

The background is a light cream color with a repeating pattern of colorful, stylized dinosaurs and plants. The dinosaurs include a pink pterosaur in the top left, a green long-necked dinosaur in the top center, a purple triceratops in the top center-right, an orange T-Rex in the top right, a red Stegosaurus on the left, a blue long-necked dinosaur on the right, an orange T-Rex on the bottom left, a green long-necked dinosaur on the bottom right, a blue long-necked dinosaur in the bottom center-left, a purple triceratops in the bottom center-right, and a pink pterosaur in the bottom right. There are also several green leaves and blue eggs scattered throughout.

C

Lógica de programação

Linguagem de programação



É uma linguagem formal que, através de uma série de instruções, permite que um programador escreva um conjunto de ordens, ações consecutivas, dados e algoritmos para criar programas que controlam o comportamento físico e lógico de uma máquina.

Em palavras mais simplificadas a linguagem de programação é como um idioma, com uma quantidade menor de palavras e escrita lógica.



Algoritmos

São conjuntos de passos finitos e organizados que, quando executados resolvem um determinado problema.



1. Bases da linguagem



• Comentários

- Para todas as linguagem o comentário tem um único sentido, deixar um breve resumo para que serve aquela parte do código ou onde começar porque se alguém for modificar ou editar seu código acha mais fácil o caminho.

// Duas barras são comentários em linha, a partir delas tudo que for escrito é

// comentário, mas apenas na linha

/*

Tudo que está dentro de barra e asterisco é comentário,

os comentários não são lidos pelo algoritmo.

*/



Estrutura da linguagem

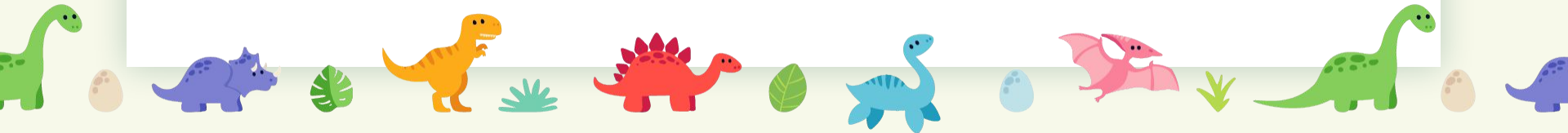
```
# include <stdio.h>
```

```
int main(){
```

```
    // Corpo do algoritmo
```

```
    return 0;
```

```
}
```





Estrutura da linguagem

```
# include <stdio.h> //Incluir Bibliotecas
```

```
// Esta biblioteca contém várias funções de entrada e saída
```

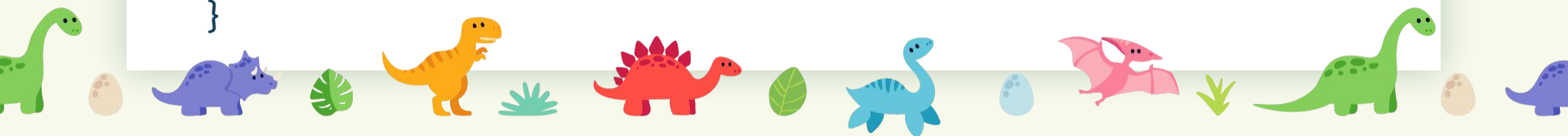
```
int main(){ // Está é a função principal (main) do nosso algoritmo
```

```
// A main é o ponto de entrada principal do programa
```

```
return 0; // Este é o valor retornado pela função
```

```
// Retorna um valor inteiro, no caso 0, pois nossa função é tipo int
```

```
}
```



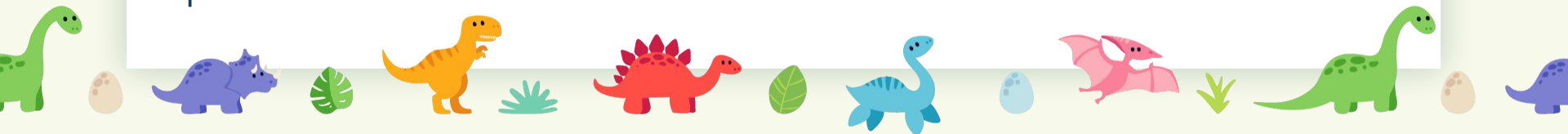


Bibliotecas

Quando você usa uma linguagem de programação existe uma possibilidade de se usar funções pré-escritas por outro programador, essas funções resolvem certos problemas sem que você precise reinventar um código. E esse conjunto de funções se chama biblioteca. É basicamente funções que já existem no programa e que você pode usar.

