(1960 - 1970) - Fortran

- Como ocorrem as atribuições, declarações e expressões
  As variáveis podem ser inteiras, reais ou literais. A declaração de uma variável deve vir antes que ela seja usada, se isto não ocorrer o compilador assumirá que as variáveis que começam com as letras I até N como inteiras (INTEGER*4) e todas as outras como reais (REAL*4).
  Esta forma de declaração implícita pode ser modificada usando o comando 'implicit tipo (a1-a2,b1-b2,...)' sendo a1, a2, b1, b2 quaisquer letras do alfabeto. A vírgula separa os intervalos de letras e o sinal – determina o intervalo. As letras que não estiverem em nenhum dos intervalos terá o seu tipo dado pela declaração implícita. O comando seguinte indica que as variáveis que começam com as letras a, b, c e de r até z são do tipo TIPO1. implicit TIPO1 (a, b, c, r-z) os espaços são usados para dar clareza e são ignorados pelo compilador.

  Quando não se deseja que nenhuma variável seja declarada implicitamente usa-se o comando 'implicit none'. Se este comando for usado e uma variável citada no programa não estiver em nenhuma outra declaração o compilador acusará um erro.

  Para se declarar variáveis que sejam matrizes e vetores deve-se indicar suas dimensões logo após o nome da variável; entre parênteses, e separadas umas das outras por vírgula. Ex.: a(4,3) indica uma matriz 'a' de 4 linhas por 3 colunas. As variáveis podem receber valores iniciais usando '/valor/', logo após sua declaração. No caso de vetores e matrizes devem ser dados os valores para todos os elementos de cada linha em sequência.

- Estruturas de Controle de Fluxo
  As estruturas básicas de controle são:

  1. Estrutura sequencial.
  2. Estrutura de tomada de decisão (if-then-else if-else-endif).
  3. Estrutura de laço:
     faça-enquanto (while-loop).
     Estrutura de laço repetição (repeat-loop).
  4. Estrutura de seleção de caso (select case).

  Os comandos GO TO transferem o controle de fluxo para um comando executável na mesma unidade de programa. Os três tipos de GO TO são:

  1. O comando GO TO incondicional transfere o controle para o comando identificado pelo número de comando especificado no GO TO.

  a forma geral do comando GO TO é

```
go to n
```

sendo que 'n' é um número de comando executável

- Abstração procedural

(1970 - 1980) - Pascal

- Como ocorrem as atribuicoes, declaracoes e expressoes

São os nomes atribuídos para as constantes e variáveis que receberão as informações.

Um identificador válido na linguagem Pascal é qualquer seqüência de caracteres que obedeça às seguintes regras:

1. Devem começar por um caractere alfabético(a, b, ..., z) ;
2. Podem ser seguidos por mais caracteres e/ou numéricos, ou o caractere _ ;
3. Não é permitido o uso de caracteres especiais;
4. Têm que ser diferentes das palavras reservadas do Pascal, como os comandos, funções, etc.
   A declaração de variáveis e constantes consiste em reservar um espaço na memória do micro para armazenar um certo tipo de informações, associando a este espaço um tipo de dados e uma identificação (nome da variável/constante).
   A declaração de variáveis e constantes consiste em reservar um espaço na memória do micro para armazenar um certo tipo de informações, associando a este espaço um tipo de dados e uma identificação (nome da variável/constante).
   No programa em Pascal temos três tipos de dados principais: Numéricos (inteiros e reais), caracteres (letras e símbolos) e lógicos (verdadeiro ou falso). Os tipos devem ser declarados no início do programa.

```
program exemplo;
var
 nome:string;
 num,cont:integer;
begin
 {corpo do programa}
end;
```

- Estruturas de Controle de Fluxo
  Essa estrutura é conhecida também como estrutura de decisão ou de seleção, ela se caracteriza por execução de determinados códigos de programação dependendo da veracidade de uma condição. Neste tópico são apresentadas as estruturas de condição simples, composta, aninhada (uma dentro da outra) e a estrutura Case.

```
   var X, Y : Integer;

   begin
     showMessage ('Entre com dois Números:'); //Mostra na tela a mensagem
     readln (X, Y); // Lê os dois números
     if (X = Y) then // Condição - SE X for igual a Y
     begin  //INICIO - Repare abaixo que existe duas instruções, dois
comandos Writeln, por isso estão entre BEGIN e END
       showMessage('X é igual que Y');
       showMessage('O Valor de X é =', X);
     end; // FIM
   end.
   // ------------------------------------------------------------
   var
    X : Integer;

   begin
     readln (X);
     case X of
          1 : Writeln ('Ola Mundo');//  E o valor de X for igual a 1, irá
executar essa linha
          2 : Writeln ('GNOIA');  //  X = 2, essa linha será executada
```

```
          3 : Writeln ('Software Livre'); // X = 3 - essa linha será
executada
     end;
   end.
```

