Desenvolvimento de Aplicações Android

SENAI

Conteúdo do Curso

- Linguagem Utilizada
- Algoritmos e Lógica
- Tipos de Dados
- Variáveis e Constantes
- Conversão entre Dados
- Operadores
- Estruturas de Decisão e Repetição
- Funções

SENAI

Linguagem Utilizada

Kotlin

É uma linguagem de programação código aberto multiplataforma criada pela JetBrains e amplamente usada por desenvolvedores Android. Totalmente orientada a objetos, mas com algumas características de linguagem funcional, com tipagem estática e executada pela JVM (Java Virtual Machine).

Tipagem estática - Precisa informar explicitamente o tipo de cada dado utilizado no sistema: variáveis, parâmetros de funções, valores de retorno, etc.

SENAI

Linguagem Utilizada

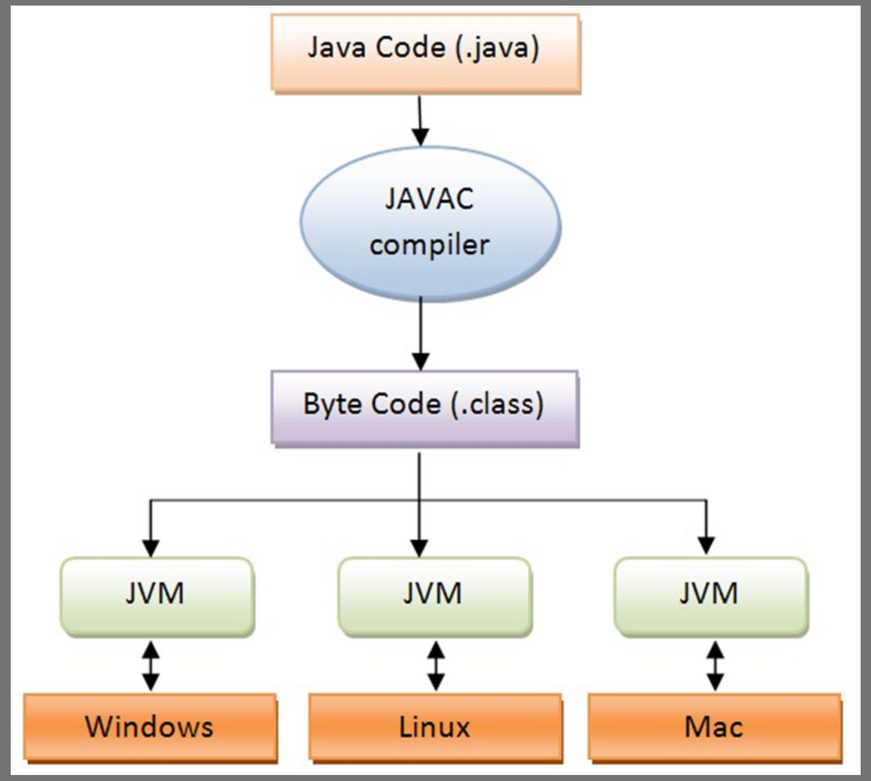
Kotlin

Java Virtual Machine é um programa que carrega e executa os aplicativos Java, convertendo os bytecodes em código executável de máquina. A JVM é responsável pelo gerenciamento dos aplicativos, à medida que são executados

SENAI

Linguagem Utilizada

Kotlin



SENAI

Linguagem Utilizada

Kotlin

As semelhanças com Java são tantas que você até pode usar Java dentro do seu código Kotlin.

Foi lançada, na sua versão estável, em 2016. Logo em seguida, o Google se juntou a JetBrains e agora, as duas empresas mantém essa linguagem através da Kotlin Foundation.

Além disso, o Google anunciou que Kotlin faz parte das linguagens oficiais do Android. Então, junto com a Java Virtual Machine, ela pode ser usada para desenvolvimento web, mobile e desktop.

SENAI

Linguagem Utilizada

Kotlin

Por que utilizar Kotlin?

Em primeiro lugar, a produtividade. Kotlin é uma linguagem enxuta e intuitiva, utilizando cerca de 40% menos códigos para representar a mesma coisa que o Java.

