











Aluno: Willian Pacheco Silva
Matrícula: 2019107753
Disciplina: Programação II

Respostas:

1 -



Como um algoritmo é uma sequência lógica e finita de instruções que devem ser seguidas para a resolução de um problema ou execução de uma tarefa, a principal função de um algoritmo é organizar e apresentar uma lista de passos a serem executados para solucionar um problema, ou executar alguma tarefa.

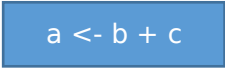

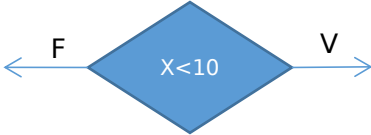
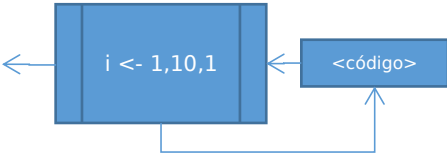
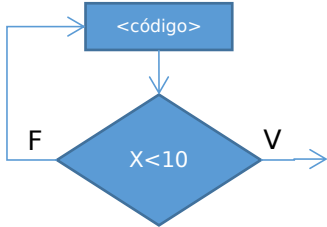
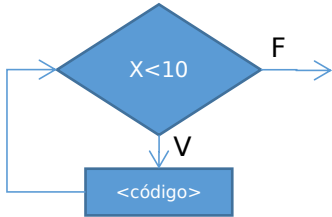
2 -

	Indica o fluxo de dados ou conecta os outros símbolos
	Início ou fim de processamento
	Processamento em geral (cálculos)
	Entrada ou saída de dados (leitura ou gravação de arquivos)
	Desvio para um ponto qualquer do programa
	Entrada manual de dados via teclado
	Processos de decisão (condicionais)
	Conector de páginas (quando o documento tem mais de uma página)
	Exibir informações na tela
	Processo pré definido

3 -

Uma semelhança entre a resolução de problemas com o pseudocódigo e com fluxograma é que ambos tem por objetivo mostrar de antemão a lógica usada para resolver o problema. Para isso, ambos desfrutam de operadores aritméticos, de estruturas condicionais e de estruturas de repetição. Outra similaridade é que ambos tratam de um programa que tem um início, meio e fim. Tanto o fluxograma, quanto o pseudocódigo tem que ter um começo e terminar no fim do diagrama ou do código, independente de que rumo o programa tomar, ou de quais estruturas condicionais ele passar, deve-se sair do início e chegar o fim do programa. Segue uma tabela que relaciona algumas estruturas usadas em pseudocódigo aos respectivos diagramas que as representam:

Pseudocódigo	Fluxograma
Mostrar("Hello World")	
Ler(variavel)	

$a \leftarrow b + c$	
Início/fim	
Se ($x < 10$) Então Fim-se	
Para $i \leftarrow 1$ Até 10 Passo 1 Faça Fim-para	
Repita Até que ($x < 10$)	
Enquanto ($x < 10$) Faça Fim-enquanto	

