UNIVERSIDADE VILA VELHA
ESPÍRITO SANTO

Disciplina: Fundamentos da Computação		**Visto:**
Professor: Abrantes Araújo Silva Filho		
Aluno:		
Turma:	**Semestre:**	**Valor:** —
Data:	**Diário** 1b: Fundamentos da Computação	

Unidade 1: Fundamentos da Computação (2ª parte)
— Diário de Aprendizagem —

- Este **Diário de Aprendizagem** é uma das atividades integrantes da disciplina de **Fundamentos da Computação** do curso de Ciência da Computação, Universidade Vila Velha (UVV).
- A confecção do diário de aprendizagem é atividade **obrigatória** e **altamente recomentada** por três motivos: a) você aprenderá muito mais a matéria se mantiver o diário; b) ao entregar todos os diários ao professor você está cumprindo parte das atividades avaliativas que contam pontos na disciplina (10% da nota); e c) as provas bimestrais discursivas seguirão o formato e conteúdo das perguntas do diário.
- Se você tiver dificuldade em responder alguma questão do diário, estude novamente a matéria. Se você realmente entendeu a matéria, não deveria ter muita dificuldade para responder o diário.
- Responda com caneta ou lápis escuro (2B, 4B, 6B).
- Verifique no calendário de sua turma a **data de entrega**. Após uma rápida avaliação e visto pelo professor ou pelos monitores, seu diário será devolvido.
- O diário não será corrigido pelo professor: cabe a você estudar e dar a resposta correta para todas as questões. Obviamente o professor está à disposição para esclarecimento de dúvidas, e os monitores podem auxiliar caso você tenha dificuldade.
- Manter o diário de aprendizagem atualizado pode ser a diferença entre você aprender a matéria e ser aprovado, ou não aprender a matéria e não ser aprovado.
- Bons estudos!

1 Algoritmos

1. Já sabemos que a Ciência da Computação é a ciência que projeta e implementa algoritmos para a resolução de problemas. Em relação a essa definição, responda:
   (a) Explique qual o interesse do cientista da computação quando ele **projeta** algoritmos?

   (b) Explique quais os três interesses do cientista da computação quando ele **implementa** algoritmos?

   (c) Explique qual o interesse do cientista da computação quando ele **resolve problemas** através de algoritmos?

2. A área da computação pode ser estudada em diferentes cursos, tais como: Ciência da Computação, Engenharia da Computação, Sistemas de Informação, Engenharia de Software e outros. O que diferencia esses cursos entre si?

3. Cite 3 disciplinas que preparam o cientista da computação para o estudo das propriedades formais dos algoritmos.

4. Cite 3 disciplinas que preparam o cientista da computação para o estudo das propriedades matemáticas dos algoritmos.

5. Cite 3 disciplinas que preparam o cientista da computação para o estudo da realização linguística dos algoritmos.

6. Cite 3 disciplinas que preparam o cientista da computação para o estudo da realização em hardware dos algoritmos.

7. Cite 3 disciplinas que preparam o cientista da computação para o estudo da realização virtual dos algoritmos.

8. Cite 3 disciplinas que preparam o cientista da computação para a criação de aplicações dos algoritmos.

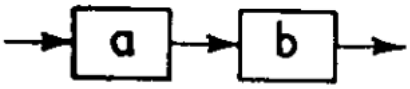
9. Por que a ciência da computação não é somente sobre programação nem somente sobre computadores?

10. Escreva uma definição **informal** para o conceito de algoritmo. Explique a definição.

11. Qualquer algoritmo, desde os mais simples até os mais complexos e sofisticados, pode ser criado utilizando-se apenas esses 4 conceitos. Cite quais são esses conceitos.
    (a) _____
    (b) _____
    (c) _____
    (d) _____

12. O que são **operações seqüenciais**?

13. Que tipo de operação está representado no fluxograma abaixo?



14. Em relação às operações seqüenciais em um algoritmo, explique o que são as **sentenças** (*statements*).

15. Em um algoritmo, cada **sentença** (*statement*) deve ser específica, única, simples, atômica, primitiva. O que isso quer dizer?

16. Por que as **sentenças** devem ser atômicas?

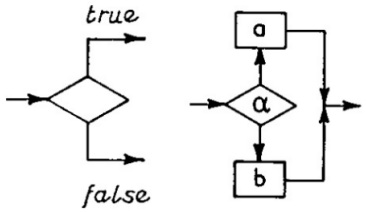
17. As **operações condicionais** têm dois sinônimos consagrados. Quais são esses sinônimos?