A linguagem de programação Java é Código verboso (aquele que precisa de mais palavras), e apesar de existir projetos para diminuir essa desvantagem, como o Lombok.

Assim, ela não é tão eficaz quanto a expressividade que Kotlin oferece nativamente. A curva de aprendizagem do Kotlin é rápida e relativamente pequena.

SENAI

Linguagem Utilizada

Kotlin

Kotlin mostra os erros cometidos na hora da escrita do código durante a compilação, em vez de dar erro na hora da execução do código, muito tempo depois. Isso torna a linguagem mais rigorosa.

Quais grandes empresas também estão usando o Kotlin?

Google – Amazon– Netflix – Pinterest – Uber – Foursquare

Trello – Capital One – Coursera.

SENAI

Linguagem Utilizada

Kotlin

Comentário em Kotlin

```
/*
 Olá, este é um comentário de bloco com várias linhas.
 Esta é outra linha.
 Mais uma linha.
*/

// este é um comentário de uma linha.
```

SENAI

Algoritmos e Lógica de Programação

Algoritmo

Em matemática e ciência da computação, um algoritmo é uma sequência finita de ações executáveis que visam obter uma solução para um determinado tipo de problema. São procedimentos precisos, não ambíguos, padronizados, eficientes e corretos.

Lógica de programação

É um paradigma de programação que faz uso da lógica matemática.

Paradigma
É a representação de um padrão a ser seguido.

SENAI

Algoritmos e Lógica de Programação

Sempre que decidimos fazer qualquer atividade em nosso dia a dia, acabamos seguindo uma sequência lógica.

Na maior parte do tempo, fazemos isso de maneira tão natural que nem nos damos conta, mas, quando percebemos, conseguimos enxergar passos que levaram ao resultado final.

SENAI

Algoritmos e Lógica de Programação

Visualize a seguinte situação, você precisa fazer um bolo:

1. Selecionar os ingredientes da receita;
2. Selecionar tigela;
3. Colocar farinha, de acordo com a medida;
4. Selecionar ovos;
5. Colocar manteiga e açúcar a gosto;
6. Colocar leite;
7. Misturar todos os ingredientes na tigela;
8. Despejar a massa na forma;
9. Levar ao forno;
10. Aguardar 40 minutos;
11. Retirar do forno;
12. Servir o bolo.

SENAI

Algoritmos e Lógica de Programação

Cada pessoa define uma sequência de passos para fazer um bolo, podendo incluir ou remover alguns já definidos. Essa lógica é aplicada a qualquer coisa que fazemos diariamente e muitas das vezes não nos damos conta.

Em atividades rotineiras, não costumamos prestar atenção quando seguimos uma mesma ordem para executar tarefas. Porém, quando o assunto é programar, definir as etapas do que deve ser feito assume uma grande importância, uma vez que instruir um computador ainda é bem diferente do que instruir uma pessoa.

SENAI

Algoritmos e Lógica de Programação

Quando a necessidade é desenvolver um programa ou rotina a ser executada pelo computador, precisamos deixar bem claro a sequência que deve ser seguida para atingir o resultado esperado.

A esse encadeamento lógico na programação, chamamos de Lógica de Programação, e a descrição de como fazer, definimos como Algoritmos.

SENAI

Algoritmos e Lógica de Programação

Algoritmos

São sequências de passos que seguimos com a intenção de atingir um objetivo, pode ser desde atravessar uma rua, fazer um bolo ou definir qual critério usar para aprovar ou reprovar um aluno, por exemplo.

SENAI

Algoritmos e Lógica de Programação

O que é preciso para desenvolver um algoritmo?

No desenvolvimento de um algoritmo, devemos definir com clareza e forma precisa o conjunto de regras ou instruções que serão utilizadas para resolver aquele problema em específico.

Portanto, antes de programar, precisamos saber o que deve ser feito e planejar o passo a passo, ou seja, criar o algoritmo e avaliar se o resultado obtido é a solução esperada. Entendido isso, então definimos uma linguagem de programação para implementar nossos algoritmos.

SENAI

Algoritmos e Lógica de Programação

Como representar um algoritmo?

Existem algumas maneiras de representar algoritmos, que entram como um passo de preparação antes da programação. Podemos criar, por exemplo, uma narrativa semelhante ao exemplo do bolo, em que se descreve a sequência de execução até a obtenção do resultado desejado ou esperado.

