|  |
| --- |
| 1. Mude o nome do ficheiro, indicando o seu nome e apelido da seguinte forma: ***Lab02\_NomeApelido.docx*** 2. Resolva os exercicios neste documento word, usando a fonte embebida. 3. Siga o exemplo de sintaxe do primeiro exercicio, resolvido a título de exemplo. 4. Indente corretamente o pseudocódigo. 5. Inicie sempre com a palavra **início** e termine com a palavra **fim**. 6. Declare o tipo e as variáveis que usará no início do código. 7. Pode utilizar as seguintes operadores:    * Aritméticos: **+**, **-**, **\***, **/**, **^, %**    * Atribuição: ****    * Comparação: **==**, **!=**, **>**, **<**, **>=**, **<=**    * lógicos: **e**, **ou** 8. Pode utilizar as seguintes instruções:    * **se senao fimse** 9. Pode utilizar as seguintes funções:    * **ler** (função que retorna o valor lido do teclado)    * **escrever** (função que escreve no ecrã o seu argumento) 10. Consulte se necessário os exemplos disponíveis nos slides. 11. No final deverá implementar em Scratch e conferir que funciona como esperado. Insira o código como imagem:     * Clique na tecla *Prnt Scrn*     * Faça paste para a janela amarela do codigo     * Selecione a imagem, e no menu formato, use o comando crop/recortar para recortar a imagem adequadamente. Veja o exemplo do 1º exercício. |

1. *(a título exemplificativo, este exercício encontra-se resolvido.)* Elabore um algoritmo que tem armazenado uma password numérica. Peça ao utilizador para inserir um número. O programa deverá ler o número e verificar se é igual à password. Caso o seja, deverá imprimir a mensagem, “acertou na password”. Caso contrário, imprima “não acertou na password”. Implemente em Scratch e confira que funciona corretamente. Grave com o nome *Lab02\_NomeApelido\_ex1.sb3*

**Resposta em Pseudocódigo:**

**inicio**

**inteiro:** password,tentativa

password  1234

**escrever**(“Indique a password:”)

tentativa  **ler**()

