Entregar todas as respostas em um arquivo pdf.

1) (1,5 ponto) Verifique se as afirmações abaixo são verdadeiras ou falsas, justifique as falsas.

a) Se *2-CNF-SAT* $\leq_p$ *2-GRAPH-COLORING* então $P = NP$.

b) Se $P = NP$ então qualquer problema pode ser resolvido em tempo polinomial.

c) Considerando um problema P_1 que tem uma solução em tempo polinomial conhecida, e um problema P_2 que é *NP-Completo*, apresentando uma redução, que pode ser executada em tempo exponencial, de P_2 a P_1 ($P_2 \leq_e P_1$) estamos provando que $P = NP$.

d) Considerando dois problemas A e B, se $A \leq_p B$ e $A \in P$ então $B \in P$.

e) Considerando dois problemas A e B, se $A \leq_p B$ e $B \in P$ então $A \in P$.

f) *NP-Hard* $\subseteq NP$.

2) (1,5 ponto) Resolva a seguinte relação de recorrência:

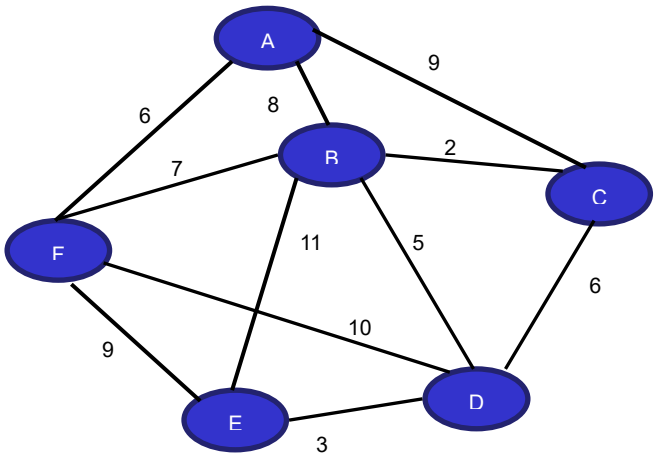
$$T(n) = T(n/2) + n.log_2\, n$$
$$T(1) = 1$$

3) (1,5 ponto) Para a função ***questao***, definida abaixo, mostre qual a relação de recorrência que descreve a complexidade de tempo de execução da função. Resolva essa relação de recorrência. Calcule a complexidade de tempo e a complexidade espaço da função. Na chamada inicial da função são passados a posição inicial e a posição final do vetor para as variáveis *p* e *r*.

```
void questao(int *v, int p, int r, int *min, int *max)
{
    int q, min1, max1;

    if (p < r)
    {
        q = (p + r)/2;
        questao(v, p, q, &min1, &max1);
        questao(v, q + 1, r, min, max);
        if (min1 < *min)
                *min = min1;
        if (max1 > *max)
                *max = max1;
    }
    else
    {
        *min = v[p];
        *max = v[p];
    }
}
```

4) (3,0 ,pontos) Mostre cada passo da execução do algoritmo de Kruskal para encontrar a árvore geradora mínima do grafo abaixo. Mostre o algoritmo e a complexidade de tempo de cada *loop* e função auxiliar. Informe a complexidade de tempo do algoritmo.



5) (0,5 ponto) Prove que o problema *SUBSET-SUM* pertence à classe *NP*.

6) (2,0 pontos) Reduza a instância abaixo de *3-CNF-SAT* ao problema do *SUBSET-SUM* (usando o algoritmo apresentado no livro: *Introduction to Algorithms*, Thomas H. Cormen et al.), mostre que uma atribuição de variáveis que torne a expressão verdade e a solução correspondente à essa atribuição na instância do SUBSET-SUM criada na redução:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$$