SENAI

Algoritmos e Lógica de Programação

Para mostrar as formas de representar um algoritmo, vamos usar como exemplo um algoritmo de cálculo da média:

No nosso caso é preciso analisar as notas de 3 bimestres para a disciplina de matemática e verificar se o aluno foi aprovado ou reprovado para uma média maior ou igual a 6:

1. Obter a nota do 1º bimestre; 2. Obter a nota do 2º bimestre; 3. Obter a nota do 3º bimestre; 4. Realizar o cálculo da média para cada aluno (maior ou igual a 6 para aprovação); 5. Informar se o aluno foi "aprovado ou reprovado"; 6. Informar a média obtida pelo aluno.

SENAI

Algoritmos e Lógica de Programação

Para mostrar as formas de representar um algoritmo, vamos usar como exemplo um algoritmo de cálculo da média:

No nosso caso é preciso analisar as notas de 3 bimestres para a disciplina de matemática e verificar se o aluno foi aprovado ou reprovado para uma média maior ou igual a 6:

1. Obter a nota do 1º bimestre; 2. Obter a nota do 2º bimestre; 3. Obter a nota do 3º bimestre; 4. Realizar o cálculo da média para cada aluno (maior ou igual a 6 para aprovação); 5. Informar se o aluno foi “aprovado ou reprovado”; 6. Informar a média obtida pelo aluno.