**se** tentativa == password **entao**

**escrever**(“Acertou na password”)

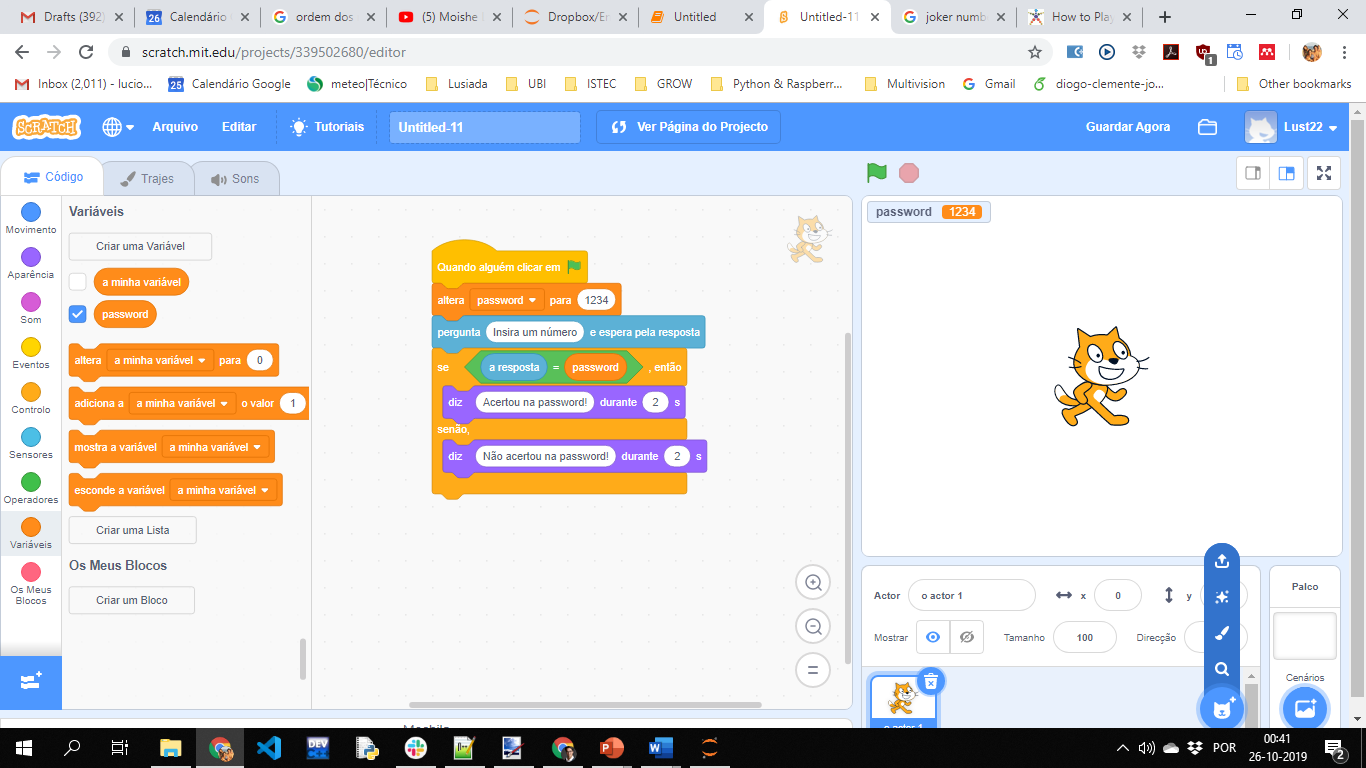
**senao**

**escrever**(“Não acertou na password”)

**fimse**

**fim**

**Resposta em Scratch**



Nota: selecione “diz ... durante”. Senão executa a instrução (diz) e passa logo para instrução seguinte, e a resposta desaparece logo.

1. Elabore um algoritmo que tem armazenado uma password numérica. Peça ao utilizador para inserir um número. O programa deverá ler o número e verificar se é igual à password. Caso o seja, deverá imprimir a mensagem, “acertou na password”. Caso o número seja inferior à password, diga “demasiado pequeno”. Caso o número seja superior à password, diga “demasiado grande”.

**Resposta em Pseudocódigo:**

inicio

inteiro: password

inteiro: passwordInput

escrever(“Gess the password (4 numbers): ”)

passwordInput  **ler**()

// Não existe “eles if” em pseudocodigo?

se passwordInput == password entao

escrever(“Acertou na password”)

fimse

se passwordInput < password entao

escrever(“Demasiado pequeno.”)

fimse

se passwordInput > password entao

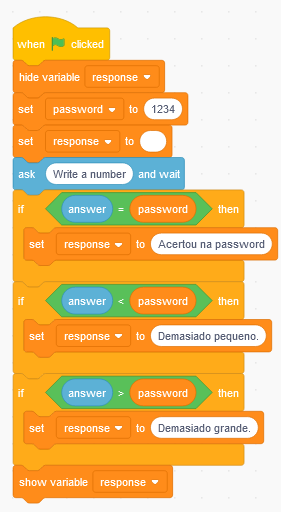
escrever(“Demasiado grande.”)

fimse

fim

Implemente em Scratch e confira que funciona corretamente. Grave com o nome *Lab02\_NomeApelido\_ex2.sb3* ecoloque uma imagem dos módulos que compôs.

**Resposta em Scratch:**



1. Escreva um algoritmo em pseudocódigo que solicite um número inteiro ao utilizador. Em relação ao número inserido, deve avaliar:

* se é par.
* se está entre 0 e 100 (inclusivé).
* se é múltiplo de 5 mas não é múltiplo de 2.
* Se está entre -20 e -10 (inclusivé) ou entre 10 e 20 (inclusivé) e é um número par
* Se é um múltiplo de 7, não é negativo e tem 3 dígitos (uma forma de ver se tem 3 dígitos é se a divisão por 100 dá entre 1 e 9)

Para cada um dos requisitos, deve construir duas condições diferentes mas equivalentes (recorrendo entre outros às regras de Morgan), e imprimir sempre o resultado das duas. Por exemplo, se se perguntar se o número é positivo, deve avaliar de duas formas, escrevendo sempre duas vezes, se for correto:

**se** n > 0 **então**

**escrever(**"O número inserido é positivo"**)**

**se** !(n <= 0) **então**

**escrever(**"O número inserido é positivo"**)**

**Resposta:**

inicio

escrever(“Type a number: ”)

inteiro: num

num <- ler()

se num >= 0 e num <= 100 então

escrever(“O número está entre 0 e 100 (inclusivé)”)

fimse

se !(num < 0) e !(num > 100) então

escrever(“O número está entre 0 e 100 (inclusivé)”)

fimse

se num % 2 == 0 então

escrever(“O número é par”)

fimse

se !(num % 2 != 0) então

escrever(“O número é par”)

fimse

se num % 5 == 0 e num % 2 != 0 então

escrever(“O número é múltiplo de 5 mas não é múltiplo de 2”)

fimse

se !(num % 5 != 0) e !(num % 2 == 0) então

escrever(“O número é múltiplo de 5 mas não é múltiplo de 2”)

fimse

se (num >= -20 e num <= -10 ou num >= 10 e num <= 20) e

num % 2 == 0

então

escrever(“O número está entre -20 e -10 (inclusivé) ou entre 10 e 20 (inclusivé) e é um número par”)

fimse

se (!(num < -20) e !(num > -10) ou !(num < 10) e !(num > 20)) e

!(num % 2 != 0)

então

escrever(“O número está entre -20 e -10 (inclusivé) ou entre 10 e 20 (inclusivé) e é um número par”)

fimse

inteiro: dividedBy100 <- num / 100;

se num % 7 == 0 e num >= 0 e dividedBy100 >= 1 e dividedBy100 <= 9

então

escrever(“O número é um múltiplo de 7, não é negativo e tem 3 dígitos”)

fimse

se !(num % 7 != 0) e !(num < 0) e !(dividedBy100 < 1) e

!(dividedBy100 > 9)

então

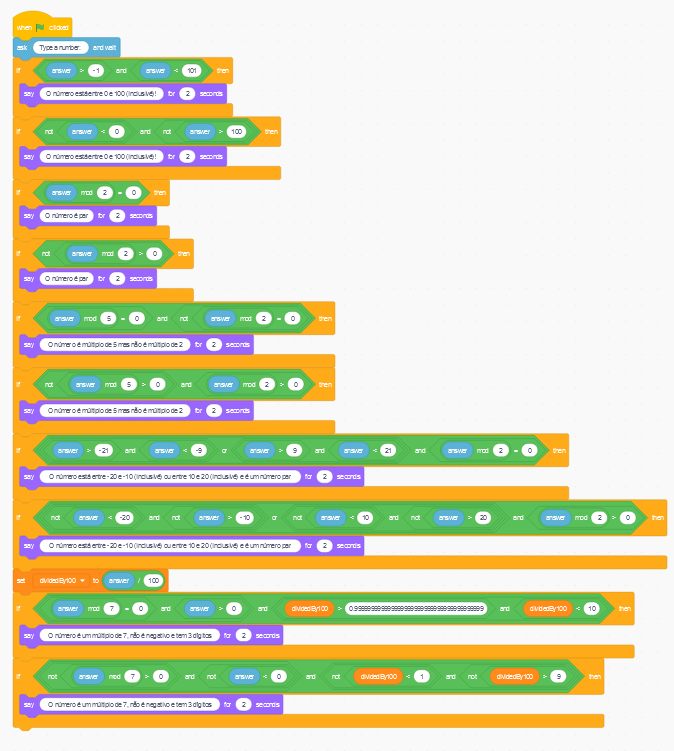
escrever(“O número é um múltiplo de 7, não é negativo e tem 3 dígitos”)

fimse

fim

Implemente em Scratch e confira que funciona corretamente. Note que a negação em Scratch corresponde ao operador “é falso que”. Grave com o nome *Lab02\_NomeApelido\_ex3.sb3*

**Resposta em Scratch:**



1. Escreva um algoritmo em pseudocódigo que simula uma calculadora.
   1. Deverá solicitar ao utilizador para inserir dois números inteiros.
   2. Depois deverá perguntar se o utilizador quer fazer uma soma, subtração, multiplicação ou divisão destes dois números (associe um número a cada uma das operações: “1. Soma, 2. Subtração, ...). A opção deverá ser armazenada numa variável.
   3. Com seletores, identifique a opção selecionada e faça a operação escolhida e apresente o resultado no ecrã.
   4. Deverá salvaguardar a possibilidade de se pretender fazer uma divisão e o divisor ser 0; nesse caso não poderá fazer a divisão e deverá imprimir um alerta.
   5. Caso a opção escolhida não seja válida, deverá imprimir uma mensagem de erro.

