Lógica de Programação para Engenharia

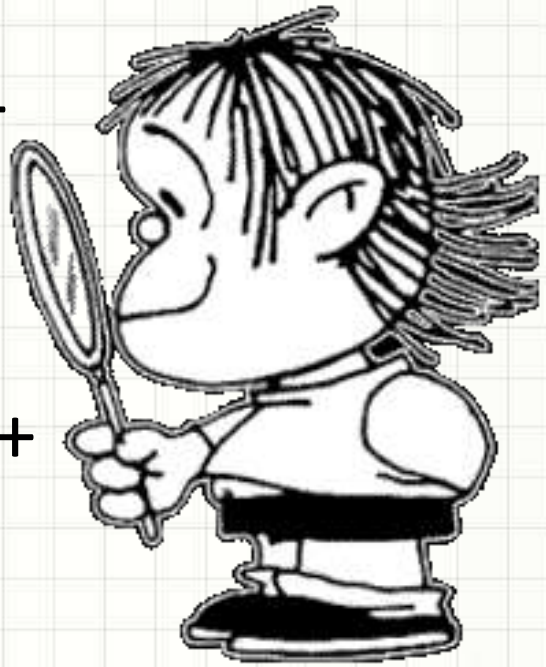
Ambiente de Programação

Prof. Dr. Daniel Caetano

2012 - 1

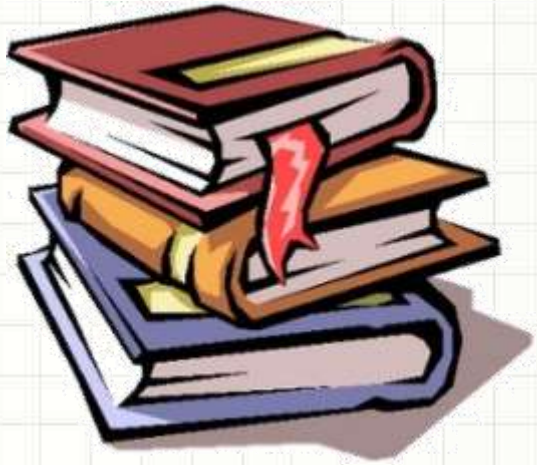
Objetivos

- Entender a utilidade do resto de divisão
- Conhecer as funções matemáticas prontas do C/C++
- Capacitar o aluno para criar algoritmos sequenciais com funções matemáticas em C/C++



- **LISTA 1**

Material de Estudo



Material	Acesso ao Material
Notas de Aula	http://www.caetano.eng.br/aulas/lpe/ (Aula 6)
Apresentação	http://www.caetano.eng.br/aulas/lpe/ (Aula 6)
Material Didático	Lógica de Programação – Fundamentos da Programação de Computadores, páginas 7 a 47.

Resto de Divisão

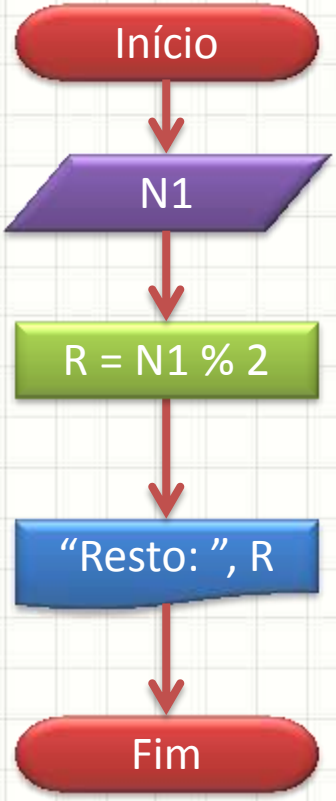
Número Par ou Ímpar?

- Como determinar se um número é par?
- Par: divisível por dois
- O que significa ser divisível por 2?
- Significa que o resto da divisão por 2 é 0!