SENAI

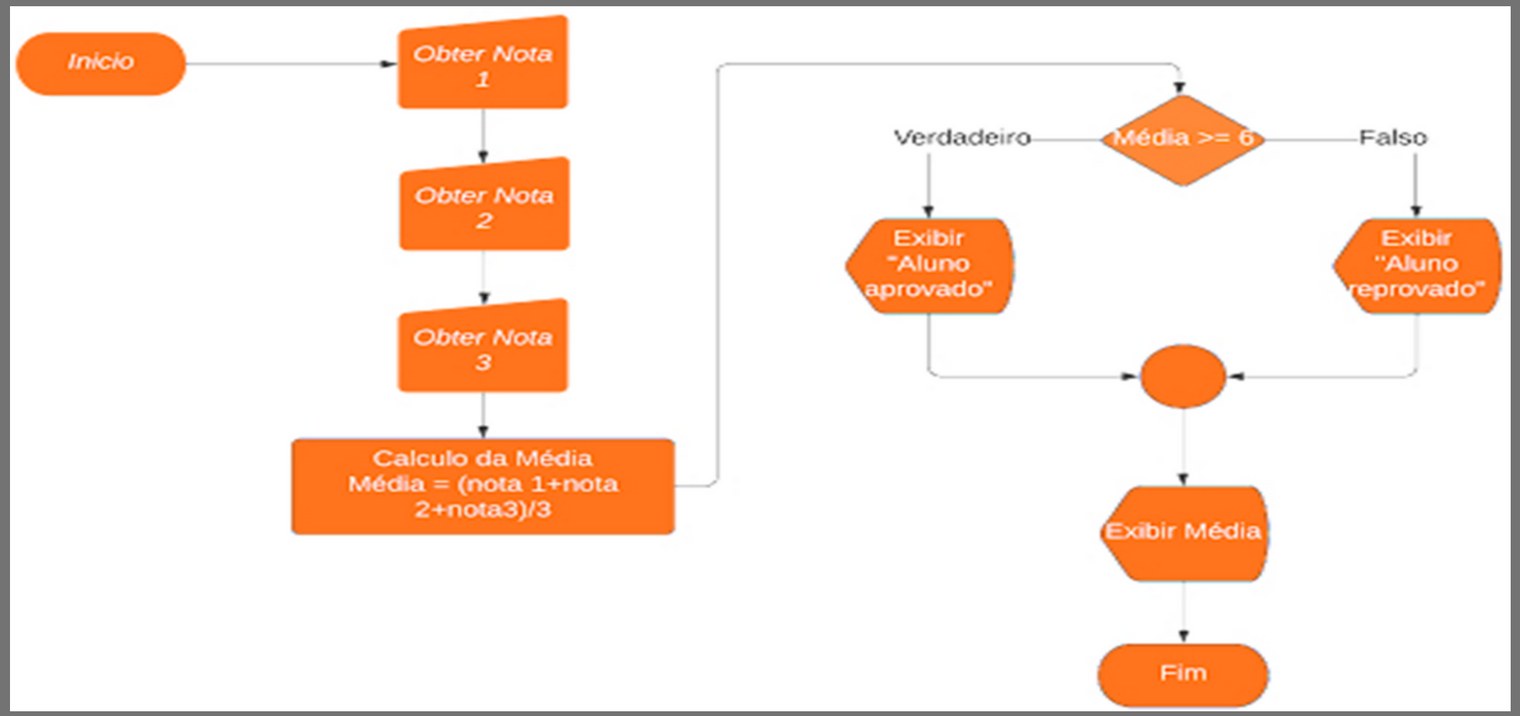
Algoritmos e Lógica de Programação

Fluxograma

Note que, na forma de um texto narrativo, ou mesmo nas outras formas de representação, é desejado que o algoritmo seja objetivo e preciso na descrição de cada passo. Existe também a opção de usar um modelo visual, como um fluxograma, veja na imagem:

SENAI

Algoritmos e Lógica de Programação



SENAI

Algoritmos e Lógica de Programação

Pseudo-linguagem

Outra forma bem interessante de representar algoritmos é utilizando uma pseudo-linguagem, que escrevemos em português (também chamado de português estruturado) sem ambiguidades e sem toda a rigidez de uma linguagem de programação.

Importante notar que um pseudo-código é escrito com frases que correspondem a estruturas utilizadas em uma linguagem de programação, destacadas em negrito no exemplo;

SENAI

Algoritmos e Lógica de Programação

```
Algoritmo CalcularMediaAlunos
 LER Nota1
   LER Nota2
   LER Nota3
   media = (Nota1+Nota2+Nota3)/3
   SE MEDIA >= 6
 ENTÃO
       IMPRIME “Aprovado”
   SENÃO
       IMPRIME “Reprovado”
 FIM-SE
 IMPRIME o valor da média
Fim-Algoritmo CalcularMediaAlunos
```

SENAI

Algoritmos e Lógica de Programação

O que é lógica de programação?

Na lógica de programação é onde aplicamos todos os conceitos de algoritmos, a definição do passo a passo e transferimos toda a lógica do algoritmo desenvolvido para uma linguagem de programação.

Em linhas gerais, lógica de programação é todo conjunto de regras e conceitos que precisamos aplicar para criar códigos que serão interpretados e executados por um computador.

SENAI

Algoritmos e Lógica de Programação

O que é preciso?

Para conseguirmos o objetivo de instruir o computador a fazer alguma coisa, precisamos de uma linguagem de programação, que é um meio estruturado para passar instruções para a máquina.

Para fazer isso, temos à disposição uma gama muito grande de linguagens como: C, Java, PHP, Python, JavaScript, entre outras.

Cada uma delas tem suas características, particularidades e cenários de aplicação.

SENAI

Algoritmos e Lógica de Programação

Exemplos de códigos e algoritmos

Vamos agora a um exemplo da codificação do algoritmo de médias usando uma linguagem de programação.

A escolhida para a representação foi a Kotlin.

SENAI

Algoritmos e Lógica de Programação

```
funmain(){
   val num1 = 8
   val num2 = 6
   val num3 = 10
   val media = (num1 + num2 + num3)/3
   println("Para ser APROVADO, a média deverá ser >= 6")
   println(media)
   if(media >=6){
      println("APROVADO")}
      else{
         println("REPROVADO")}
}
```

SENAI

Algoritmos e Lógica de Programação

Na codificação em Kotlin, temos a implementação do algoritmo desenvolvido anteriormente, mas agora, após escolher uma linguagem de programação, devemos nos atentar às suas próprias regras de sintaxe (estrutura para escrita) e semântica (significado dado aos símbolos e comandos).

É importante ressaltar que a linguagem de programação escolhida é somente mais uma ferramenta que você, enquanto pessoa desenvolvedora, precisará conhecer. A ideia do algoritmo não é ligada a nenhuma linguagem em específico.