Exemplo:
`<math.h>`

*Biblioteca que tem
funções matemáticas
uma de suas funções é a
`sqrt()` que calcula raiz
quadrada.*





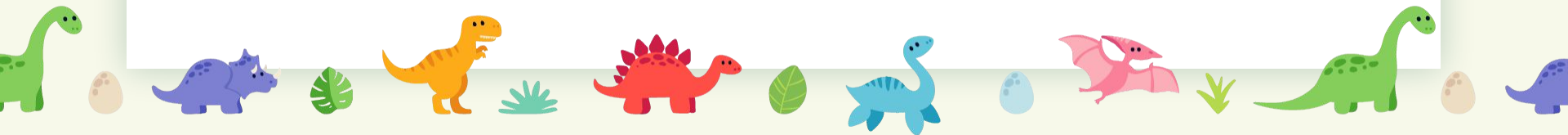
Outros tipos de funções

`void main() { }` // Função sem retorno -- void == vazio

`int main() { return 0; }` // Função Main tipo inteiro

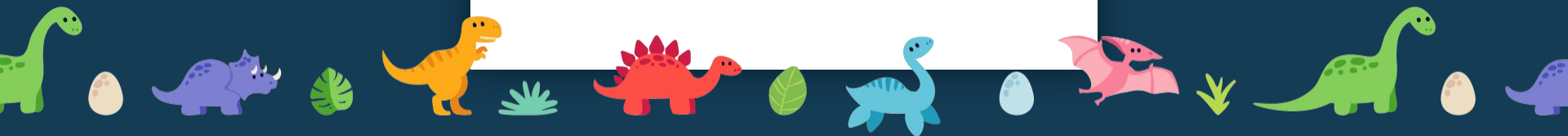
`Float main() { return 0; }` // Função Main tipo real

`double main() { return 0; }` // Função Main tipo real (maior)





Algoritmo dizer ola





Estrutura da linguagem

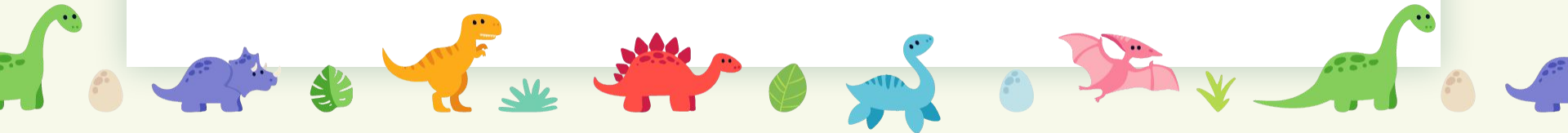
```
# include <stdio.h>
```

```
void main(){
```

```
    // '\n' eh quebra de linha
```

```
    printf("Hello, it's me the good advice capcuck \n\n");
```

```
}
```



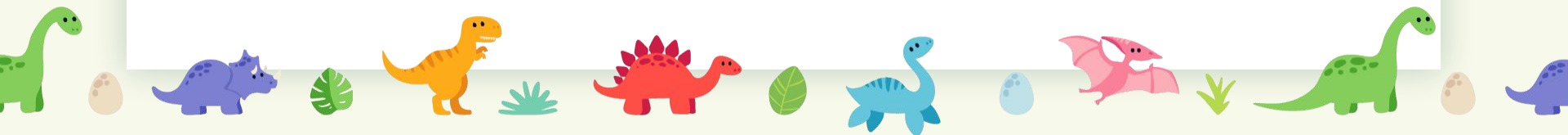


Variável

Na programação, uma variável é um objeto capaz de reter e representar um valor ou expressão. Enquanto as variáveis só "existem" em tempo de execução, elas são associadas a "nomes", chamados identificadores, durante o tempo de desenvolvimento do programa.