- Vamos experimentar:
  - Algoritmo que imprime “0” se o número é **par** e “1” se o número é **impar**

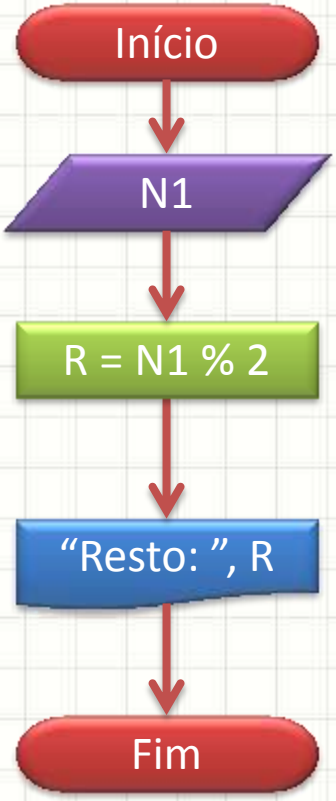
Verificando Paridade

- Linguagem Natural
  1. Leia um número
  2. Calcule o resto da divisão por 2
  3. Imprima o resto
- Fluxograma



Verificando Paridade

- Fluxograma



- Portugol

```
Algoritmo “Calcula Paridade”
Var
N1, R: INTEIRO
Inicio
Escreva(“Digite Um Número:”)
Leia(N1)
R ← N1 % 2
Escreva(“Resto:”, R)
FimAlgoritmo
```

Verificando Paridade

```
Algoritmo "Calcula Paridade"
Var
N1, R: INTEIRO
Inicio
Escreva("Digite Um Número:")
Leia(N1)
R ← N1 % 2
Escreva("Resto:", R)
FimAlgoritmo
```

Verificando Paridade

- Portugol

```
Algoritmo "Calcula Paridade"
Var
N1, R: INTEIRO
Inicio
Escreva("Digite Um Número:")
Leia(N1)
R ← N1 % 2
Escreva("Resto:", R)
FimAlgoritmo
```

- Linguagem C

```
#include <stdio>
#include <iostream>
using namespace std;
int main(void)  {
    int N1, R;
    cout << "Digite Um Número:";
    cin >> N1;
    R = N1 % 2;
    cout << "Resto: " << R;
    getchar();
}
```

Verificando Paridade

```
#include <stdio>
#include <iostream>
using namespace std;
int main(void)  {
    int N1, R;
    cout << “Digite Um Número:”;
    cin >> N1;
    R = N1 % 2;
    cout << “Resto: ” << R;
    getchar();
}
```

Verificando Paridade

```
#include <stdio>
#include <iostream>
using na
int ma
    int
    co
    cin
    R = N1 % 2;
    cout << “Resto: ” << R;
    getchar();
}
```

Como imprimir “Par” se o número é par e “Ímpar” se o número é ímpar?

Conversão de Segundos para H:M:S

- Convertendo 54.346 segundos em
  - Horas, Minutos e Segundos
- Sabemos que 1 hora tem 3.600 segundos
- E que 1 minuto tem 60 segundos...

- Se realizarmos a divisão...

  54.346 / 3.600 = 15,09611111... horas

Conversão de Segundos para H:M:S

54.346 / 3.600 = 15,09611111... Horas

- Isso significa que, em 54.346 segundos, temos:
  - 15 horas completas e...
  - ...uma quantidade de segundos que não completa uma hora (não chegam a 3.600)
- Para saber quantos segundos sobraram, usamos o resto de divisão por 3.600

54.346 % 3.600 = 346 segundos

Conversão de Segundos para H:M:S

54.346 % 3.600 = 346 segundos

- Então, 54.346s = 15 horas e 346 segundos
  - É assim que representamos usualmente?
- **Faltam os minutos!**
- Quantos minutos há em 346 segundos?

