TP 1 EX 1

Grupo 5:

Breno Fernando Guerra Marrão A97768

Tales André Rovaris Machado

Inicialização

Usamos as bibliotecas Or-tools (com interface CP_model) e random para resovler o problema de alocação proposto

In [1]:
```
from ortools.sat.python import cp_model
from random import randrange
crip = cp_model.CpModel()

def L(m,n):
    return l[m][n]
```

Inicialização da Matriz L e declaração das variaveis

In [2]:
```
l = {}
d = 30
M = 32
N = 8
q = 71

for m in range(M):
    l[m] = {}
    for n in range(N):
        l[m][n] = randrange(-d,d+1)
```

Inicialização do vetor e, e do vetor auxiliar k de tamanho m que são as soluções se possiveis da operação mod

In [3]:
```
e = []

for m in range(M):
    e.append(crip.NewIntVar(-1,1,f'e[{m}]'))

k = []
for n in range(N):
    k.append(crip.NewIntVar(-round(d*M/q),round(d*M/q),f'k[{n}]'))
```

Para certificar que o vetor e não é nulo devemos conferir os valores absolutos do minimo e maximo do vetor e aplicar a operação logica ou para certificar que todos são diferentes de 0

$$|max_e| \vee |min_e|$$

In [4]:
```
#checar se todos os elementos de e são nulos

max_vec = crip.NewIntVar(-1, 1, "maxvec")
crip.AddMaxEquality(max_vec, e)

min_vec = crip.NewIntVar(-1, 1, " minvec")
crip.AddMinEquality(min_vec, e)

absmin = crip.NewBoolVar('absmin')
absmax = crip.NewBoolVar('absmax')
crip.AddAbsEquality(target = absmin, expr = min_vec)
crip.AddAbsEquality(target = absmax, expr = max_vec)


crip.AddBoolOr(absmax,absmin)
pass
```

$$\forall i < n \cdot \sum_{j<m} e_j \times \mathsf{L}_{j,i} \equiv 0 \mod q$$

In [5]:
```
for i in range(N):
    crip.Add(sum(L(j,i)*e[j] for j in range(M))== q*k[i])
```

Execução do solver

In [ ]:
```
solver = cp_model.CpSolver()
status = solver.Solve(crip)


if status == cp_model.OPTIMAL or status == cp_model.FEASIBLE:
    print("Matriz :")
    for a in range(N):
        arr = []
        for b in range(M):
            arr.append(L(b,a))
        print(arr)
    print()
    vec = []
    for n in range(M):
        vec.append(solver.Value(e[n]))
    print("vetor e: ")
    print(vec)
else:
    print("IMPOSSIVEL")
```