**Resposta:**

inicio

escrever(“Type an integer number: ”)

inteiro: num1

num1 <- ler()

escrever(“Type another integer number: ”)

inteiro: num2

num2 <- ler()

escrever(“Select the calculation type: ”)

escrever(“1. Addition, 2. Subtraction, 3. Multiplication, 4. Division”)

int calcType

calcType <- ler()

switch calcType

caso 1:

escrever(num1 + num2)

abacar

caso 2:

escrever(num1 – num2)

abacar

caso 3:

escrever(num1 \* num2)

abacar

caso 4:

se num2 == 0 então

escrever(“You can not divide by 0.\nPlease, try again.”)

inicio

fimse

escrever(num1 / num2)

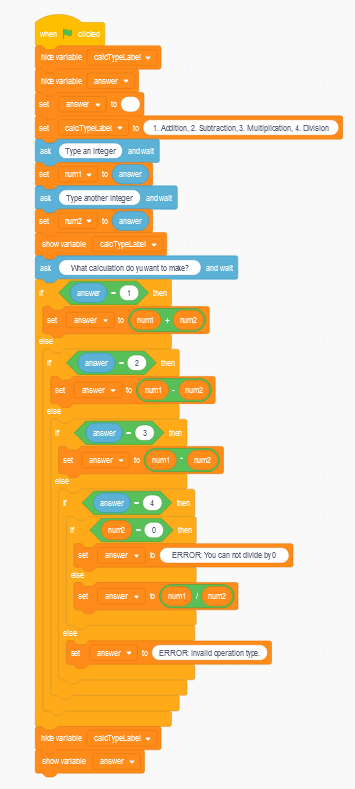
abacar

fimswitch

fim

Implemente em Scratch e confira que funciona corretamente. Grave com o nome *Lab02\_NomeApelido\_ex4.sb3*

**Resposta em Scratch:**



1. Escreva um algoritmo em pseudocódigo que pede ao utilizador para inserir um número positivo. Enquanto o número não for positivo, volte a pedir.

**Resposta:**

inicio

inteiro: input

executa

escrever(“Give me a positive number: ”)

input <- ler()

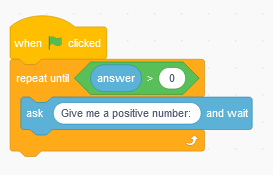
equanto ( input <= 0 )

escrever(“Thank you!”)

fim

Implemente em Scratch e confira que funciona corretamente. Grave com o nome *Lab02\_NomeApelido\_ex5.sb3*

**Resposta em Scratch:**



1. Escreva um algoritmo em pseudocódigo que pede ao utilizador para inserir um número positivo. Enquanto o número não for positivo, volte a pedir. Imprima a soma de todos os inteiros de 1 até o número escolhido. Por exemplo, se inserir o número 4, deverá apresentar como resultado 10 (que é 1+2+3+4 = 10).

**Resposta:**

inicio

inteiro: input

executa

escrever(“Give me a positive number: ”)

input <- ler()

equanto ( input <= 0 )

escrever("These are all the positive integer numbers from 1 to your number:")

inteiro: i

inteiro: result <- 0

para i <- 1 enquanto i <= input passo ++i executa

i +<- result

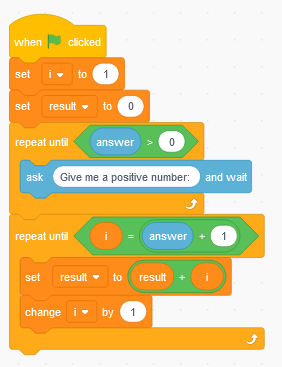
fimpara

escrever(result)

fim

Implemente em Scratch e confira que funciona corretamente. Grave com o nome *Lab02\_NomeApelido\_ex6.sb3*

**Resposta em Scratch:**



ISTEC, 2019-2010

Licenciatura de Informática, 1º Ano

João Neves