4 -

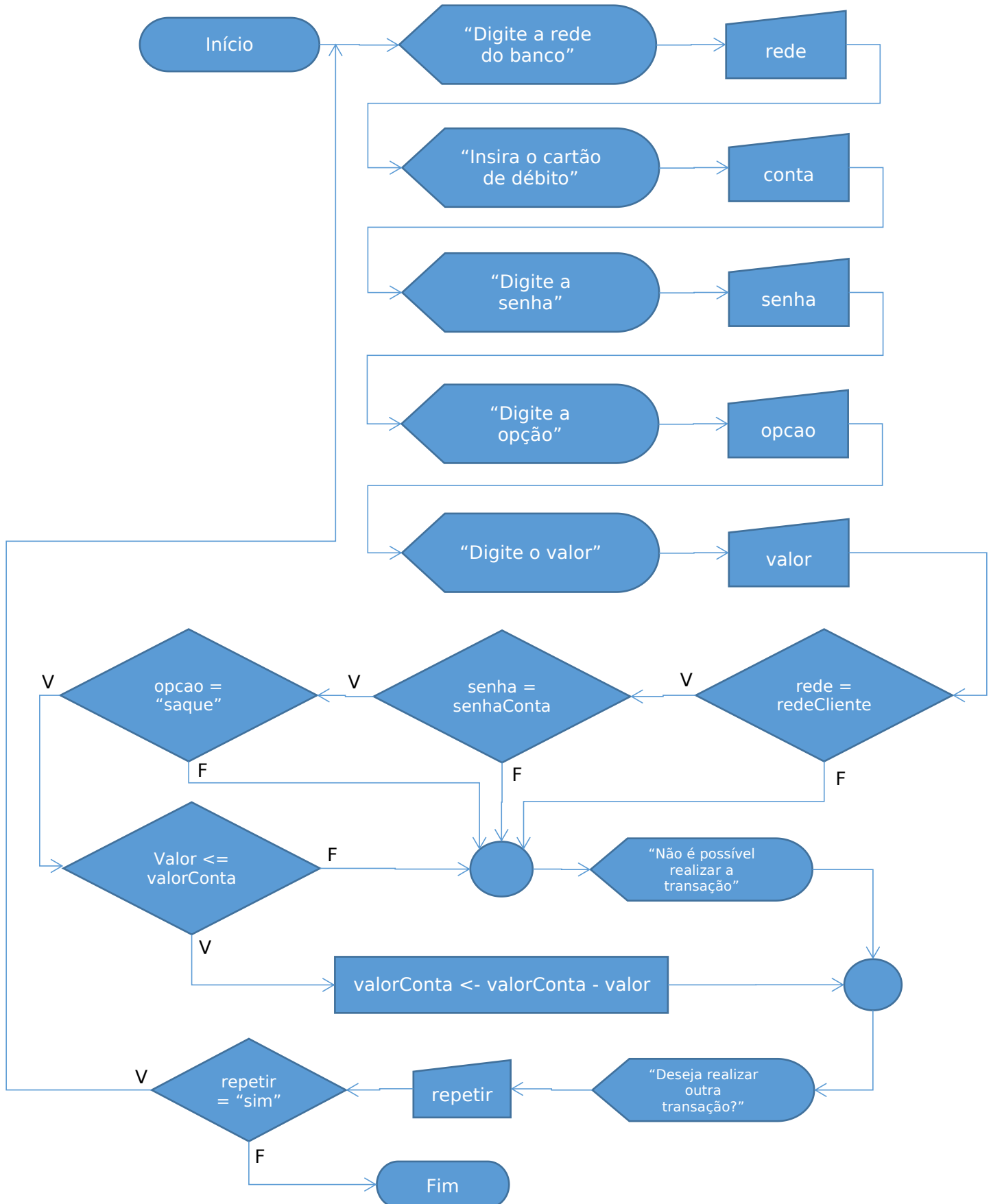
1. Ligar o computador se não estiver ligado
2. Inserir o pendrive
3. Abrir o gerenciador de arquivos
4. Localizar o pendrive no gerenciador de arquivos
5. Localizar o arquivo que deseja abrir
6. Selecionar um editor/visualizador de texto para abrir o arquivo
7. Abrir o arquivo

5 -

1. Procurar o caixa eletrônico mais próximo
2. Verificar se o caixa eletrônico pertence a rede do seu banco
3. Inserir o cartão de débito
4. Digitar a senha
5. Selecionar a opção de saque
6. Verificar se o saldo da conta é maior que o valor que deseja sacar
7. Digitar o valor que deseja sacar (100 reais)
8. Espere a máquina processar a transação
9. Confirmar a transação

10. Pegar o dinheiro
11. Decidir se deseja realizar outra transação
12. Retirar o cartão
13. Guardar o dinheiro na carteira
14. Guardar a carteira no bolso
15. Voltar para casa

6 - (Fiz pensando como se fosse implementar o sistema do caixa eletrônico)



7 - (Fiz o código supondo que seja apenas a parte interna do caixa que trata de depósitos e saques, desconsiderando a parte de inserir o cartão e outras verificações)