18. O que são as **operações condicionais**?

19. O que é uma **expressão booleana**?

20. Qual a relação existente entre as operações condicionais e as expressões booleanas?

21. A figura abaixo representa operações condicionais através de um fluxograma. O que a estrutura marcada com a letra grega α representa? Por que essa estrutura é importante?



22. O que são **operações de repetição**?

23. Existem, a grosso modo, dois grandes tipos de operações de repetição: as operações onde uma determinada ação pode ser **repetida** 0 **ou mais vezes** e as operações onde uma determinada ação pode ser **repetida** 1 **ou mais vezes**. Isso significa que existem instruções de repetição que não garantem que a ação desejada será executada nem uma única vez, e existem instruções de repetição que garantem que a ação desejada será executada, pelo menos, uma vez.

    (a) Desenhe um fluxograma que represente uma ação de repetição que garante que uma ação desejada seja executada pelo menos uma vez:

    (b) Desenhe um fluxograma que represente uma ação de repetição que não garante que uma ação desejada seja executada pelo menos uma vez:

24. O que são **estruturas de dados**?

25. Qual a estrutura de dados mais simples?

26. Por que consideramos que:

$$\textbf{Programas} = \textbf{Algoritmos} + \textbf{Estruturas de Dados}$$

27. Por que os algoritmos devem ser expressos em um nível extremamente alto de detalhes?

28. O que queremos dizer falamos que **um algoritmo resolve todos os problemas de uma mesma "classe"**?

29. Em relação à criação de algoritmos para a solução de problemas, sabe-se que existem **problemas insolúveis**. O que são esses problemas? Que matemático provou que existem problemas insolúveis?

30. Em relação à criação de algoritmos para a solução de problemas, a Conjectura de Goldbach, "Todo inteiro par positivo maior do que 2 é a soma de dois números primos", é um exemplo de um problema para o qual nós não sabemos se existe ou não uma solução algorítmica. O que isso quer dizer?

31. O problema do "Caixeiro Viajante" é um dos problemas clássicos para os quais não existe uma solução algorítmica eficiente. Pesquise sobre esse problema e explique o que é e porque não tem solução eficiente.

32. Qual a diferença entre um **algoritmo** e uma **heurística**?

33. A definição formal de algoritmo que adotaremos neste curso é a seguinte:

> **Definição de Algoritmo**
>
> É uma coleção bem ordenada de operações efetivamente computáveis, definidas e não ambíguas que, quando executada sobre uma entrada produz uma saída e termina em uma quantidade finita de passos e de tempo

Em relação a essa definição forma, responda:

(a) O que significa "coleção bem ordenada"?

(b) O que significa "operações efetivamente computáveis"?

(c) O que significa "definidas e não ambíguas"?

(d) O que significa "sobre uma entrada"?

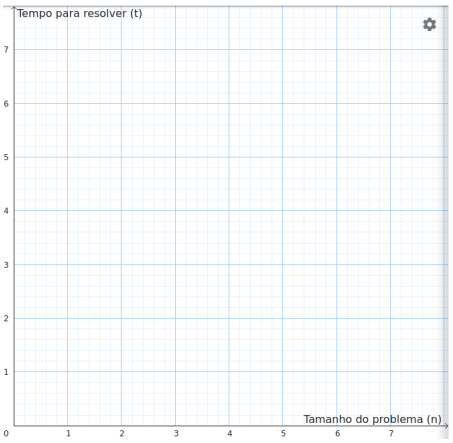
(e) O que significa "produz uma saída"?

(f) O que significa dizer que o algoritmo "termina"?

(g) O que significa dizer que o algoritmo termina "em uma quantidade finita de passos"?

(h) O que significa dizer que o algoritmo termina "em uma quantidade finita de tempo"?

34. Quando medimos a eficiência de um algoritmo geralmente estamos interessados no tempo t que esse algoritmo leva para resolver um problema de tamanho n, conforme o gráfico abaixo:



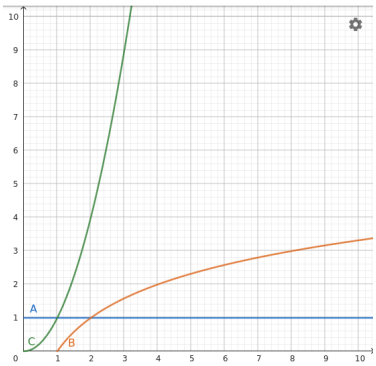
A particularidade dessa medida de eficiência dos algoritmos é que o tempo t não é medido em número de segundos ou minutos, seja, não é medido em tempo de "relógio", mas, sim, como o **número de operações que o algoritmo realiza**. Explique porque o tempo de execução é medido em número de operações realizadas e não pelo tempo de relógio.

