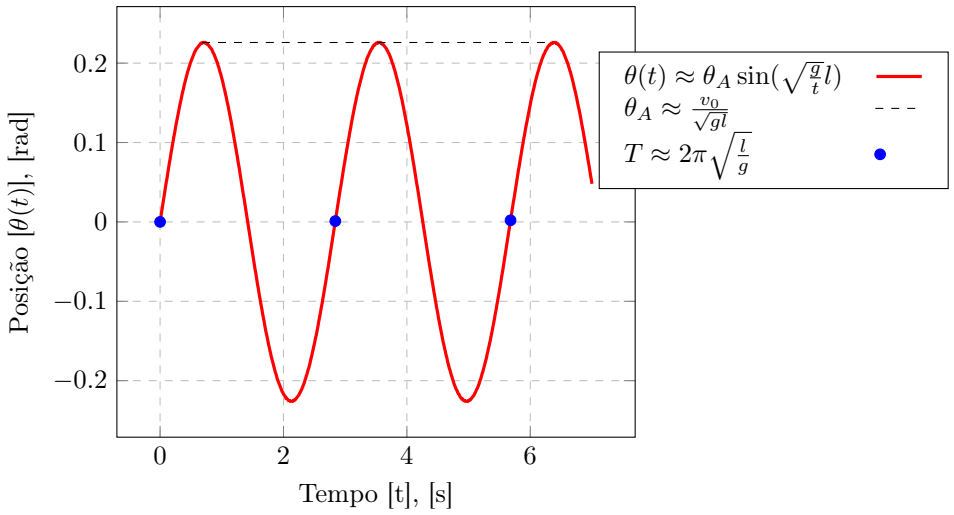
```
    dt = 0.01;
    t0 = 0.0;
    t = 7.0 + dt;

    for (t0 = 0; t0 <= t; t0+=dt)
    {
    teta = tetaA * sin(sqrt(g/l) * t0);
    tetaA = v0/sqrt(g * l);
    fprintf(out, "\n%.2f    %.3f", t0, teta);
    }
    tetaA = v0/sqrt(g * l);
    fprintf(out, "\n\nAmplitude: %.3f\n", tetaA);
    Te = 2 * pi * sqrt(l/g);
    fprintf(out, "\nPeríodo: %.2f s\n\n", Te);
    fclose(out);
  //-------------------------------------------------//
     FILE *out2;
    out2 = fopen("II) Dados teóricos", "w");

    fprintf(out2, "II) Resultados para plotagem gráfica fora das aproximações: \n\n");
    fprintf(out2, "\n\nTempo, t [s] | Posição, teta(t) [rad]\n\n");

    DT = 0.01;
/* ---------------------------------------------------------*/
   /* condição inicial teta(0) */       teta0 = 0.0;
   /* tempo */                          T0 = 0.0;
    fprintf(out2, "\n%.2f    %.3f", T0, teta0);

   /* teta(1) */                        teta1 = v0/l * DT;
   /* tempo */                          T0 = DT;

    fprintf(out2, "\n%.2f    %.3f", T0, teta1);
/* ---------------------------------------------------------*/
    DT = 0.01;
    T0 = 0.0;
    T = 7.0;
    Tf = T/DT;
    teta0 = 0.0;
    teta1 = v0/l * DT;
    for (Ti = 2; Ti <= Tf; Ti++)
    {

    T0 = Ti * DT;
    tetaD = 2 * teta1 - (g/l) * (DT*DT) * sin(teta1) - teta0;
    if(tetaD>pi)
    {
    tetaD -= 2*pi;
    teta1 -= 2*pi;
    teta0 -= 2*pi;
    }
    fprintf(out2, "\n%.2f    %.3f", T0, tetaD);
    //------------//
    teta0 = teta1;
    teta1 = tetaD;
    //------------//
    }
    costetaA = 1 - ((v0 * v0)/(2 * g * l));
    arccostetaA = acos(costetaA);
    fprintf(out, "\n\nAmplitude: %.3f\n", arccostetaA);
//-------------------------------------------------//
    DT2 = 0.01;
    T02 = 0.0;
```

```
teta02 = 0.0;
teta12 = v0/l * DT2;
Pe = -1.0;
Ti = 2.0;

while(Pe<0.0)
{
tetaD2 = 2 * teta12 - (g/l) * (DT2*DT2) * sin(teta12) - teta02;
if(tetaD2>pi)
{
tetaD2 -= 2*pi;
teta12 -= 2*pi;
teta02 -= 2*pi;
}

if(tetaD2*teta12<0.0)
{
Pe = (Ti + tetaD2/(teta12 - tetaD2)) * DT2;

if(tetaD2<0.0)
Pe *= 2.0;
}
//------------//
teta02 = teta12;
teta12 = tetaD2;
//------------//
Ti++;
}
fprintf(out2, "\nPeríodo: %.2f s\n\n", Pe);

fclose(out2);

return 0;
}
```