Alg_7

Variáveis

op: inteiro

valorConta, valor:real

continuar:literal

Início

Repita

Mostrar("1 - Depósito")

Mostrar("2 - Saque")

Mostrar("3 - Mostrar saldo")

Mostrar("Digite a opção: ")

Ler(op)

Escolha(op)

Caso 1:

Mostrar("Digite o valor: ")

Ler(valor)

valorConta <- valorConta + valor

Caso 2:

Mostrar("Digite o valor: ")

Ler(valor)

Enquanto(valor > valorConta) faça

Mostrar("Saldo indisponível")

Mostrar("Digite o valor: ")

Ler(valor)

fim-enquanto

valorConta <- valorConta - valor

Caso 3:

Mostrar("Valor da conta: ", valorConta)

Caso contrário:

Mostrar("Opção inválida!")

Fim-escolha

Mostrar("Deseja continuar?")

Ler(continuar)

Até que (continuar = "nao")

Fim

8 -

1. Ligar o computador
2. Conectar a Internet
3. Abrir o navegador
4. Procurar pelo site de vendas na URL
5. Localizar o produto
6. Selecionar quantidade
7. Adicionar ao carrinho de compras
8. Abrir carrinho de compras
9. Localizar produto
10. Confirmar compra
11. Selecionar cartão de crédito como a forma de pagamento
12. Informar os dados do cartão
13. Conferir dados do produto
14. Conferir dados do cartão
15. Conferir dados da conta
16. Finalizar compra

9 -

1. Baixo requisito de memória
2. Não precisa de nada complexo para ser compilado como os ambientes de desenvolvimento Java, apenas um bloco de notas e um compilador
3. Utiliza tipos de dados simples
4. Existem compiladores C para quase todos os computadores
5. É possível trabalhar com a linguagem em um baixo nível

10 -

A -

Constante : Preço
Variável : Quantidade

Fórmula: $\text{ValorTotal} \leftarrow \text{preco} * \text{quantidade}$

B -

Variável : comprimento e altura

Fórmula: $\text{Area} \leftarrow \text{comprimento} * \text{altura}$

C -

Constante: valor 4
Variável: a,b,c

Fórmula: $\text{Delta} \leftarrow b * b - 4 * a * c$

11 -

A -

Ordem operações: *, -, <-
Resultado: 133

B -

Ordem operações: < >, <-
Resultado: 1 (true)

C -

Ordem operações: (), /, <-
Resultado: 2.75

D -

Ordem operações: /, +, <-
Resultado: 9.5

E -

Ordem operações: /, mod, <-
Resultado: 0

F -

Ordem operações: (), /, <-
Resultado: 0

G -

Ordem operações: (), *, -, <-
Resultado: 53

H -

Ordem operações: *, +, -, <-
Resultado: 53

- I -**
 Ordem operações: +, >, .e., <-
 Resultado: .f. .e. .f. = .f. = 0
- J -**
 Ordem operações: *, +, >, <>, .ou.
 Resultado: .v. .ou. .v. = .v. = 1

12 -

As variáveis armazenam dados temporariamente. No caso de uma linguagem de programação, elas armazenam dados na memória RAM, e esses dados são utilizados durante o processamento para resolver o problema em questão. Já as constantes são valores que não sofrem alteração. Durante o desenvolvimento do programa, elas permanecem inalteradas.

13 -

Enquanto: O laço de repetição enquanto é representado pelo While na linguagem C. Essa estrutura faz o teste no início, se o teste for verdadeiro, o código dentro da estrutura de repetição é executado, caso ele seja falso, o programa não entra nesse bloco de código. Isto é, caso a condição seja falsa na primeira vez, o programa não executará este trecho de código nenhuma vez, caso seja verdadeiro, ele executa até que a condição seja falsa. Isso implica que a estrutura enquanto é executada quando a condição for verdadeira, e é deixada quando a condição for falsa, e existe a possibilidade dela não ser executada (quanto o primeiro teste for falso).