- Abstracao procedural

(1980 - 1990) - C

- Como ocorrem as atribuicoes, declaracoes e expressoes

  Uma "definição" de uma variável estabelece as mesmas associações que uma declaração, mas também faz com que ocorra a alocação de armazenamento para a variável.
  Por exemplo, as funções main, find e count e as variáveis var e val são definidas em um arquivo de origem, nesta ordem:

```
int main() {}

int var = 0;
double val[MAXVAL];
char find( fileptr ) {}
int count( double f ) {}
```

- Estruturas de Controle de Fluxo

  Dificilmente um programa em C irá executar sempre as mesmas instruções, na mesma ordem, independentemente do que tenha acontecido anteriormente ou do valor que foi fornecido. É muito comum que alguém queira que um pedaço de código só seja executado se uma certa condição for verdadeira; também é comum querer que um pedaço de código seja repetido várias vezes, de tal maneira que simplesmente copiar o código não resolveria o problema ou seria trabalhoso demais. Para casos como esses, existem as estruturas de controle de fluxo.

  Em C, existem várias instruções relacionadas ao controle de fluxo:

  if, que executa um bloco apenas se uma condição for verdadeira;
  switch, que executa um bloco de acordo com o valor de uma expressão ou variável;
  for, que executa um bloco repetidas vezes enquanto uma condição for verdadeira, executando uma instrução (geralmente de incremento ou decremento de uma variável) após cada execução;
  while, que executa um bloco enquanto uma condição for verdadeira;
  do, semelhante ao while, mas a condição é avaliada após a execução (e não antes);
  goto, que simplesmente pula para um lugar pré-definido.

- Abstracao procedural

(1990 - 2000) - C++

- Como ocorrem as atribuicoes, declaracoes e expressoes

  Para declarar uma variável, basta indicar o seu tipo seguido do seu nome. Existem várias convenções sobre o nome das variáveis. Alguns preferem separar as diferentes partes do nome com _. Outros preferem escrever uma letra maiúscula para separá-los.

```
int minha_variavel;
int variavel_2 = 100;
```

- Estruturas de Controle de Fluxo

  Em C++ os métodos de tomada de decisão presentes na linguagem C estão disponíveis para as tarefas mais corriqueiras que o programa deve executar. Além desta forma de controle de decisões, C++ provê certas funcionalidades relacionadas a objetos que modificam a forma como o código é estruturado e, por consequência, decidem como o programa deve se comportar em determinadas situações. Examinemos os métodos básicos e analisemos de forma simples as estruturas de decisão presentes no modelo de programação orientado a objetos, para entender como isso poderá nos ajudar a tornar o código mais bem construído.

  De modo geral, a maioria das linguagens de programação existentes utiliza-se das estruturas if-else ou switch-case

```
#include <iostream>
using namespace std;
int main(void)
 {
   if(deposito < 20) {
        cout << "Depósito de gasóleo inferior a 20%";
    } else if(deposito < 50) {
        cout << "Tem menos de metade do depósito";
    } else {
        cout << "Ainda tem meio depósito ou mais";
    }
   char grade;
   cout << "Enter your grade (A to F): ";
   cin >> grade;
   switch (grade)
   {
   case 'A':
            cout << "Your average must be between 90 - 100"<< endl;
            break;
   case 'B':
            cout << "Your average must be between 80 - 89"<< endl;
            break;
   case 'C':
            cout << "Your average must be between 70 - 79"<< endl;
            break;
   case 'D':
            cout << "Your average must be between 60 - 69"<< endl;
            break;
   default:
            cout << "Your average must be below 60" << endl;
   }
   return 0;
 }
```

- Abstracao procedural

(2000 - 2010) - PHP

- Como ocorrem as atribuicoes, declaracoes e expressoes

  A declaração de variáveis em PHP é bastante simples. Como a linguagem é fracamente tipada, não é necessário informar o tipo de dado na declaração.

```
// variável do tipo String
$nome = 'Meu nome 123';

// variável do tipo inteiro
$ano = 2017;

// variável do tipo float
$pi = 3.14159265;

// variável do tipo booleano
```