2. Considere um pêndulo de comprimento $l = 2.00m$ cuja velocidade inicial é $v_0 = 1.00m/s$ na posição vertical. O valor local da intensidade do campo gravitacional é $g = 9.8m/s^2$. Dados estes valores com os algarismos significativos apresentados, justifica-se ou não a aproximação de pequenas oscilações (4.7), (4.8), (4.9) e (4.10) neste caso?

Ao rodar o programa geral apresentado acima (em específico no seu primeiro arquivo imprimido, que trata sobre as equações apresentadas no livro como estando dentro das aproximações), sim, as aproximações podem ser justificadas.

3.Construa o gráfico da posição θ em função do tempo t, no caso do problema anterior.



4.Mostre que o período de oscilação do pêndulo vale $T \approx 2\pi\sqrt{\frac{l}{g}}$ (4.11) dentro da aproximação de pequenas oscilações, e verifique este resultado no caso do gráfico do problema anterior.

$$\tau = r_{\perp}F \tag{16}$$

$$\tau = -l(Fgsen\theta) \tag{17}$$

$$\tau = I\alpha \tag{18}$$

$$-lmgFgsen(\theta) = I\alpha \tag{19}$$

$$\alpha = -\frac{mgl}{I}sen\theta \tag{20}$$

$$\alpha = -\frac{mgl}{I}\theta \tag{21}$$

$$a(\tau) = -w^2x(\tau) \tag{22}$$

$$\alpha = a(I) \tag{23}$$

$$w^2 = \frac{mgl}{I} \tag{24}$$

$$w = \sqrt{\frac{mgl}{I}} \tag{25}$$

$$w = \frac{2\pi}{T} \tag{26}$$

$$I = ml^2 \tag{27}$$

$$T = 2\pi\sqrt{\frac{ml^2}{mgl}} \tag{28}$$

$$\boxed{T = 2\pi\sqrt{\frac{l}{g}}}$$

5. Com o uso da conservação de energia, demonstre a equação (4.12).

$$E = K + U \tag{29}$$

$$E = \Delta K + \Delta U \tag{30}$$

$$K_i + U_f = K_f + Ui \tag{31}$$

$$K_f = -U_i \tag{32}$$

$$\frac{1}{2}mv^2 = -mgy \tag{33}$$

Lembrando que:

$$v^2 = v_o^2 + 2gl \tag{34}$$

$$y = lcos\theta \tag{35}$$

Continuando:

$$\frac{1}{2}mv^2 = -mglcos\theta \tag{36}$$

$$\frac{v^2}{2} = -glcos\theta \tag{37}$$

$$cos\theta = -\frac{v^2}{2gl} \tag{38}$$

$$cos\theta = -\frac{v_o^2 - 2gl}{2gl} \tag{39}$$

$$cos\theta = \frac{2gl - v_o^2}{2gl} \tag{40}$$

$$cos\theta = \frac{2gl}{2gl} - \frac{v_o^2}{2gl} \tag{41}$$

$$\boxed{cos\theta = 1 - \frac{v_o^2}{2gl}}$$

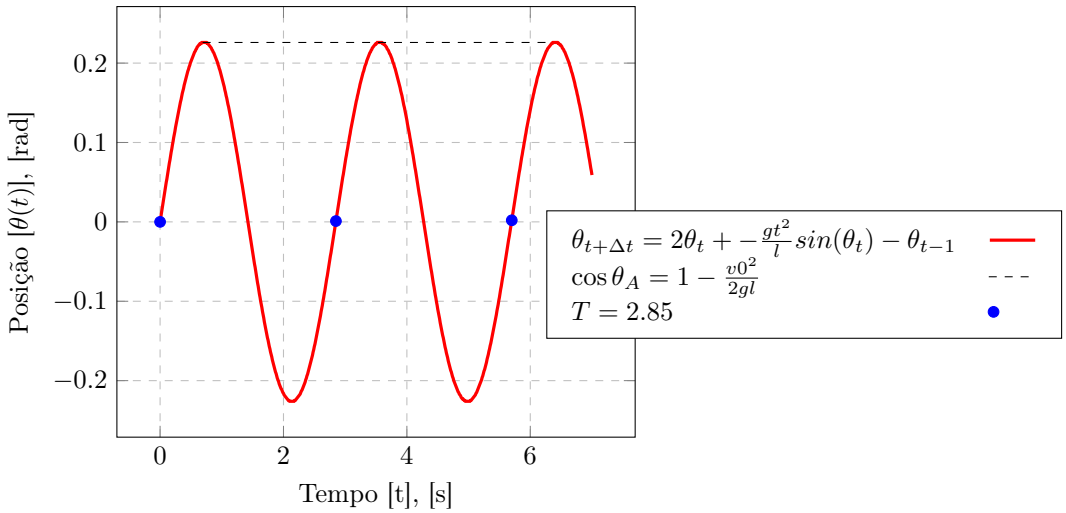
6. Resolva numericamente a equação (4.5) do pêndulo, através do mapa (4.18), com os mesmos dados do problema 4-2. Escolha um intevalo Δt adequado à precisão compatível com a dos dados apresentados. Além de $\theta_0 = 0$, você necessitará conhecer a priori o valor de θ_1, a partir da condição inicial (4.6). Para tanto, basta notar que a derivada de $\theta(t)$ no instante $t = 0$ pode ser escrita como $\frac{\mathrm{d}\theta}{\mathrm{d}t}, t = 0 \approx \frac{\theta_1}{\Delta t}$

Programa mostrado anteriormente (visualizar o segundo arquivo imprimido que mostra a resolução da equação fora das aproximações).

7. A partir da tabela θ_0, θ_1, θ_2, etc obtida no problema anterior, determine o período de oscilação T, e compare o resultado com o valor previsto pela aproximação (4.11). Ao medir o período, procedimentos simples de interpolação são recomendáveis para aumentar a precisão.

De acordo com o programa já mostrado, temos que: Para o primeiro caso (e o primeiro arquivo imprimido pelo programa) que trata das expressões dentro das aproximações, usando as condições dadas pelo exercício 2 e com $dt = 0.01$, o período é:
$T = 2.84s$
Para o caso das expressões diferenciais fora das aproximações e com as mesmas condições citadas, o período é de:
$T = 2.85s$
Ou seja, um erro percentual equivalente a $\approx 0.35\%$

8. A partir da tabela θ_0, θ_1, θ_2, etc obtida no problema 4-6, construa o gráfico da posição θ em função do tempo t, e compare-o com o obtido no problema 4-3.



Através dos resultados obtidos no programa, é factível dizer que houveram erros bem pequenos.