35. O que é a notação "Big-O"? Para que ela serve?

36. Quando falamos que um algoritmo é $O(n)$, o que estamos querendo dizer?

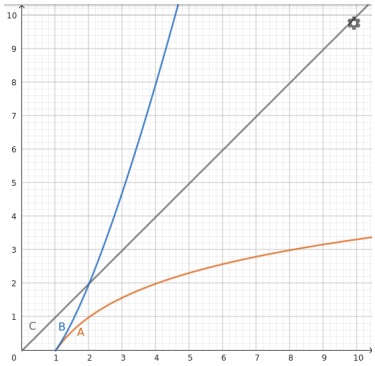
37. Quando falamos que um algoritmo é $O(\log_2 n)$, o que estamos querendo dizer?

38. A figura abaixo mostra o tempo de execução de três algoritmos diferentes para resolver o mesmo problema:



Sabe-se que esses algoritmos tem tempos de $O(1)$, $O(n^2)$ e $O(\log_2 n)$. Qual dos algoritmos acima (A, B ou C) é o algoritmo de tempo quadrático?

39. A figura abaixo mostra o tempo de execução de três algoritmos diferentes para resolver o mesmo problema:



Sabe-se que esses algoritmos tem tempos de $O(n)$, $O(n \log_2 n)$ e $O(\log_2 n)$. Qual dos algoritmos acima (A, B ou C) é o algoritmo de tempo log-linear?

40. Que algoritmo é mais lento: um de tempo $O(n \log_2 n)$ ou um de tempo $O(2^n)$?

41. Por que a linguagem natural não é um bom meio de expressar um algoritmo?

42. Por que linguagens de programação não não um bom meio de expressar um algoritmo?

43. Por que fluxogramas são um bom meio de experssar um algoritmo?

44. O que é **pseudocódigo**?

45. Por que pseudocódigo é um bom meio de expressar um algoritmo?

46. Por que não podemos começar a solucional um problema programando, ou seja, colocando a mão no teclado?

2 Pensamento Computacional

47. Cite três ou mais características presentes em **problemas complexos**:

48. O que é o **pensamento computacional**?

49. Para que serve o pensamento computacional?

50. Quais os cinco componentes principais do pensamento computacioanl:
    (a) _____
    (b) _____
    (c) _____
    (d) _____
    (e) _____

51. O que é a **decomposição**?

52. Quando fazemos a decomposição de um problema, estamos interessados em hierarquizar, modularizar e manter a regularidade entre as partes decompostas. No contexto da decomposição, o que quer dizer:
    (a) Hierarquia:
    (b) Modularidade:
    (c) Regularidade:

53. O que é o **reconhecimento de padrões**?

54. O que é, de modo geral, a **abstração**?

55. Por que a **representação de dados** é essencial no pensamento computacional?

56. O que é o **pensamento algorítmico**?

3 Abstração

57. A nossa capacidade de **abstração** é a chave para a resolução de problemas complexos na computação. Em geral a abstração tem 2 grandes objetivos: a remoção de detalhes/foco no essencial, e a capacidade de generalização. Explique o que quer dizer cada uma dessas coisas:

    (a) Remoção de detalhes/foco no essencial:

    (b) Generalização

58. O que é a **interface** fornecida por uma abstração?

59. Uma interface separa o uso da implementação. Dê um exemplo de uma abstração cuja interface permaneceu a mesma, mesmo que a implementação interna tenha mudado. Explique como isso ocorreu.

60. Por que a abstração facilita a decomposição?

61. Dê três exemplos de abstração que você utiliza no dia a dia. Indique qual é a remoção de detalhe/-foco no essencial, e a generalização de cada abstração.

62. Por que funções são um ótimo exemplo de abstração?

63. Por que a **interface** é uma barreira entre o uso e a implementação de uma abstração?

64. Por que a **interface** é um "contrato" entre quem usa e quem implementa uma abstração?

65. Qual o componente mais importante do pensamento computacional para o controle da complexidade de problemas?