346 / 60 = 5,76666666.... minutos

Conversão de Segundos para H:M:S

346 / 60 = 5,76666666.... minutos

- Isso significa que 346 segundos significam:
  - 5 minutos e...
  - ... Uma quantidade de segundos (que não chegam a 1 minuto)
- Podemos achar o resto de divisão de 346 por 60 para calcular quantos segundos restaram
- 346 % 60 = 46 segundos

Conversão de Segundos para H:M:S

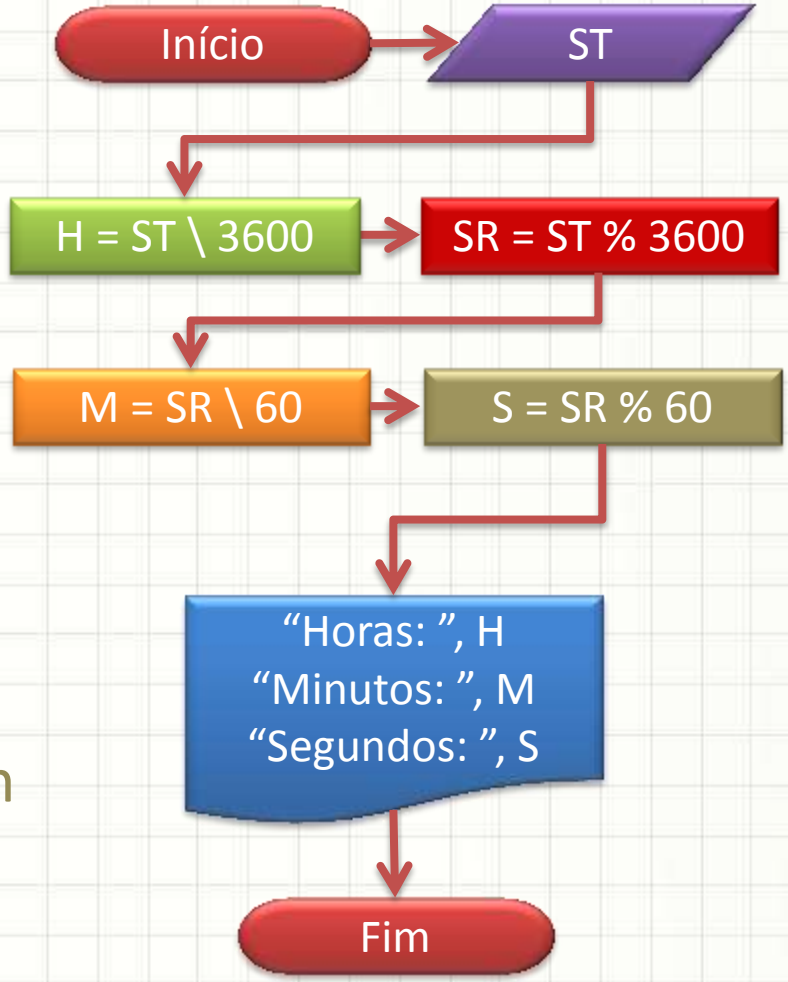
- Resumindo
  - 54.346 / 3.600 = **15**,09611111... horas
  - 54.346 % 3.600 = 346 segundos
  - 346 / 60 = **5**,76666666.... minutos
  - 346 % 60 = **46** segundos
- Assim:
  - 54.346s = 15h, 5min, 46s
- Como fazer um programa que calcule isso?