9. Repita o problema 4-6, desta vez usando uma velocidade inicial 5 vezes menor, $v_0 = 0.200m/s$. Compare o período T e a amplitude θ_A obtidos com os valores previstos pelas aproximações (4.11) e (4.10).

```
#include <stdio.h>
#include <math.h>

#define g 9.8
#define v0 0.200
#define l 2.0
#define deltat 0.0001
#define PI 3.1415926535

int main()
{
    float teta, tetaA, tetaAA, t, T, cons, amp, amp2;
    FILE *rt;
    rt = fopen("DADOS","w");

    t = tetaAA = 0.0;
    fprintf(rt,"%.4f      %.6f\n", t, tetaAA);

    t = deltat;
    tetaA = (v0*deltat) / l;
    fprintf(rt,"%.4f      %.6f\n", t, tetaA);

    cons = (g*deltat*deltat)/l;
    amp = v0 / sqrt(g*l);
    amp2 = cos(amp);
    T = 2*PI*sqrt(l/g);

    for(t=0.01; t<=6.0; t)
    {
        teta = (2*tetaA)-(cons*sin(tetaA))-tetaAA;
        t = t + deltat;
        fprintf(rt,"%.4f      %.6f\n", t, teta);
        tetaAA = tetaA;
        tetaA = teta;
    }

    fprintf(rt,"\n\n");
    fprintf(rt,"O valor do período é (para aproximação): %f\n", T);
    fprintf(rt,"O valor da amplitude é (para aproximação): %f\n", amp);
    fprintf(rt,"O valor da amplitude é: %f\n", amp2);

    return 0;
}
```

O programa acima determina os valores da amplitude usando uma equação dentro das aproximações e usando, também, uma equação fora das aproximações, sendo essa segunda mais precisa. Para o período, o programa acima apenas trás o resultado usando a equação dentro das aproximações. Para obter o resultado o período usando a equação fora das aproximações, usaremos um programa encontrado no livro "Física em Computadores"dos autores Paulo Murilo Castro de Oliveira e Suzana Maria Moss de Oliveira, com pequenas modificações:

```
#include <stdio.h>
#include <math.h>

#define g 9.8
```

```
#define l 2.0
#define v0 0.002
#define deltat 0.0001

int t;
double teta,tetaA,tetaAA,cons;
float T=-1.0;

int main()
  {
   cons = g*deltat*deltat/l;
   tetaAA = 0.0;
   tetaA = v0*deltat/l;
   t = 2;

    while(T<0.0)
    {
    teta = 2*tetaA-cons*sin(tetaA)-tetaAA;

      if(teta*tetaA<0.0)
      {
       T = (t+teta/(tetaA-teta))*deltat;

         if(teta<0.0)
         T *= 2.0;

          }
           tetaAA = tetaA;
           tetaA = teta;
           t++;
          }

printf("\n\n O valor do período é = %f\n",T);

    return 0;
    }
```

Podemos analisar o período através do gráfico também:
Percebe-se que olhando pelo gráfico, o período consiste em um valo entre 2.5 e 3, mas não temos uma precisão. Diferente de quando usamos o programa.

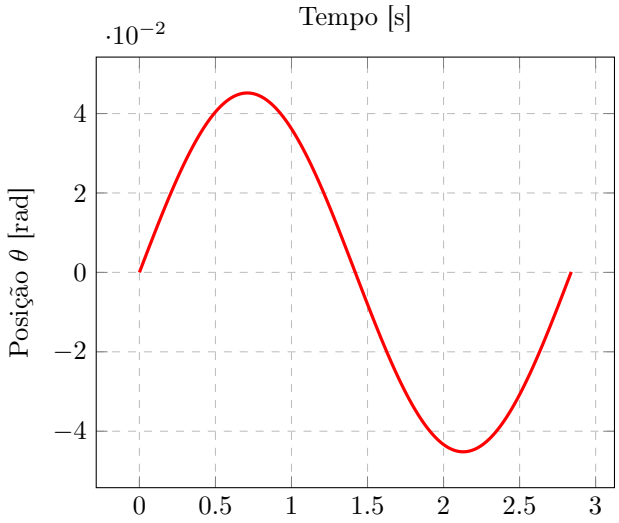


Gráfico da função $\theta_{t+\Delta t} = 2\theta_t + -\frac{gt^2}{l} sin(\theta_t) - \theta_{t-1}$

10. Repita novamente o problema 4-6, desta vez usando uma velocidade inicial 5 vezes maior, $v_0 = 5.00m/s$. Verifique que as aproximações (4.11) e (4.10) não são mais válidas, mas o resultado (4.12) sim.

Para isso, é questão apenas de análise de dados do problema geral apresentado no início. Em seguida, é só questão de alterar a velocidade inicial e observar o período e amplitude do pêndulo ppara tentar argumentar fisicamente e matematicamente sobre o problema.