<i>tipo</i>	<i>Leitura com scanf</i>
<i>char</i>	<code>%c</code>
<i>int</i>	<code>%d</code> ou <code>%i</code>
<i>float</i>	<code>%f</code>
<i>double</i>	<code>%lf</code> ou <code>%f</code>





Tempo de vida de uma variável

escopo é um contexto delimitante
aos quais valores e expressões
estão associados

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

```
void main(){ ← // início
```

```
    Int num = 0; // variável do tipo int com nome - num
```

```
    while (num < 10){ ←
```

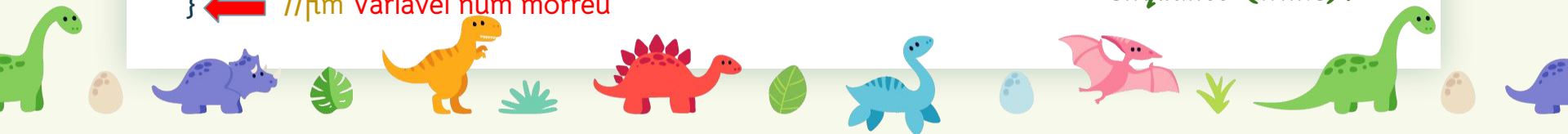
```
        num++; // num = num + 1
```

```
        Int num2 = num;
```

```
    } ← //Variável num2 morreu
```

```
} ← //fim Variável num morreu
```

*A variável num
pode ser usado
durante toda a
main, já que foi
criada nesse
escopo, já a
variável num2 só
pode ser usada
dentro o escopo do
enquanto (while).*





Scanf e printf

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

```
void main(){ // inicio
```

```
    Int n1;
```

```
    // Função para exibir na tela
```

```
        printf(" Digite a primeira nota (valor inteiro): \n ");
```

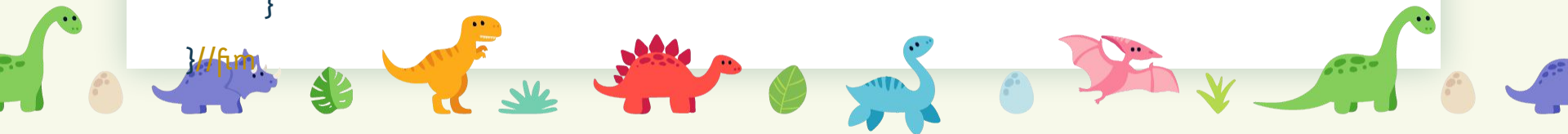
```
    //Função para o usuário digitar o valor pedido
```

```
        scanf("%d", &n1);
```

```
    }
```

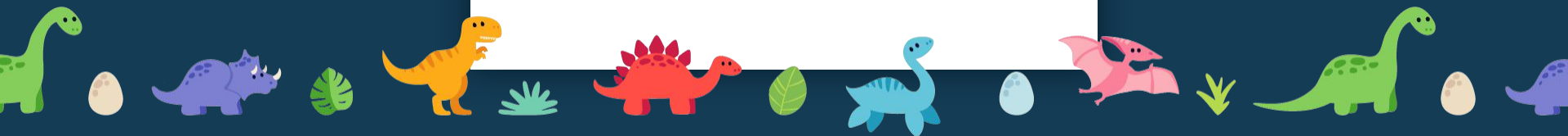
```
    //fim
```

*Uma variável tem
que ser criada
antes que você dê
valor a ela ou a
chame em algum
lugar no programa.*