Conversão de Segundos para H:M:S

- Linguagem Natural

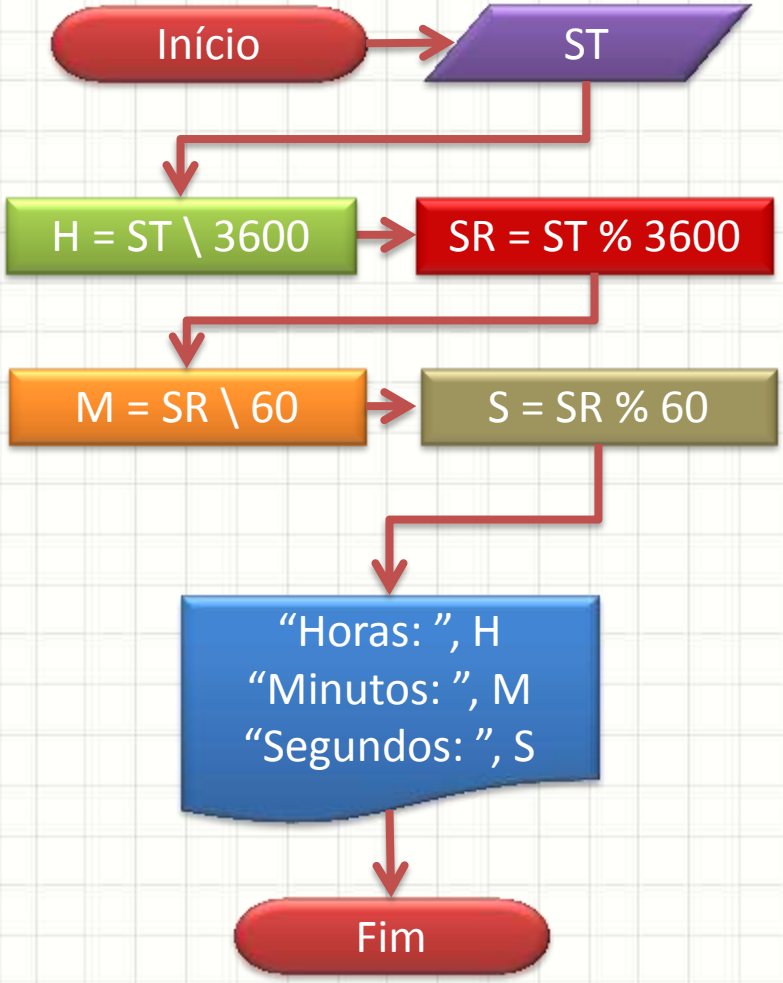
1. Leia o número de segundos totais
2. Calcule o número de horas dividindo por 3600 (divisão inteira)
3. Calcule os segundos restantes com o resto de divisão por 3600
4. Calcule o número de minutos dividindo o resto anterior por 60 (divisão inteira)
5. Calcule os segundos finais com o resto de divisão por 60
6. Imprima o número de horas, minutos e segundos

- Fluxograma



Conversão de Segundos para H:M:S

- Fluxograma



- Portugol

```
Algoritmo “Número de Semanas”
Var
ST, H, SR, M, S: INTEIRO
Inicio
Escreva(“Quantos segundos? ”)
Leia(ST)
H ← ST \ 3600
SR ← ST % 3600
M ← SR \ 60
S ← SR % 60
Escreval(“Horas: ”, H)
Escreval(“Minutos: ”, M)
Escreval(“Segundos: ”, S)
FimAlgoritmo
```

Conversão de Segundos para H:M:S

```
Algoritmo "Número de Semanas"
Var
ST, H, SR, M, S: INTEIRO
Inicio
Escreva("Quantos segundos? ")
Leia(ST)
H ← ST \ 3600
SR ← ST % 3600
M ← SR \ 60
S ← SR % 60
Escreval("Horas: ", H)
Escreval("Minutos: ", M)
Escreval("Segundos: ", S)
FimAlgoritmo
```

Conversão de Segundos para H:M:S

- Portugol

```
Algoritmo "Número de Semanas"
Var
ST, H, SR, M, S: INTEIRO
Inicio
Escreva("Quantos segundos? ")
Leia(ST)
H ← ST \ 3600
SR ← ST % 3600
M ← SR \ 60
S ← SR % 60
Escreval("Horas: ", H)
Escreval("Minutos: ", M)
Escreval("Segundos: ", S)
FimAlgoritmo
```

- Linguagem C

```
#include <stdio>
#include <iostream>
using namespace std;
int main(void)  {
    int ST, H, SR, M, S;
    cout << "Quantos segundos? ";
    cin >> ST;
    H = ST / 3600;
    SR = ST % 3600;
    M = SR / 60;
    S = SR % 60;
   cout << "Horas: " << H << endl;
   cout << "Minutos: " << M << endl;
   cout << "Segundos: " << S << endl;
getchar();
}
```

Conversão de Segundos para H:M:S

```
#include <stdio>
#include <iostream>
using namespace std;
int main(void)  {
    int ST, H, SR, M, S;
    cout << “Quantos segundos? ”;
    cin >> ST;
    H = ST / 3600;
    SR = ST % 3600;
    M = SR / 60;
    S = SR % 60;
   cout << “Horas: ” << H << endl;
   cout << “Minutos: ” << M << endl;
   cout << “Segundos: ” << S << endl;
getchar();
}
```