SENAI

Algoritmos e Lógica de Programação

Note que o algoritmo é o mesmo, o passo a passo está ali, contudo, escrito em uma linguagem diferente. Portanto, após definida a sequência de instruções, escolha a linguagem que você mais se identifica para implementar o algoritmo.

A lógica de programação utilizada para desenvolver uma solução, como um sistema web, desktop ou mobile, é toda estruturada a partir da definição dos algoritmos, por isso a importância e o relacionamento entre estes dois temas.

SENAI

Algoritmos e Lógica de Programação

Conclusão

Estudar algoritmos e lógica de programação é o passo inicial para quem deseja entrar no mundo de desenvolvimento de software e começar a criar suas primeiras aplicações.

Com esses conceitos bem consolidados você passa a ter bem mais preparo na utilização de uma linguagem de programação.

SENAI

Tipos de dados

Long: inteiro de 64 bits.

Int: inteiro de 32 bits.
É um tipo que representa um número inteiro, um dos muitos tipos numéricos que podem ser representados em Kotlin, você também pode usar Byte , Short , Long , Float e Double , dependendo dos seus dados numéricos.

SENAI

Tipos de dados

Short: inteiro de 16 bits.

Byte: inteiro de 8 bits.

Double: ponto flutuante de 64 bits (igual a 8 bytes).
Valores numéricos com casas após a vírgula (positivos ou negativos).

Float: ponto flutuante de 32 bits (igual a 4 bytes).
Valores numéricos com casas após a vírgula (positivos ou negativos).

SENAI

Tipos de dados

Boolean: Seu valor pode ser true ou false. As operações de disjunção(||) e conjunção(&&) podem ser executadas em tipos booleanos.

String: são usadas para armazenar texto, contém uma coleção de caracteres entre aspas duplas.

Char: É um tipo de variável que aceita a inserção de um caractere apenas.

SENAI

Variáveis e Constantes

Declaração de variável

O Kotlin usa duas palavras-chave diferentes para declarar variáveis: val e var.

Use val para uma variável cujo valor nunca muda. Não é possível reatribuir um valor a uma variável que tenha sido declarada usando val.

Use var para uma variável cujo valor possa ser mudado.

SENAI

Variáveis e Constantes

Declaração de variável

Val: Requer sua inicialização logo que é declarada.

```
val nome = "João"
val sobrenome : String = "Paulo“
```

Var: Não precisa ser inicializada no momento da sua declaração.

```
var nome : String
nome = "João"
var sobrenome : String = "Paulo"
```

SENAI

Variáveis e Constantes

Constantes

É um valor que não pode ser mudado.

use val para declarar uma constante.
use var para declarar uma variável.

Você pode especificar um tipo como Stringapós o nome da variável. No exemplo abaixo, foi declarada uma constante primeironome do tipo String com a palavra-chave val.

```
val primeironome: String = "João"
```

SENAI

Variáveis e Constantes

$ e ${}

Podemos inserir variáveis e expressões em uma Sting

```
funmain(){
   var valor1 = 20
   var valor2 = 10
   println(valor1)
   println("valor1")
   println("valor1 é $valor1")
   println("valor1 $valor1 + valor2 $valor2 = ${valor1 + valor2}")
}
```

SENAI

Conversão entre tipos

Para converter um número de um tipo para outro, você precisa chamar explicitamente a função de conversão correspondente.

Cada tipo de número tem funções auxiliares que convertem de um tipo de número para outro:

toByte(), toInt(), toLong(), toFloat(), toDouble(), toChar(), toShort().

SENAI

Conversão entre tipos

Conversão entre tipos inteiros para ponto flutuante

```
val vint = 987
val vfloat = vint.toFloat()
```

Conversão entre tipos texto para tipos numéricos inteiros

```
val vtexto = "987"
val vint = vtexto.toInt()
```

SENAI

Conversão entre tipos

Conversão entre tipos texto para tipos ponto flutuante

```
val vtexto = "987"
val vfloat = vtexto.toFloat()
```

Conversão entre tipos inteiros para tipos texto

```
val vint = 987
val vtexto = vint.toString()
```

SENAI

Operadores

Operadores aritméticos

+ Adição
- Subtração
*** Multiplicação**
/ Divisão
% Módulo – Resto da Divisão
++ Incremento (+1)
-- Decremento (-1)

