



CONCURSO ITA 2025
EDITAL: 02/ITA/2025
CARGO: PESQUISADOR

PERFIL: PQ-12

CADERNO DE QUESTÕES

1. Esta prova tem duração de **4 (quatro) horas**.
2. Você poderá usar **apenas** caneta esferográfica de corpo transparente com tinta preta, lápis ou lapiseira, borracha, régua transparente simples e compasso. **É proibido portar qualquer outro material escolar ou equipamento eletrônico.**
3. Você recebeu este **caderno de questões e um caderno de respostas** que deverão ser devolvidos ao final do exame.
4. O caderno de questões é composto por 6 **questões dissertativas**.
5. As **questões dissertativas devem ser respondidas exclusivamente no caderno de respostas**. Responda sequencialmente as questões, usando caneta preta.
6. **É obrigatória a devolução do caderno de questões e do caderno de respostas**, sob pena de desclassificação do candidato.
7. **Aguarde o aviso para iniciar a prova. Ao terminá-la, avise o fiscal e aguarde-o no seu lugar.**

(2,0) Questão 1. Fale sobre as principais características do paradigma de programação orientada a objetos. Destaque as vantagens em relação ao paradigma procedural. Do ponto de vista da segurança de software, argumente sobre a orientação a objetos.

(2,0) Questão 2. A UML é uma linguagem de modelagem orientada a objetos. Explique e exemplifique o uso dos seguintes diagramas: diagrama de objetos, diagrama de classes e diagrama de sequência. Apresente exemplos no contexto da segurança de sistemas e explique a interrelação entre esses diagramas.

(1,8) Questão 3. Dado um vetor com n inteiros, descreva informalmente um algoritmo que, utilizando um heap, apresenta os k maiores elementos desse vetor em tempo $O(n \log k)$, em que $1 < k < n$.

Considere o exemplo abaixo, onde $n = 7$:

6	4	5	3	8	2	6
---	---	---	---	---	---	---

Alguns valores de k com suas correspondentes respostas:

- a. $k = 2$: 8, 6
- b. $k = 3$: 8, 6, 6
- c. $k = 4$: 8, 6, 6, 5

(1,4) Questão 4. Considere um vetor $v[1..n]$, uma pilha s inicialmente vazia, e o código `gera()` descrito a seguir:

```
void gera(int i) {
    print(s);    \\ imprime numa mesma linha todos os elementos de s
    for (k=i; k<=n; k++) {
        s.push(v[k]);
        gera(k+1);
        s.pop();
    }
}
```

Supondo $n=3$ e $v=[a,b,c]$, responda:

- a. Qual é a saída de `gera(1)`?
- b. O que faz este pseudocódigo recursivo?

(1,4) **Questão 5.** O seguinte código implementa uma estrutura de dados "HashTable" utilizada em um serviço no qual os usuários têm controle sobre o valor de "key", mas não podem fornecer valores de "key" com mais de 16 bytes.

```
uint32_t crc32(const char* data, size_t length) {
    uint32_t crc = 0xFFFFFFFF;
    uint32_t polynomial = 0xEDB88320;
    for (size_t i = 0; i < length; ++i) {
        crc ^= (uint8_t)data[i];
        for (int j = 0; j < 8; ++j)
            crc = (crc & 1) ? (crc >> 1) ^ polynomial : crc >> 1;
    }
    return crc ^ 0xFFFFFFFF;
}

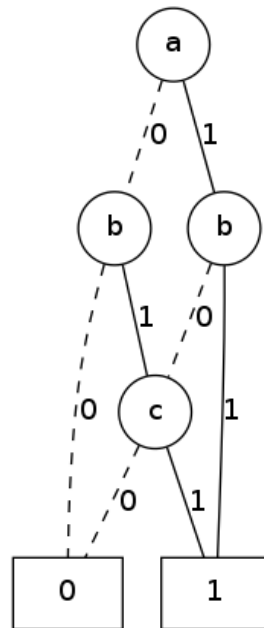
struct Node {
    char* key; int value; Node* next;
    Node(const char* k, int v) : key(strdup(k)), value(v), next(nullptr) {}
    ~Node() { free(key); }
};

class HashTable {
private:
    static const int TABLE_SIZE = 1<<27;
    Node* table[TABLE_SIZE] = {};
public:
    void insert(const char* key, int value) {
        uint32_t hash = crc32(key, strlen(key)) % TABLE_SIZE;
        Node* node = table[hash];
        while (node) {
            if (strcmp(node->key, key) == 0) {
                node->value = value;
                return;
            }
            node = node->next;
        }
        Node* newNode = new Node(key, value);
        newNode->next = table[hash];
        table[hash] = newNode;
    }
    // [...]
};
```

Por simplicidade, suponha que o servidor possui memória infinita.

- Qual pilar da segurança da informação (confidencialidade, integridade ou disponibilidade) está comprometido pela vulnerabilidade do código acima? Demonstre escrevendo o código de uma prova de conceito de ataque.
- Corrija o código para sanar a vulnerabilidade.

(1,4) Questão 6. ROBDDs (*Reduced Ordered Binary Decision Diagrams*) são grafos de decisão que representam funções booleanas. A figura abaixo apresenta, como exemplo, o ROBDD da função " $f = a \cdot b + a \cdot c + b \cdot c$ ".



Fonte: Chris Drake, 2014, pyeda.readthedocs.io.

Os vértices circulares representam variáveis de entrada, enquanto os vértices retangulares representam o resultado final da função. As arestas representam os valores que cada variável pode assumir. Para avaliar a função, inicia-se no vértice correspondente à primeira variável (no exemplo, a) e percorrem-se as arestas correspondentes aos valores de entrada até atingir um vértice final (retangular).

Seja n a quantidade de variáveis booleanas de entrada de uma função f qualquer representada por um ROBDD. Responda às seguintes perguntas considerando os melhores algoritmos atualmente conhecidos.

- No pior caso, qual a complexidade assintótica de espaço do ROBDD? Justifique sua resposta.
- No pior caso, qual a complexidade assintótica do tempo de execução para construir o ROBDD a partir de f ? Justifique sua resposta.
- No pior caso, qual a complexidade assintótica do tempo de execução para avaliar o resultado da função para um conjunto de entradas, a partir de um ROBDD previamente construído? Justifique sua resposta.
- Que implicações a descoberta de algoritmos mais eficientes (com menor complexidade assintótica) para ROBDD traria para as aplicações de criptografia? Discuta exemplos.

RASCUNHO

RASCUNHO

RASCUNHO

RASCUNHO