Funções Matemáticas

Funções Matemáticas

- Que bando de cálculo “fuleiro”!
  - Onde estão as contas complexas?
  - Cadê o logaritmo, a raiz quadrada e outros?
- Esses “caras” são chamados de **funções**
- Existem várias funções prontas no C/C++

Funções Matemáticas

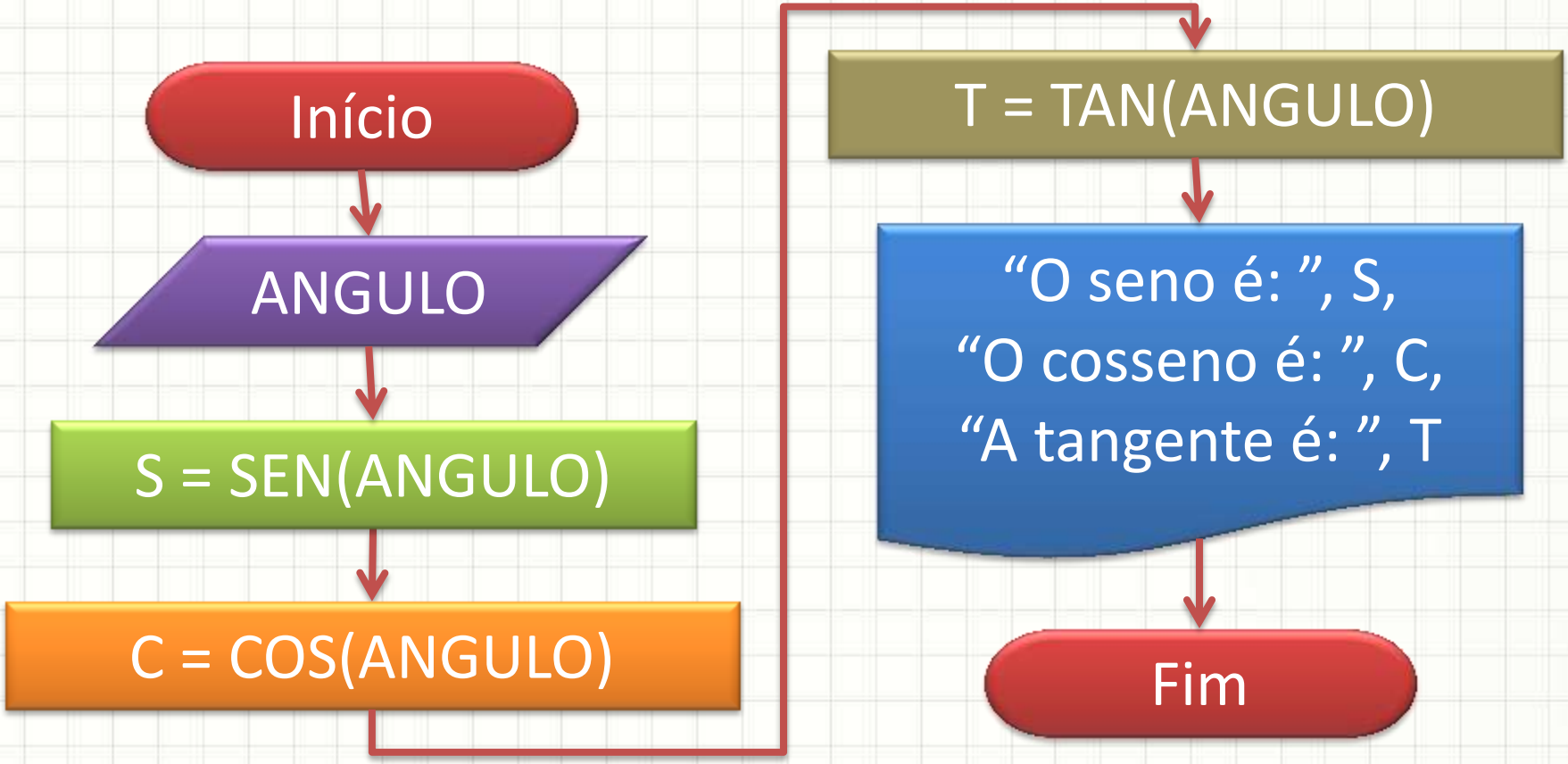
Portugol	C/C++	Comentário
-	ceil(x)	Devolve o valor de x arredondado para cima
cos(x)	cos(x)	Devolve o cosseno de x (com x em radianos)
-	exp(x)	Devolve o valor de e_x
abs(x)	abs(x)	Devolve o valor absoluto (sem sinal) de x
-	floor(x)	Devovle o valor de x arredondado para baixo
sen(x)	sin (x)	Devolve o seno de x (com x em radianos)
raizq(x)	sqrt(x)	Devolve a raiz quadrada de x
tan(x)	tan(x)	Devolve a tangente de x (com x em radianos)
Pi	4 * atan(1.0)	Devolve o valor de PI (3,141592...)