Ao colocar velocidades pequenas como $v0 = 0.2m/s$, se analisar com cuidado resultados, dá pra notar que a amplitude e o período são ou aproximadamente ou exatamente como devem ser (a amplitude = o ponto de oscilação mais alto, o período = o tempo que o pêndulo levou pra completar um ciclo). Já quando você coloca velocidades maiores como $v0 = 5m/s$, os resultados que são imprimidos de amplitude e período são significativamente errados (é só observar com atenção) e isso acontece porque quando aumentamos a velocidade, consequentemente fizemos com que o pêndulo atingisse um ponto maior (resumindo, aconteceram grandes oscilações, e não pequenas), e é por isso que o livro apresenta a equação 4.12, que permite aconteça uma generalização para velocidades altas e consequentemente amplitudes altas. Quando aplicamos os valores conhecidos do exercício 4-2 na expressão generalizada 4.12, conseguimos: $\cos\theta_A = 0.97448$.

Esse resultado é essencialmente e trivialmente provado através da aplicação do arccos, que é: 0.22640 que condiz com os pontos (0.68, 0.226) que é, nesse caso, a amplitude vista no nos dados do programa e no gráfico.

11. Mostre que $v_c = 2\sqrt{gl}$

$$v^2 = v_c^2 - 2gby \tag{42}$$

$$0 = v_c^2 - 2g(2l) \tag{43}$$

$$v_c^2 = 4gl \tag{44}$$

$$v_c = \sqrt{4gl} \tag{45}$$

$$\boxed{v_c = 2\sqrt{gl}}$$

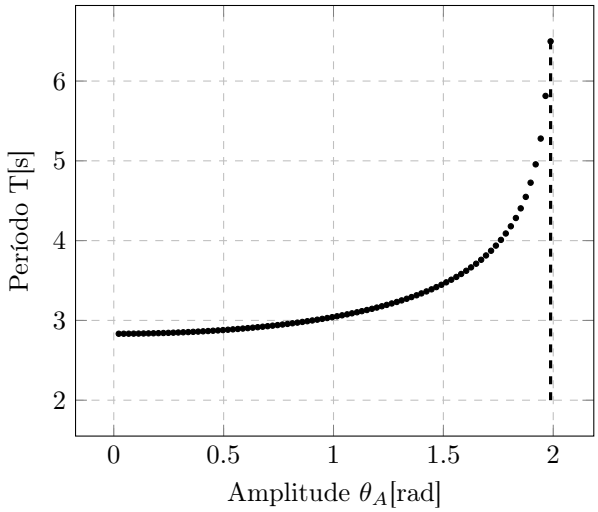
12. Repita o problema 4-6 para diferentes valores de v_0 todos menores do que o limite v_c, e meça o período T e a amplitude θ_A em cada caso. Construa um gráfico de T em função de θ_A

```
#include <stdio.h>
#include <math.h>

#define g 9.8 /*aceleração da gravidade */
#define l 2.0  /* comprimento da corda */
#define deltat 0.1 /* intervalo de tempo) */
#define PI 3.1415926535

int main()
{
    float vmax, T, amp, v0;
    FILE *rt;
    rt = fopen("DADOS","w");

    v0 = 0.01; /* velocidade inicial */
    vmax = 2 * (sqrt(g * l));

    fprintf(rt,"VEL. INICIAL | AMPLITUDE\n\n");

    for(v0 = 0.01 ; v0<=vmax; v0)
    {
        v0 = v0 +deltat;
        T = (2 * PI) * sqrt(l/g);
        amp = v0 / (sqrt(g * l));
        fprintf(rt,"%.2f             %.4f\n", v0, amp);
    }
    return 0;
}
```

O programa acima foi utilizado para obter uma tabela de amplitude, utilizando diferentes velocidades iniciais. O programa para de rodar quando atinge o valor máximo compatível com o movimento oscilátorio. A equação que usamos para saber essa velocidade máxima está no programa acima como "vmax".

Para o período, foi utilizado o programa usado no exercicio 4.9 (o segundo programa mostrado no exercício), usando os mesmos valores de velocidade usado no programa acima, porém, ao invés de obter uma tabela com todos os valores sequenciais,os valores usados foram pegos um por um do programa.