Estrutura de seleção

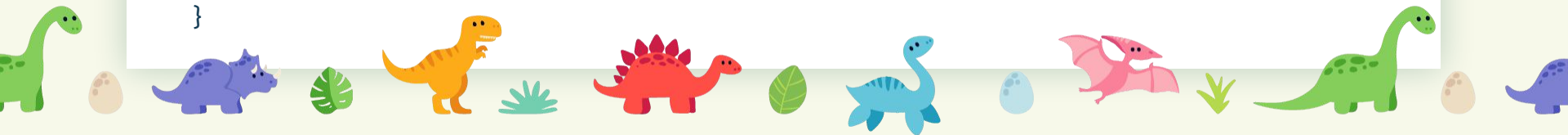




Estrutura - Se e se não(if else)

```
# include <stdio.h>

void main(){
    int num = 10;
    if(num < 10){ //Se a condição for verdadeira
        printf("É verdade esse bilete"); //Exibe essa mensagem na tela
    } else{ //Se não for verdadeira
        printf("Num é verdade esse bilete"); //Exibe essa mensagem na tela
    }
}
```





If else

A condição do if é um teste lógico, se ele for true é executado

Nem todo if tem um else, mas todo else tem um if;

Um If para um else
`if(condição){`

`} else{`

`}`

Varios If's sem else
`if(condição){`

`} if(condição){`

`} if(condição){`

`}`

Varios If's para um else
`if(condição){`


`} else if(condição){`

`} else if(condição){`

`} else {`

`}`

Estrutura - escolha caso (switch case)



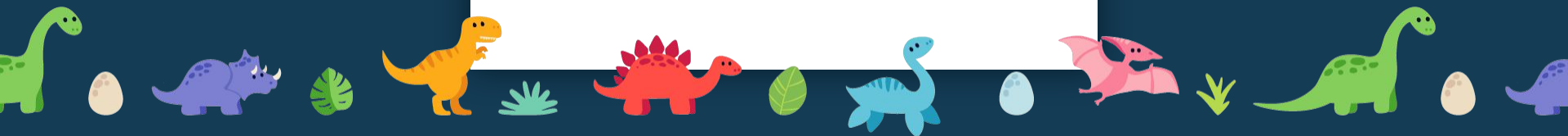
```
#include <stdio.h>

void main(){
    int num = 1;
    switch (resultado) { //A variável resultado vai determinar qual caso será escolhido
        // Determine os casos que poderão ocorrer
        case 1: // Se resultado igual a 1, então
            printf("Você escolheu um");
            break;
        default: // Caso não seja um caso definido
            printf("Você não escolheu uma opção válida;");
            break;
    }
```

Escolha caso é semelhante com o if e else, só que é normalmente usado para casos mais específicos, com um número pequeno de casos. O case é como se e o default como else.



Laço de repetição





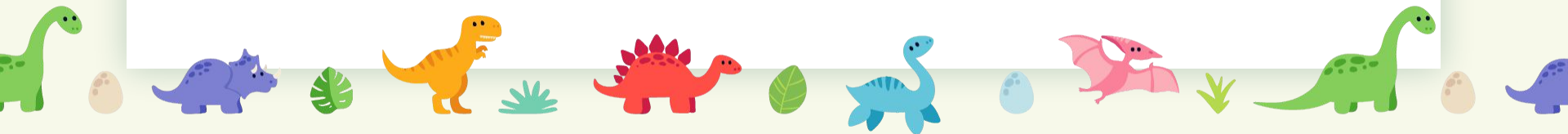
While - Enquanto

```
# include <stdio.h>

void main(){ // inicio
    Int num = 0; // variável do tipo int com nome - num
    while (num < 10){
        num++; // num = num + 1 (para não criar um loop infinito)
    }
} //fim
```

Enquanto a condição (o teste lógico), for verdadeiro o laço de repetição será executado.

Loop infinito é quando a condição nunca deixa de ser verdadeira





For - para faça

```
# include <stdio.h>

void main(){ // inicio

    // Para n de 0 até 9 faça n recebe n + 1
    for (int n = 0; n < 10; n++){ /
    }

} //fim
```

num laço for, já temos uma inicialização de uma variável de controle e a iteração obrigatória, o que garante certa “segurança” na repetição, visto que no while podemos esquecer de fazer isso