Senos, Cossenos e Tangentes

- Como um exemplo, vamos calcular senos, cossenos e tangentes

1. Vamos ler um ângulo
2. Vamos calcular os valores
3. Vamos imprimir os valores

Senos, Cossenos e Tangentes



Senos, Cossenos e Tangentes - P

```
Algoritmo "Senos, cossenos e tangente"
Var
ANGULO, S, C, T: REAL
Inicio
Escreva("Digite um ângulo – 0 a 2*PI: ");
Leia(ANGULO)
S ← sen(ANGULO)
C ← cos(ANGULO)
T ← tan(ANGULO)
Escreval("Seno: ", S)
Escreval("Cosseno: ", C)
Escreval("Tangente: ", T)
FimAlgoritmo
```

Senos, Cossenos e Tangentes - C

```
#include <stdio>
#include <iostream>
#include <math>
using namespace std;
int main(void)  {
    float ANGULO, S, C, T;
     cout << “Digite um ângulo – 0 a 2*PI: ”;
     cin >> ANGULO;
     S = sin(ANGULO);
     C = cos(ANGULO);
     T = tan(ANGULO);
     cout << “Seno: ” << S << endl;
     cout << “Cosseno: ” << C << endl;
     cout << “Tangente: ” << T << endl;
getchar();
}
```

