UNIVERSIDADE da MADEIRA

Exame - Época de Recurso

Nome: _____

Nº Aluno: _____

Curso: LEI / LEET /LEC | **Unidade Curricular:** Estruturas de Dados e Algoritmos | **Ano letivo:** 2023/2024
Docentes: Filipe Quintal, Karolina Baras, Fábio Mendonça | **SEM CONSULTA**
Data: 12/06/2024 | **Duração:** 2h 30 m (exame global), 1h 30m (1ª ou 2ª frequência) | **Tolerância:** 10 minutos
---

Notas:
- Qualquer tentativa de fraude implica a anulação do teste;
- Utilize uma caligrafia legível.
- Assuma a inclusão da biblioteca <iostream> e a utilização do namespace std em todas as questões da frequência

Grupo 1 (10 valores)

	1. Classifique as afirmações seguintes como verdadeiras (V) ou falsas (F) (1.75 valores)	V	F
a	Nalguns casos, a função main() não é a primeira a ser executada.		
b	O casting permite converter o resultado de uma operação para um determinado tipo de dados.		
c	No ciclo *while*, o corpo do ciclo é executado o mesmo número de vezes que a condição de paragem**.**		
d	Se int x = 2; int* px = &x; a instrução cout << *px+1; irá escrever 3 na consola.		
e	A função *sizeof*(tipoDados) devolve o espaço necessário em bytes para alocar em memória uma variáveldo tipo de dados indicado.		
f	A operação 1 && 0 na linguagem C/C++ dá 0.		
g	O algoritmo Selection Sort aplicado ao vetor 2, 11, 6, 10, 7 executa apenas duas trocas para ordenar o vetor por ordem crescente.		

2. Considere a struct abaixo que define um item para a implementação do tipo de dados abstratos da Fila:

a. Implemente a função **int conta(Fila& f, int num)** que recebe uma Fila por referência e um inteiro e retorna o número de ocorrências desse inteiro na Fila (assuma que a Fila foi correctamente inicializada) (0.75 valores)

```
struct Fila{
    struct Item{
        int valor;
        Item* next;
    };
    Item * ap_inicio;
};
```

b. Complete o código abaixo responsável por retornar verdadeiro ou falso conforme o número passado no argumento exista ou não na fila. Indique o código a colocar em a), b), c), d) e e). (1 valor)

```
a) existe(Fila & f, int num){
     Fila::Item* apt = b);
     while(apt !=NULL)
        if(apt->valor c) num)
           return true;
        apt = d);
     }
     return e);
}
```

a) **b)** **c)** **d)** **e)**

3. Especifique o código responsável por

a. Declarar um array dinâmico, "**melhor_jogador"**, que será usado para guardar os nomes dos melhores jogadores de cada jornada de um campeonato de futebol. Este campeonato terá 20 jornadas (0.75 valores).

b. Implementar a função **contaPremio** que recebe o array definido acima, o seu tamanho e o nome de um jogador. Esta função deverá retornar o número de vezes que o jogador recebido foi eleito melhor jogador (1 valor).

4. Considere a struct à direita que representa um elemento de uma lista ligada de alunos. Cada aluno é identificado por: um número de aluno, o seu nome, o número de disciplinas em que está inscrito, um array com uma nota para cada disciplina, e um apontador para o aluno seguinte:

```
struct aluno{
    int numero;
    string nome;
    int num_disciplinas;
    float * notas;
    aluno * seguinte;
}
```

a. Especifique o código responsável por implementar a função criaAluno, que recebe o número, nome, e número de disciplinas. Esta função deverá alocar memória para um novo aluno e retornar um apontador para o mesmo (0.75 valores).

b. Complete o código em a), b), c), d) responsável por implementar a função **void imprimeNotas(alunos * inicio, int numero)**. Que recebe um apontador para a lista ligada de alunos, e um número de aluno. Esta função deverá imprimir as notas do aluno com o número recebido (1.25 valores).

```
void imprimeNotas(aluno * inicio, int numero){
    aluno * it = inicio;
    while( a) ){
        if( b) ){
            for(int i = 0; i< c) ;i++)
                cout << it-> d) <<endl;

            break;
        }
        it = it->seguinte;
    }
}
```

a) **b)** **c)** **d)**

5. Considere o código abaixo:

```
void modifyValue(int x) {
    x = 10;
}

void modifyReference(int &x) {
    x = 10;
}

int main() {
    int a = 5;
    int b = 5;

    modifyValue(a);
    modifyReference(b);

    return 0;
}
```

Quais serão os valores de 'a' e 'b' após a execução da função main. Justifique a sua resposta (1 valor)

6. Descreva sucintamente as principais diferenças entre a utilização de arrays vs lista ligadas, para a inserção de elementos no início de uma lista/array (0.75 valores).

