In [3]:

```
class TSP():
    def load_data_tsp(self, arquivo, fim_cabecalho):
        data = {}
        cabecalho = {}
        num_linha = 1

        arquivo = open(arquivo, "r")
        for linha in arquivo:
            if linha.startswith('EOF'):
                break

            if num_linha <= fim_cabecalho:
                dados = str(linha).strip('\n').split(':')
                dados_limpos = [item.strip() for item in dados]

                try:
                    cabecalho[dados_limpos[0]] = dados_limpos[1]
                except:
                    pass

            else:
                dados = str(linha).strip('\n').split(' ')

                filtro = set([''])
                dados_limpos = [dado for dado in dados if dado not in filtro]

                data[dados_limpos[0]] = (int(dados_limpos[1]), int(dados_limpos[2]))

            num_linha += 1

        arquivo.close()

        self.data = data
        self.nome = cabecalho['NAME']
        self.problema = cabecalho['COMMENT']
        self.tipo = cabecalho['TYPE']
        self.tamanho = int(cabecalho['DIMENSION'])
        self.calculo = cabecalho['EDGE_WEIGHT_TYPE']

    def __init__(self, problema):
        self.load_data_tsp('input/{}'.format(problema), 6)

    def __repr__(self):
        return 'NOME : {}\nPROBLEMA : {}\nTIPO : {}\nTAMNHO: {}\nCALCULO : {}'.format(s
elf.nome, self.problema, self.tipo, self.tamanho, self.calculo)
```