Faculdade de Tecnologia de São Paulo
Departamento de Tecnologia da Informação
Prof. Dr. Silvio do Lago Pereira
Aux. Doc. Lucio Nunes de Lira

1. [0,5] Analise as afirmações a seguir e indique a alternativa correta.

I. *Um algoritmo é uma sequência de passos que resolve um problema, transformando dados de entrada em dados de saída.*
II. *A codificação dos passos de um algoritmo em uma linguagem de programação é conhecida como implementação.*
III. *Especificação é um contrato que descreve como um algoritmo transforma sua entrada em uma saída correspondente.*

(***a***) Apenas as afirmações I e II são verdadeiras.
(***b***) Apenas as afirmações I e III são verdadeiras.
(***c***) Apenas as afirmações II e III são verdadeiras.
(***d***) Nenhuma das anteriores.

2. [0,5] Dada uma sequência **`v`** com **`n`** itens, a função a seguir exibe a fatia de **`v`** que começa na posição **`p`** e termina na posição **`u`** (para **`p≤u`**). Com base nessa informação, complete o comando **`assert`**, que especifica a precondição desta função (para simplificar, ignore restrições sobre tipos de dados).

```
def fatia(v,p,u):

    assert _____________________________________

    for i in range(p,u+1): print(v[i])
```

3. [0,5] Analise as afirmações a seguir e indique a alternativa correta.

I. *A análise de correção verifica se um algoritmo produz saídas corretas, quando ele recebe entradas corretas.*
II. *A análise de correção empírica mostra que um algoritmo produz saídas corretas, para algumas entradas corretas.*
III. *A análise de correção formal mostra que um algoritmo produz saídas corretas, para todas as entradas corretas.*

(***a***) Apenas as afirmações I e II são verdadeiras.
(***b***) Apenas as afirmações I e III são verdadeiras.
(***c***) Apenas as afirmações II e III são verdadeiras.
(***d***) Nenhuma das anteriores.

4. [0,5] Analise as afirmações a seguir e indique a alternativa correta.

I. *Análise de eficiência visa relacionar o tamanho da entrada de um algoritmo com a quantidade de recursos computacionais que ele consome para produzir uma saída correspondente.*
II. *Análise de eficiência temporal formal usa análise assintótica de funções para relacionar o tamanho da entrada de um algoritmo com a quantidade de memória que ele consome.*
III. *Análise de eficiência temporal empírica usa experimentos automatizados para medir, registrar e avaliar o tempo consumido por um algoritmo, para diferentes tamanhos de entrada.*

(***a***) Apenas as afirmações I e II são verdadeiras.
(***b***) Apenas as afirmações I e III são verdadeiras.
(***c***) Apenas as afirmações II e III são verdadeiras.
(***d***) Nenhuma das anteriores.

5. [0,5] Assinale a alternativa cuja afirmação sobre a função **`f(n)`**, é verdadeira (supondo que **`n`** é um número natural maior que **`3`**).

```
def f(n):
    i = n//2
    while i < 2*n: i = i + i//2
    return 2*i
```

(***a***) O valor devolvido é sempre igual ao dobro da entrada.
(***b***) O valor devolvido é sempre igual ao quádruplo da entrada.
(***c***) O valor devolvido é sempre maior que o dobro da entrada.
(***d***) Nenhuma das anteriores.

6. [1,5] Crie a função **recursiva `bits(n)`**, que devolve a quantidade de bits necessários para representar o número natural **`n`** em binário.

7. [1,5] Usando **memoização** (i.e., um dicionário `m` para guardar as soluções de subproblemas previamente resolvidos), crie uma versão otimizada da função recursiva a seguir, denominada `gm()`, na qual as chamadas recursivas redundantes sejam eliminadas.

```
def g(i,j):
    if i==j: return 1
    if i>j: return g(i-1,j) + g(i,j+1)
    if i<j: return g(i+1,j) + g(i,j-1)
```

8. [1,5] Prove por **refutação**, usando MP e MT, que $\neg c$ é consequência lógica das premissas $\{a \to b, \neg a \to \neg c, b \to \neg d, d\}$.

9. [1,5] Prove por **indução** que $2^1 + 2^2 + 2^3 + \cdots + 2^n = 2^{n+1} - 2$, para todo natural $n \geq 1$.

10. [1,5] Mostre os passos necessários para obter a **forma fechada** do somatório $\sum_{i=0}^{n} \left(3i + 2^i\right)$.