O gráfico a seguir mostra o período(T) pela amplitude(θ_A):


Período T[s]
6
5
4
3
2
0
0.5
1
1.5
2
Amplitude θ_A[rad]

13. Repita o problema 4-6, desta vez para uma velocidade inicial $v_0 = 10m/s$, maior portanto do que v_c. Determine o período T do movimento em voltas completas sucessivas.

Para a resolução desse exercício usaremos o programa geral. Apenas colocamos como velocidade inicial 10.0(m/s), o programa mostrará o período, ele possui um pequeno erro, que está a partir da terceira casa decimal. Utilizando o programa obtemos o seguinte gráfico:

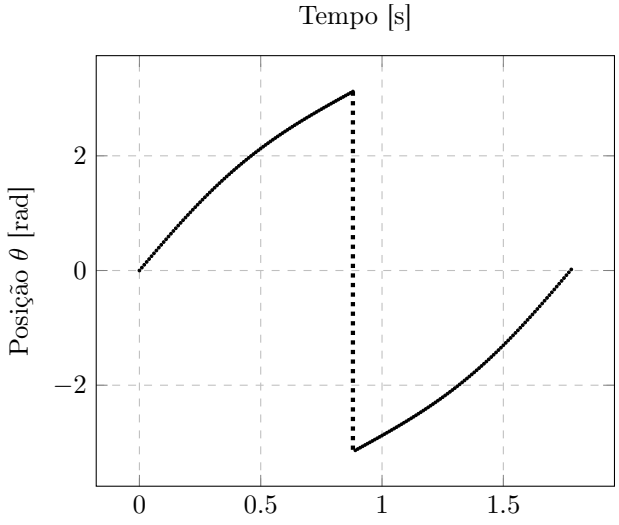
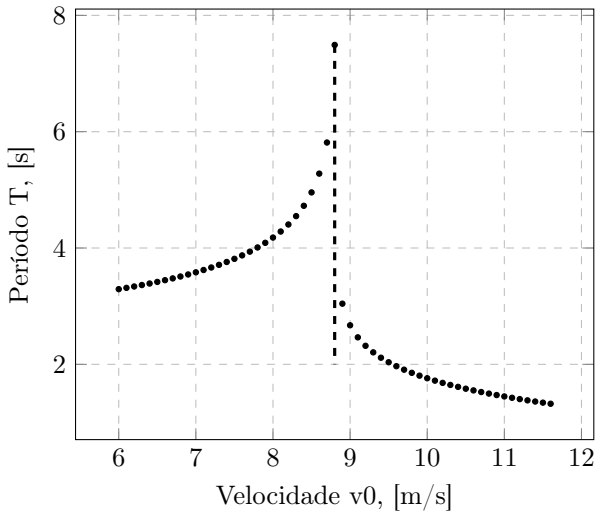


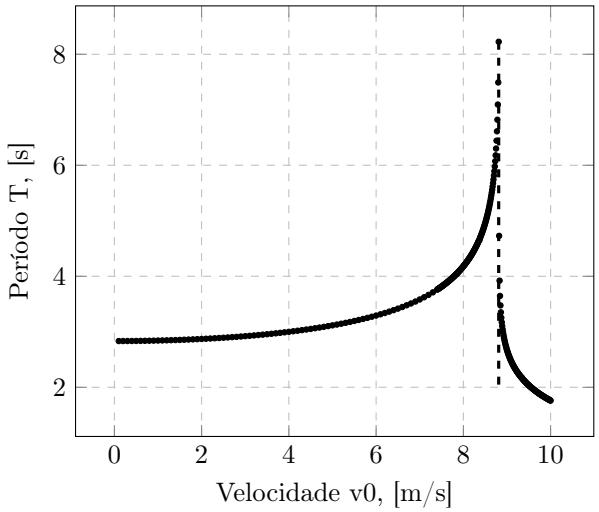
Gráfico da função $\theta_{t+\Delta t} = 2\theta_t + -\frac{gt^2}{l} sin(\theta_t) - \theta_{t-1}$

Como essa velocidade ultrapassou a máxima, o grafico não possui mais a forma de senoide, pois a velocidade do pêndulo está alta. Pelo gráfico nós verificamos que o período está entre 1.5 e 2.0 segundos, quase na metade dos dois, mas não é preciso.