Faça enquanto: Essa estrutura é característica por fazer o teste no final da execução do trecho de código que está dentro dela. Isso quer dizer que o programa executa pelo menos uma vez o que está dentro do bloco de código, e depois testa a condição. Caso a condição seja verdadeira, ele se mantém no loop, caso ela seja falsa, o programa deixa o loop.

Repita: Essa estrutura de repetição também faz o teste no final do bloco de código, ou seja, o trecho de código é executado pelo menos uma vez antes que o teste seja realizado. Após o teste ser realizado, o programa continua dentro do loop caso a condição seja falsa, e ele sai do laço caso a condição seja verdadeira.

14 -

Alg_14_diferenca

Variáveis

h1,m1,s1,h2,m2,s2,tot: inteiro

Inicio

Mostrar("Digite os valores de hora, minuto e segundo correspondente ao primeiro tempo)

Ler(h1)

Ler(m1)

Ler(s1)

Mostrar("Digite os valores de hora, minuto e segundo correspondente ao segundo tempo)

Ler(h2)

Ler(m2)

Ler(s2)

t1 <- h1*3600 + m1*60 + s1;

t2 <- h2*3600 + m2*60 + s2;

Se t2>t1 então

```

        tot <- t2-t1
Senão
        tot <- t1-t2
Fim-se
Mostrar("Horas: ", tot/3600);
Mostrar("Minutos: ", (tot%3600)/60);
Mostrar("Segundos: ", (tot%3600)%60);

```

Alg_14_soma
Variáveis

```

Início
    Mostrar("Digite os valores de hora, minuto e segundo correspondente ao
primeiro tempo)
    Ler(h1)
    Ler(m1)
    Ler(s1)
    Mostrar("Digite os valores de hora, minuto e segundo correspondente ao
segundo tempo)
    Ler(h2)
    Ler(m2)
    Ler(s2)

     $t1 = h1*3600 + m1*60 + s1$ 
     $t2 = h2*3600 + m2*60 + s2$ 

    tfinal <- t1+t2

    Mostrar("Horas: ", tot/3600);
    Mostrar("Minutos: ", (tot%3600)/60);
    Mostrar("Segundos: ", (tot%3600)%60);

```

17 -

```

Alg_17
Variáveis
    num: inteiro
    somaPar, prodImpar: real
    somaPar<- 0
    prodImpar <-1
Início
    Repita
        Mostrar("Digite o número: ")
        Ler(num)
        Se (num>0) então
            Se ((num mod 2)=0) então
                somaPar = somaPar + num
            Senão
                prodImpar = prodImpar * num
        Fim-se
    Fim-se
    Até que (num<=0)
    Mostrar("Soma dos pares: ", somaPar)
    Mostrar("Produto dos ímpares: ", prodImpar)
Fim

```

18 -

Alg_18

Variáveis

a[100]: inteiro //tem que usar vetor e não tem como definir o número de casas

depois de ler qtd

i,j,k: inteiro

fat:real

Início

i<-0

Mostrar("Digite a quantidade de números: ")

Ler(quantidade)

Repita

Mostrar("Digite o número: ")

Ler(a[i])

Para j<-0 até i passo 1 faça

fat<-1

Para k<- 1 até a[j] passo 1 faça

fat<-fat *k

Fim-para

Mostrar("Valor lido : ", a[j])

Mostrar("Fatorial do valor: ", fat)

Fim-para

i<-i+1

Até que (i>=quantidade)

Fim

19 -

PSEUDOCÓDIGO	LINGUAGEM C
vogais[5]:literal	char vogais[5][1];
altura[10]:real	float altura[5];
mesesAno[12]:literal	char vogais[12][20];

29 -

Alg_29

Variáveis

i, v1[10], v2[10], maior: inteiro

Início

Para i <- 0 até 9 passo 1 faça

Mostrar("Digite o valor :")

Ler(v1[i])

Se (i=0) então

maior = v1[i]

Senão

Se (v1[i]> maior) então

Maior <- v1[i]

Fim-se

Fim-se

Fim-para

Para i <- 1 até 10 passo 1 faça

v2[i] <- v1[i]*maior

Mostrar(v2[i])

Fim-para

Fim