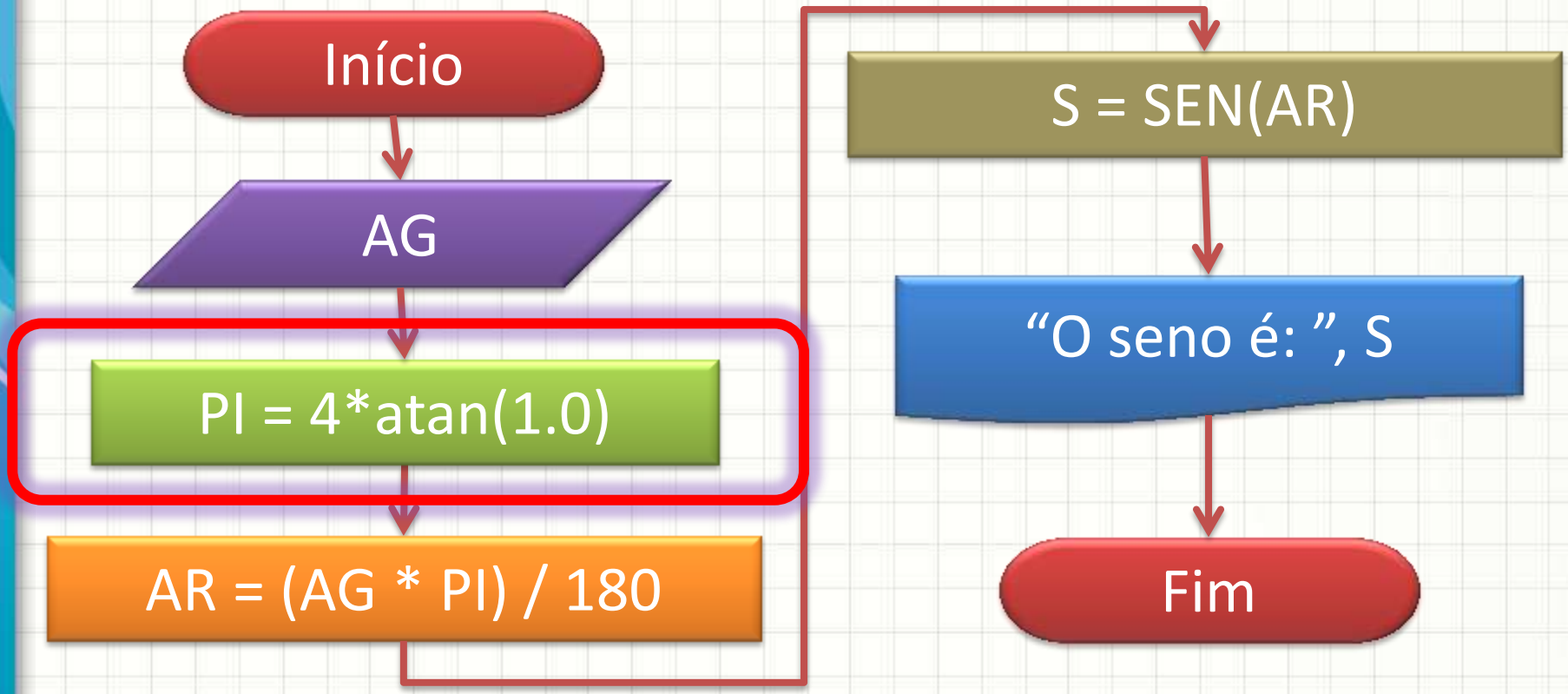
Senos, Cossenos e Tangentes

- E se quisermos ler o ângulo em GRAUS?

1. Vamos ler um ângulo (em graus)
2. **Vamos convertê-lo em radianos**
3. Vamos calcular os valores
4. Vamos imprimir os valores

- ANGULO = (ANGULO_GRAUS * PI) / 180
- PI = 4 * ATAN(1.0)

Senos, Cossenos e Tangentes



DESNECESSÁRIO NO PORTUGOL!

Senos, Cossenos e Tangentes - P

```
Algoritmo "Ângulo em graus"
Var
AG, AR, S: REAL
Inicio
Escreva("Digite um ângulo – 0 a 360: ");
Leia(AG)
AR ← (AG * PI) / 180
S ← sen(AR)
Escreval("O seno é: ", S)
FimAlgoritmo
```

Senos, Cossenos e Tangentes - P

```
#include <stdio>
#include <iostream>
#include <math>
using namespace std;
int main(void)  {
    float AG, AR, PI, S;
    cout << "Digite um ângulo – 0 a 360: ";
    cin >> AG;
    PI = 4.0 * atan(1.0);
    AR = (AG * PI) / 180.0;
    S = sin(AR);
    cout << "O seno é: " << S << endl;
getchar();
}
```

ARREDONDAMENTO

Função de Arredondamento

- Ainda que o Portugol não forneça uma função de arredondamento, ela existe no C/C++

```
float valor, arredondado;
valor = 1.55;
arredondado = floor(valor);
```

- Qual o valor de **arredondado**?
- **arredondado** vale **1.0**!
- Arredondamento em C: funções limitadas!

Arredondando Números - C

```
#include <stdio>
#include <iostream>
#include <math>
using namespace std;
int main(void)  {
    float NUM, ARRED;
    cout << “Digite um número fracionário: ”;
    cin >> NUM;
    ARRED = floor(NUM);
    cout << “Arredondado para: ” << ARRED << endl;
getchar();
}
```

Função de Arredondamento

- **floor** arredonda para o inteiro anterior
- **ceil** arredonda para o próximo inteiro

- Para pensar:
  - Como arredondar seguindo a regra padrão?
  - Como fazer para arredondar com 1 casa decimal?
  - E com 2 casas decimais?
  - E com 3?

Conclusões

Resumo

- O uso de resto de divisão é útil para fracionar números em unidades menores e para verificar divisibilidade
- As bibliotecas do C/C++ fornecem uma porção de cálculos matemáticos prontos para serem usados

- **TAREFA!**
  - **Lista de Exercícios 1**

Próxima Aula

- Como posso reaproveitar algoritmos complicados?
  - Vou precisar reprogramar todas as vezes?

PERGUNTAS?

Bom Descanso a Todos!