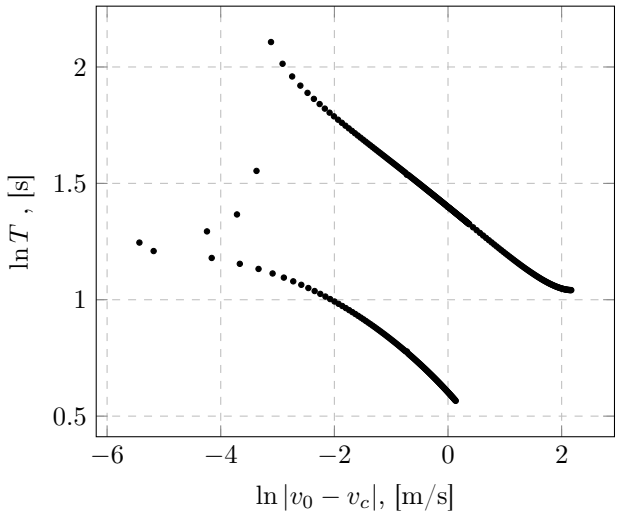
14. Repita o problema 4-6 para diferentes valores de $v_0 = 10m/s$ maiores do que v_c, e meça o período T em cada caso. Com estes dados e mais os obtidos anteriormente no problema 4-12, construa o gráfico do período T em função de v_0.



15. Ainda considerando os mesmos dados do problema 4-2, meça o período T em função de v_0, tomando valores bastante próximos de v_c. Alguns serão menores, outros maiores que v_c. Trata-se da mesma tarefa já solicitada nos problemas 4-12 e 4-14, mas desta vez uma análise mais delicada da precisão numérica se faz necessária. Para cada caso, repita o cálculo para diferentes valores do intervalo de tempo Δt, e, comparando os resultados, retenha apenas os algarismos realmente significativos de T.



16. Com os dados do problema anterior, construa dois gráficos de $\ln T \times \ln|v_0 - v_c|$, um para $v_0 < v_c$ e outro para $v_0 > v_c$.



O que explica o desvio que liga as duas espécies de curva é que os dois gráficos foram plotados juntos com velocidades e intervalos bem próximos, então sugiro considerar apenas as espécies de curvas. A superior é do caso onde $v_0 < v_c$ e a inferior representa $v_0 > v_c$.

17. A unidade natural de T é $\sqrt{\frac{l}{g}}$. Desta forma, $\frac{T}{\sqrt{\frac{l}{g}}}$ é uma grandeza adimensional. Da mesma forma, $|1 - \frac{v_0}{v_c}|$ também é uma grandeza adimensional. Com os mesmos dados do problema 4-15, construa dois gráficos de $\frac{T}{\sqrt{\frac{l}{g}}} \times |1 - \frac{v_0}{v_c}|$, um para $v_0 < v_c$ e outro para $v_0 > v_c$.

Por imprecisões ou intervalos de tempo não tão pequenos ou erros de cálculo, o gráfico não condiz com o que deveria (uma reta).

18. Explique porque o período T do movimento oscilatório, com v_0 ligeiramente menor do que v_c, é o dobro do período T do movimento em voltas completas sucessivas, com v_0 ligeiramente maior (simétrico) do que v_c. Sugestão: observe os gráficos $\theta \times t$ nas duas situações.

Quando v0 é muito pequeno, o pêndulo funciona em sua forma mais simples (oscilando e atingindo amplitudes não muito altas) ou simplesmente representando as equações dentro das aproximações. De acordo com que ele vai se aproximando da velocidade crítica o período aumenta, mas supondo que a gente comece a analisar um pêndulo desde a velocidade pequena, ele vai atingir uma amplitude pequena e consequentemente vai realizar um período bem mais rápido do que se tivesse com uma velocidade alta. E de acordo com que o pêndulo se aproxima de velocidades maiores, a amplitude vai aumentando a ponto de que pode deixar de ser um movimento periódico se ele completar uma volta (na cordinha que ele foi fixado). Isso pode significar que na medida em que o pêndulo atinge uma alta amplitude, ele atinge também uma posição em radiano que deve ser limitada (ou deixa de ser um movimento periódico). Ao colocar colocar uma velocidade muito maior que a crítica, por exemplo, o período vai tender a 0 (provando que o pêndulo deixou de se comportar como antes).