SENAI

Operadores

```
fun main(){
  var numero1 = 20
  var numero2 = 10
  println(numero1 + numero2)
  println(numero1 - numero2)
  println(numero1 * numero2)
  println(numero1 / numero2)
  println(numero1 % numero2)
  println(++numero1)
  println(--numero2)
}
```

SENAI

Operadores

```
fun main(){
  var numero1 = 20
  var numero2 = 10
  println("valor $numero1 + $numero2 = ${numero1 + numero2}")
  println("valor $numero1 - $numero2 = ${numero1 - numero2}")
  println("valor $numero1 * $numero2 = ${numero1 * numero2}")
  println("valor $numero1 / $numero2 = ${numero1 / numero2}")
  println("valor $numero1 % $numero2 = ${numero1 % numero2}")
  println("valor Incremental de $numero1 é ${++numero1}")
  println("valor Decremental de $numero2 é ${--numero2}")
}
```

SENAI

Operadores

Operadores relacionais

> Maior que
< Menor que
> = Melhor que ou igual a
<= menos que ou igual a
== é igual a
! = diferente

SENAI

Operadores

```
fun main(){
   var valor1 = 20
   var valor2 = 10
   println(valor1 > valor2)
   println(valor1 < valor2)
   println(valor1 >= valor2)
   println(valor1 <= valor2)
   println(valor1 == valor2)
   println(valor1 != valor2)
}
```

SENAI

Operadores

```
funmain(){
   var valor1 = 20
   var valor2 = 10
   println("valor $valor1 > $valor2 = ${valor1 > valor2}")
   println("valor $valor1 < $valor2 = ${valor1 < valor2}")
   println("valor $valor1 >= $valor2 = ${valor1 >= valor2}")
   println("valor $valor1 <= $valor2 = ${valor1 <= valor2}")
   println("valor $valor1 == $valor2 = ${valor1 == valor2}")
   println("valor $valor1 != $valor2 = ${valor1 != valor2}")
}
```

SENAI

Operadores

Operadores lógicos

&&(e) retorna verdadeiro se todas as expressões são verdadeiras

```
funmain(){
   if (10==10 && 10<20){
       println("Verdadeiro")}
   else{
       println("Falso")}
}
```

SENAI

Operadores

Operadores lógicos

||(ou) retorna verdadeiro se alguma expressão for verdadeira

```
funmain(){
   if (10==10 || 20<20){
       println("Verdadeiro")}
   else{
       println("Falso")}
}
```

SENAI

Estruturas de Decisão

Condição Simples

A Estrutura Condicional Simples executa um comando ou vários comandos se a condição for verdadeira. Se a condição for falsa, a estrutura é finalizada sem executar os comandos. O comando que define a estrutura é representado pela palavra if.

SENAI

Estruturas de Decisão

Condição Simples

```
funmain(){

   val idade = 18

   println(idade)

   if (idade >= 18){

      println("Maior de Idade")}
}
```

SENAI

Estruturas de Decisão

Condição Composta

A Estrutura Condicional Composta segue o mesmo princípio da Estrutura Condicional Simples, com a diferença de que quando a condição não é satisfeita, será executado o outro comando. O comando que define a estrutura é representado pelas palavras if e else.

SENAI

Estruturas de Decisão

Condição Composta

```
funmain(){
   val idade = 18
   println(idade)
   if (idade >= 18){
      println("Maior de Idade")}
   else{
      println("Menor de Idade")}
```

SENAI

Estruturas de Decisão

Condição Composta

```
funmain(){
   val num1 = 8
   val num2 = 6
   val num3 = 10
   val media = (num1 + num2 + num3)/3
   println("Para ser APROVADO, a média deverá ser > 7")
   println(media)
   if(media > 7){
      println("APROVADO")}
      else{
      println("REPROVADO")}
}
```

SENAI

Estruturas de Decisão

Condição Aninhadas

Uma estrutura de condição é aninhada dentro de outra, como bloco verdadeiro ou falso.

Neste caso, para que a estrutura de condição mais interna seja avaliada, é necessário que uma determinada condição seja satisfeita na estrutura de condição.