Ao declarar uma variável dentro de uma classe, antes do nome atribuído é comum especificar o modificador de acesso.

```
class Exemplo {
  private $variavelPrivada = 'Private';
  public $variavelPublica = 'Public';
  protected $variavelProtegida = 'Protected';
}
```

- Estruturas de Controle de Fluxo

Os operadores condicionais são um dos recursos mais básicos da programação e são essenciais no desenvolvimento da lógica de qualquer sistema ou aplicação. São utilizados quando é necessário que determinado código seja executado apenas se atender a uma condição. Veremos neste artigo as estruturas de condição if/else, elseif (ou else if) e operador ternário.

Sintaxe do if/else:

```
<?php

$a = 8;
$b = 2;

if ($a/$b==2) {

  echo "O resultado da divisão é 2";

} elseif ($a/$b==4) {

  echo "O resultado da divisão é 4";

} else {

  echo "O resultado da divisão não é 2 nem 4";

}
?>
```

While é a estrutura de repetição mais simples do PHP e com ele informamos que um bloco de código deve ser repetido enquanto a condição declarada for verdadeira. É um comando de repetição (laço) simples, pois executa um conjunto de comandos até que uma determinada condição seja atendida. Veja abaixo a sintaxe do comando:

```
$i = 1;
while ($i <= 10) {
    echo $i;
    $i++;
}
```

O for é a estrutura de repetição do PHP que utilizamos quando sabemos a quantidade de repetições que devem ser executadas. A sua sintaxe é apresentada abaixo:

```
for ($i=0; $i < 10; $i++) {
  echo $i."\n";
}
```

Com o comando FOREACH trabalhamos com coleções de dados, ou seja, vetores e matrizes, funcionando a partir da versão 4 do PHP.

```
<?php
$meu_vetor = array("A", "B", "C", "D", "E");
   foreach ($meu_vetor as $elemento){
   echo $elemento;
 }
?>
```

Diferente da estrutura if/else, o switch/case avalia apenas condições de igualdade. Ou seja, ela verifica se o valor recebido como parâmetro é igual a alguma das opções especificadas em seu corpo. Esse comportamento equivale à utilização de vários if/else em sequência, porém, com uma sintaxe mais enxuta e de fácil leitura.

```
$escolhaNumero = 2;

switch ($escolhaNumero) {
  case 0:

    echo "numero vale 0";
    break;

  case 1:

    echo "numero vale 1";
    break;

  case 2:

    echo "numero vale 2";
  break;
}
```

- Abstracao procedural

(2010 - 2020) - Swift

- Como ocorrem as atribuicoes, declaracoes e expressoes

Os objetos em Swift, assim como em demais linguagens de programação, são instâncias de classes declaradas. Quando um objeto ainda não está instanciado, ele é nulo (**nil**, no caso do Swift). Objetos estáticos são instanciados com o comando **let**, enquanto objetos variáveis são instanciados com o comando *var.* Ao instanciar uma classe usando o construtor padrão, usamos (), ou um valor default.

```
var annotation:POIAnnotation = POIAnnotation()
  var nome:String = “”
  let nomeFixo:String = “Ola Sr.”
  var arr:Array = []
  var arr2:Array = ["item1", "item2", "item3"]
  var dic1:Dictionary = [”chave” : ”valor"]
  var dic2:Dictionary = Dictionary<String, String>()
```

- Estruturas de Controle de Fluxo

A estrutura de repetição *for-in* é utilizada — e muito — quando precisamos percorrer uma determinada coleção, itens dentro de uma sequência, intervalo ou progressão.

```
let semestre = ["Janeiro", "Fevereiro", "Março", "Abril", "Maio", "Junho"]

for mes in semestre{
    println("Estamos no mês \(mes)")

}
```

Utilizamos o laço `for` quando desejamos que um determinado bloco de código seja percorrido repetidas vezes até que uma condição declarada na criação do laço seja atendida.

```
for var x = 1; x <= 10; x++ {
    println(x)
}
```

O `while` é responsável por executar um bloco de instruções enquanto a sua condição for verdadeira. A diferença é que fazemos as operações dentro do bloco para atendermos à condição da declaração.

```
var y = 1

while y <= 10 {
    println("Rodei \(y) vezes")
    y++
}
```

- Abstracao procedural