PROVA 2 – LLP

Arthur Severo de Souza - 20183021106
Victor Le Roy - 20183021222

a) Desenvolva esse jogo em Haskell.

```haskell
-- Auxiliares
tamanho :: [Int] -> Int
tamanho [] = 0
tamanho (a:x) = 1 + tamanho x

qsort :: [Int] -> [Int]
qsort [] = []
qsort (a:x) = qsort [ b | b <- x, b <= a ] ++ [a] ++ qsort [ b | b <- x, b > a ]

-- Execucao do jogo
falta :: Int -> Int -> [Int]
falta x y
  | x+1 == y = []
  | x+1 < y = [x+1] ++ falta (x+1) y
  | otherwise = []

execucao :: [Int] -> [Int]
execucao [] = []
execucao (x:y:l)
  | tamanho(y:l) > 1 = (falta x y) ++ execucao (y:l)
  | otherwise = falta x y ++ execucao l

game :: [Int] -> [Int]
game l = execucao (qsort l)
```

```
ghci> game [3, 1, 4, 6, 5]
[2]
ghci> game [3, 1, 2, 4]
[]
ghci> game [2, 6, 7, 5]
[3,4]
```

b) Desenvolva esse jogo em Prolog.

```
maior([N], N).
maior([N|L], MAIOR) :- maior(L, MAIOR), MAIOR > N.
maior([N|L], N) :- maior(L, MAIOR), N > MAIOR.

menor([N], N).
menor([N|L], MENOR) :- menor(L, MENOR), MENOR < N.
menor([N|L], N) :- menor(L, MENOR), N < MENOR.

gerar(FINAL, FINAL, [FINAL]).
gerar(PRIMEIRO, FINAL, [PRIMEIRO|L]) :- N is PRIMEIRO + 1, gerar(N, FINAL, L).

remover_elemento(_, [], []).
remover_elemento(N1, [N1|L], LR) :- remover_elemento(N1, L, LR).
remover_elemento(N1, [N2|L], [N2|LR]) :- N1 \== N2, remover_elemento(N1, L, LR).

remover([],[],[]).
remover([], GERADO, L) :- L = GERADO.
remover([CABECA_LISTA|LISTA], GERADO, L) :- remover_elemento(CABECA_LISTA, GERADO, GERADO_RESULTADO), remover(LISTA, GERADO_RESULTADO, L).

game([],[]).
game(LISTA,L) :- menor(LISTA, MENOR), maior(LISTA, MAIOR), gerar(MENOR, MAIOR, GERADO), remover(LISTA, GERADO, L).
```

```
1 ?- game([3, 1, 4, 6, 5], X).
X = [2] .

2 ?- game([3, 1, 2, 4], X).
X = [] .

3 ?- game([2, 6, 7, 5], X).
X = [3, 4] .
```

Obs: Todos os programas estão na localizados na pasta code.