SENAI

Estruturas de Decisão

Condição Encadeadas

```
funmain() {
   val idade = 65
   println(idade)
   if (idade >= 18 && idade < 65) {
      println("Maior de Idade")}
   else {
      if (idade >= 65) {
         println("Terceira Idade")}
      if (idade < 18) {
         println("Menor de Idade")}}}
```

SENAI

Estruturas de Decisão

```
funmain() {
   val num1 = 10
   val num2 = 10
   val num3 = 10
   val media = (num1 + num2 + num3)/3
   println("Para ser APROVADO, a média deverá ser >= 7")
   println(media)
   if(media >= 7){
      println("APROVADO")}
      else{
      if(media <=2){
         println("REPROVADO")}
      else{
         println("RECUPERAÇÃO")}}}
```

SENAI

Estruturas de Repetição

for

O laço for é uma estrutura de repetição muito utilizada nos programas. É muito útil quando se sabe quantas vezes a repetição deverá ser executada. Este laço utiliza uma variável para controlar a contagem do loop, bem como seu incremento.

SENAI

Estruturas de Repetição

```
funmain(){
   for (i in 1..10){
    print("$i ")}
    println()

   for (i in 0..20 step 2){
    print("$i ")}
    println()

   for (j in 20 downTo 0 step 2) {
       print("$j ")}}
```

SENAI

Estruturas de Repetição

```
funmain(){
    print("qual o seu nome: ")
    val text = readLine()
    for (i in 1..5)
        println(text)
}
```

SENAI

Estruturas de Repetição

while

Esta instrução é usada quando não sabemos quantas vezes um determinado bloco de instruções precisa ser repetido. Com ele, a execução das instruções vai continuar até que uma condição seja verdadeira. A condição a ser analisada para a execução do laço de repetição deverá retornar um valor booleano.

SENAI

Estruturas de Repetição

```
fun main(){
    var i = 0
    while (i < 20){
        print("$i ")
        i++}

    println()
    var n = 20
    while (n > 0){
        print("$n ")
        n--}}
```

SENAI

Estruturas de Repetição

do while

Tem quase o mesmo funcionamento que o while, a diferença é que com o uso dele teremos os comandos executados ao menos uma única vez.

SENAI

Estruturas de Repetição

```
fun main(){
var i = 0
do {
  print("$i ")
  i++
 }while (i < 10)

do {
        print("qual o seu nome: ")
        val value = readLine()
   } while (value == "")}
```

SENAI

Estruturas de Repetição

A relação de consumo entre álcool e gasolina é de 70%, ou seja, se temos o mesmo volume de combustível, a energia gerada por 70% do consumo de gasolina é o mesmo de 100% do consumo de álcool.

Tendo esta informação, conseguimos saber, na hora de abastecer, qual combustível é o mais vantajoso, dividindo o preço do álcool pelo preço da gasolina.

- **Se este valor for superior a 0,7, vale a pena a gasolina.**
- **Se este valor for inferior a 0,7, vale a pena o álcool.**
- **Se o valor for exatamente 0,7, tanto faz abastecer com Gasolina ou Etanol.**

SENAI

Estruturas de Repetição

break

A quebra rotulada é usada para sair para o bloco desejado quando satisfaz uma condição específica sem verificar a condição no loop while. Em seguida, transfere o controle para a instrução seguinte do bloco while.

SENAI

Estruturas de Repetição

```
funmain() {
    var i = 0
    while (i < 1000) {
        if (i == 50) {
            break
        }
        print("$i ")
        i++
    }}
```

SENAI

Funções

É um tipo de subalgoritmo que descreve uma sequência de ordens/instruções, as quais devem executar uma tarefa específica de uma aplicação maior.

Você pode agrupar uma ou mais expressões em uma função. Em vez de repetir a mesma série de expressões sempre que precisar de um resultado, você pode unir as expressões em uma função e chamar essa função.