4.5 Problemas Cap. 5 (Difusão)

1. Uma barra metálica com temperatura inicialmente uniforme é submetida a um choque térmico localizado no seu centro $(x = 0)$**. Conside então a função** $u_0(x)$ **nula em todos os pontos da barra, com exceção do ponto central onde** $u_0(0) = 1$**. Considere também** $\Delta x = 1$ **e** $\Delta t =$ **unidades arbitrárias nas quais o coeficiente de difusão vale** $D = 0.1$
Determine a distribuição de temperaturas ao longo da barra, ou seja, a função $u_t(x)$**, para** $t = 1$, 2, 4, 8, 16, 32, 64, 128, 256, 512 **e** 1024**.**

2. Com os dados do problema anterior, calcule $\Delta(t) = \sqrt{\sum_{x=-\infty}^{\infty} x^2 u(x,t)}$ **nos mesmos instantes citados, e mostre que** $\Delta(t) \sim t^{1/2}$

3. Faça iteração numérica da equação (5.19)**, a partir da condição inicial** (5.20)**, para** $p = 1/4$**. Observe o comportamento da função** $u_t(x)$ **ao longo do eixo** x**, a medida em que o tempo passa. Repita para** $p = 3/4$**.**

4. Considere uma barra de alumínio de comprimento $L = 1m$**, sujeita às seguintes condições:**
$T_{x=0,t} = T_{x=L,t} = 0K$, $T_{x,t=0} = 100K$ $(parax \neq 0, L)$
Isso significa que as extremidades da barra são mantidas à temperatura de $0K$**, independente do tempo, e inicialmente** $(t = 0)$ **todos os outros pontos da barra estão a** $T = 100K$**. A partir da equação de calor, chega-se à seguinte relação de recorrência:**

$$T_{i,i+j} = T_{i,j} + \eta[T_{i+1,j} + T_{i-1,j} - 2T_{i,j}], \eta = \frac{\kappa \Delta t}{C\rho \Delta x^2} \tag{46}$$

onde $T_{i,j}$ **denota a temperatura na posição** $x = i\Delta x$ **no instante** $t = j\Delta t$**,** κ **é a condutividade térmica do material,** C **o calor específico e** ρ **a densidade. Para o alumínio,** $\kappa = 237W/(mK)$**,** $C = 900J/(kgK)$ **e** $\rho = 2700kg/m^3$**.**

a. Escreva um programa que implemente a relação (46)**.**

b. Defina um arranjo bidimensional $T[101][2]$ **para a temperatura como função do espaço e do tempo. O primeiro índice é para as** 100 **divisões da barra, enquanto o segundo índice designa o instante atual e o imediatamente anterior.**

c. Para o instante $t = 0$**, inicialize** T **de forma que todos os pontos da barra (exceto as extremidades) estejam a** $100K$**. Fixe as extremidades em** $0K$**.**

d. Comece com 10 passos de tempo e analise se o programa está funcionando adequadamente. Use então milhares de passos, escrevendo o tempo e a temperatura ao longo da barra a cada 500 passos.

e. Analise e comente seus resultados. Utilize gráficos, se possível.

Escreva sobre seu aproveitamento na disciplina (o que aprendeu, as dificuldades, sugestões etc.).

5 Referências Bibliográficas

[1] Paulo M. de Oliveira and Suzana M. de Oliveira. *Física em Computadores*. ., 2010.

[2] Nicholas J Giordano and H Nakanishi. *Computational Physics*. Pearson Education India, 2012.

[3] BJ de SOUSA, DIAS JÚNIOR, and A de A FORMIGA. Introdução a programação. *João Pessoa: Editora da UFPB*, 2014.