7. Considere o programa abaixo e indique o seu output (assuma todos os includes necessários ao seu funcionamento) (1 valor)

Output:

```
int main()
{
    int i=0;
    float v[]   = {1,2,3,4,5,6,7};
    float value = v[++i];
    float * pv = &value;
    cout << value++<<endl;
    cout << *pv<<endl;
    pv = v+2;
    cout << *pv<<endl;
    cout <<pv[1]<<endl;
    return 0;
}
```

UNIVERSIDADE da MADEIRA

Nome: _____

Nº Aluno: _____

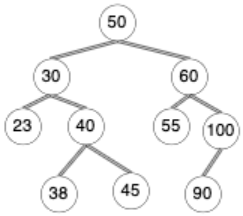
Curso: LEI / LEET /LEC | **Unidade Curricular:** Estruturas de Dados e Algoritmos | **Ano letivo:** 2023/2024
Docentes: Filipe Quintal, Karolina Baras, Fábio Mendonça| **SEM CONSULTA**
Data: 12/06/2024 | **Duração:** 2h 30 m (exame global), 1h 30m (1ª ou 2ª frequência) | **Tolerância:** 10 minutos
Notas:
● Qualquer tentativa de fraude implica a anulação do teste;
● Utilize uma caligrafia legível.
● Assuma a inclusão da biblioteca <iostream> e a utilização do namespace std em todas as questões da frequência

Grupo 2 (10 valores)

	8. Classifique as afirmações seguintes como verdadeiras (V) ou falsas (F) (1.75 valores)	V	F
a	Uma pesquisa numa árvore de pesquisa binária tem como pior caso uma complexidade O(log n)		
b	Numa travessia sufixa a raiz da árvore de pesquisa binária será o último nó a ser visitado		
c	A utilização da função de Pesquisa Linear (Linear Probing) elimina o efeito de agrupamento que acontece em tabelas de hash com endereçamento aberto.		
d	Num grafo completo, não orientado, a matriz de incidências terá uma dimensão de $n^2.(n-1)/2$		
e	No limite posições DELETED numa tabela hash, poderão levar a uma complexidade de pesquisa com uma complexidade $O(n^2)$.		
f	Uma travessia em Largura num grafo não orientado conexo, irá garantidamente visitar todos os nós do mesmo		
g	Uma função recursiva poderá ser implementada sem uma condição de paragem, desde que as chamadas recursivas sejam feitas para conjuntos menores do input.		

9. Descreva sucintamente o fenómeno conhecido como clustering que poderá acontecer durante o endereçamento de uma tabela hash. Em que tipo de endereçamento poderá acontecer este fenómeno? Como poderá ser resolvido? (0.75 valor).

10. Considere a árvore de pesquisa binária à direita.



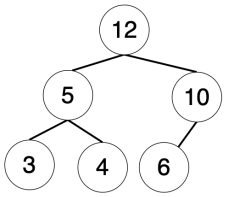
a. Indique que valores poderão ser inseridos (consecutivamente) na mesma para tornar esta árvore completa (0.5 valores).

b. Após inserir o valor 49 à direita do nó 45 a árvore tornar-se-á desequilibrada. Seguindo o algoritmo AVL efetue as rotações necessárias para que a mesma volte a estar equilibrada, apresente todos os passos intermédios (1 valor).

11. Preencha a tabela abaixo com a complexidade dos algoritmos de ordenação para o melhor e pior caso (n representa o tamanho da lista a ordenar) (0.75 valor):

Algoritmo	Melhor Caso	Pior Caso
Insertion Sort	**n**	**n^2**
Selection Sort		**n^2**
Bubble Sort		**n^2**
Merge Sort		
Quick Sort	**n.log(n)**	

12. Considere o heap à direita. Aplique o algoritmo heapsort à representação vetorial da árvore. Apresente todos os cálculos intermédios (estado da árvore e array). (1 valor)



13. Considere a estrutura abaixo que define um nó de uma árvore de pesquisa binária. Indique o código que:

a. Implementa a função bool folha(node * no) que recebe um apontador para nó de uma árvore de pesquisa binária e retorna um booleano indicando se este nó é uma folha . (0.75 valores)

```
struct nodo {
   int dados;
   nodo* esquerda;
   nodo* direita;
};
```

b. Implementa a função insereIt, que recebe um apontador para a raiz da árvore e um apontador para um novo nodo. Esta função deverá inserir o nodo recebido na árvore de forma iterativa. NOTA:Poderá assumir que é recebida uma árvore não vazia.(1.25 valores).

14. Considere o grafo orientado apresentado abaixo.

a. Represente a matriz de adjacências para este grafo (0.75)

b. Indique o caminho mais curto entre A e G utilizando o algoritmo de Dijkstra. Apresente todos os cálculos intermédios (tabela com as distâncias e antecessores). (1.5 valores)