Universidade Estadual de Campinas – Instituto de Computação

MO417A – Complexidade de Algoritmos – 1º Semestre de 2020

`https://www.ic.unicamp.br/~lehilton/mo417a`

Tarefa 2

- As repostas das questões devem ser manuscritas, digitalizadas e submetidas como um arquivo em formato PDF. Escaneie ou utilize um aplicativo para corrigir a perspectiva de foto. O PDF deve ser submetido em `https://www.ic.unicamp.br/~lehilton/mo417a/submit/` com a chave fornecida até a data lá anotada.
- Cada questão deve estar em páginas separadas (preferencialmente até uma, no máximo duas), seja conciso. Escreva com uma letra legível, de tamanho razoável e com bastante espaçamento entre as linhas de forma a permitir anotações. Faça um rascunho e depois passe a limpo.
- Só serão aceitas listas com todas questões respondidas, mas serão corrigidas **apenas** aquelas sorteadas na página `https://www.randomresult.com/ticket.php?t=1641336RSZPV`.

Questão 1. Considere o algoritmo a seguir:

Algoritmo 1 Algoritmo de Euclides

Procedimento EUCLIDES(a, b) ▷ Obtém o MDC de a e b
    $r \leftarrow a \bmod b$
    enquanto $r \neq 0$ **faça** ▷ Já sabemos a resposta se r é 0
        $a \leftarrow b$
        $b \leftarrow r$
        $r \leftarrow a \bmod b$
    devolve b ▷ O MDC é b

Você deverá mostrar formalmente que o algoritmo está correto utilizando uma invariante de laço.

(a) Escreva uma invariante de laço adequada para o algoritmo.
(b) Demonstre a invariante de laço.
(c) Demonstre que o algoritmo está correto utilizando a afirmação acima.

Questão 2. Seja $f(x)$ uma função real contínua e suponha que ela tem uma ou mais raízes entre a, b (uma raiz é um número $r \in (a, b)$ com $f(r) = 0$). Dado $\varepsilon > 0$, uma ε-aproximação de uma raiz, é um número $x \in (a, b)$ tal que $x \in (r - \varepsilon, r + \varepsilon)$ em que r é uma raiz. O método de aproximação da raiz baseado em busca binária executa é descrito no algoritmo a seguir:

```
Bisect(a, b, ε)
    enquanto b − a > ε:
        se f((a + b)/2) ≤ 0:
            a ← (a + b)/2
        senão:
            b ← (a + b)/2
    devolva (a + b)/2
```

(a) Analise o tempo de execução desse algoritmo em termos $b-a$ e ε.

(b) Dê condições (suficientes) sobre o entrada para que o algoritmo termine com uma resposta correta. Depois demonstre que o algoritmo está correto (quando dada uma entrada válida).

Questão 3. Considere o problema da subsequência consecutiva par máxima em que, dado um vetor, queremos encontrar uma subsequência (de números consecutivos) cuja soma seja um número par com o maior valor. Projete um algoritmo usando indução para esse problema. Analise o seu tempo de execução usando uma recorrência.

Questão 4. (Baseado em Solved Exercise 1 de Kleinberg e Tardos) Suponha que você tenha um vetor A de n números naturais. Suponha que esse vetor tem a seguinte propriedade:

- se o máximo ocorre em $A[p]$ para um índice p, então $A[1] \leq A[2] \leq \ldots A[p]$ e $A[p] \geq A[p+1] \geq \ldots A[n]$;
- se $A[i] = A[j] = x$, então $A[k] = x$ sempre que $i \leq k \leq j$.

Seu objetivo é encontrar o valor máximo nesse vetor. Escreva o algoritmo mais eficiente que conseguir. Justifique a complexidade e a correção.

Questão 5. Este é um problema real de um mestrando do IC. A entrada é formada por m listas L_i com n_i inteiros ordenados do menor para maior e um número k. A saída são as k tuplas $(a_1, a_2, \ldots, a_m)$ que têm as menores somas. A soma de uma tupla $(a_1, a_2, \ldots, a_m)$ é $a_1 + a_2 + \cdots + a_m$ onde a_i é um elemento da lista L_i. Construa o algoritmo mais eficiente que conseguir para esse problema.

Questão 6. (Manber) (6.29) A entrada são d sequências de elementos tais que cada sequência já está ordenada e há um total de n elementos. Projete um algoritmo que executa em tempo $O(n \log d)$ para juntar todas as sequências em uma única sequência ordenada.