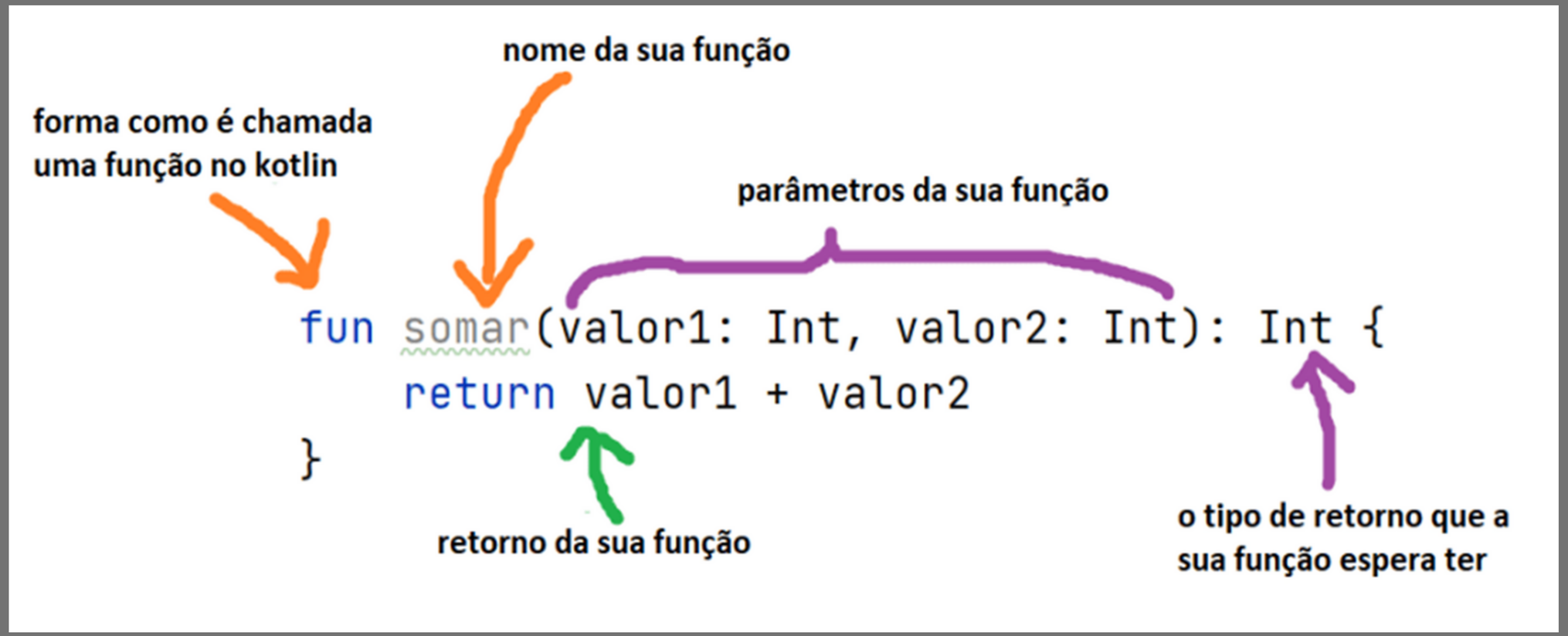
SENAI

Funções

Para declarar uma função, use a palavra-chave fun seguida pelo nome da função. Em seguida, defina os tipos de entrada que sua função assume, se houver, e declare o tipo de saída retornada. No corpo de uma função, você define expressões que são chamadas quando sua função é invocada.

SENAI

Funções



SENAI

Funções

```
fun main() {
 val x = somar(20, 30)
 println(x)
}

fun somar(valor1: Int, valor2: Int): Int {
 return valor1 + valor2
}
```

SENAI

Funções

```
fun main() {
mostrar()
}
fun mostrar(){
val num1 = 8
val num2 = 6
val num3 = 10
val media = (num1 + num2 + num3)/3
println("Para ser APROVADO, a média deverá ser > 7")
println("Média: $media")
if(media > 7){
println("APROVADO")}
else{
println("REPROVADO")}
}
```

SENAI

Funções

```
fun main(){
   val text = "João Paulo"
   println(text)
   println(text.plus(" Paulo")) // Adiciona um texto no final de outro texto
   println(text.get(0)) // Pega a posição do Texto
   println(text.reversed()) // Escreve de trás para frente
   println(text.substring(11,16)) //Pegar um parte do texto com o intervalo
   println(text.length) //Tamanho do Texto
   println(text.first()) //Primeira letra do Texto
   println(text.last()) //Última letra do Texto
}
```

SENAI

Fim

SENAI