

**UNIDADE CURRICULAR:** Arquitetura de Computadores

**CÓDIGO:** 21010

**DOCENTE:** Gracinda Carvalho, José Coelho

**NOME:** Paulo Jorge Martins Nicolau

**N.º DE ESTUDANTE:** 1800465

**CURSO:** Licenciatura em Engenharia Informática

**DATA DE ENTREGA:** 20 de Janeiro de 2020

**TRABALHO / RESOLUÇÃO:**

Para realizar o trabalho foi utilizado a versão online do simulador de P3, disponibilizado na plataforma de e-learning, de modo a testar o código realizado.

**Alínea A:**

Para esta alínea pretendia-se criar um programa que permita calcular a distância de Manhattan do espaço vazio até à posição final num puzzle com 15 peças (vetor de tamanho 16).

Para este programa decidi separar o código em duas secções, sendo que na primeira secção iria procurar a posição no vetor onde se encontrava o espaço vazio (peça 16), e a segunda secção iria com base nessa posição realizar o calculo da distância de Manhattan.

Na procura da posição do espaço vazio, vai-se atribuir o primeiro valor do vetor a um registo, e vai-se percorrendo as posições de memória do vetor verificando se o valor dessa posição é maior que a do registo, trocando em caso afirmativo. No final é guardado numa posição de memoria o valor da posição.

Foi reservado na posição seguinte ao vetor, um espaço vazio (com valor 0) que vai servir para indicar que o vetor chegou ao final.

Tendo em conta que o vetor possui 16 posições de tamanho, a posição final, num quadrado com tamanho 4, será (4,4). Já a posição inicial do espaço vazio pode variar assim foi necessário utilizar as seguintes formulas para calcular o ponto num espaço bidimensional a partir da posição encontrada na secção inicial do programa.

Para obter a posição do espaço vazio no vetor:

Para converter essa posição, numa posição de uma matriz :

* é o quociente da divisão , onde neste caso, e é o resto da divisão , fazendo em seguida as seguintes correções:
  + No caso de
  + No caso de

Após obter a coordenada bidimensional calcula-se o valor da distância de Manhattan através da formula:

**Alínea B:**

Para esta alínea era pedido um programa que permita calcular as inversões do vetor. Para realizar essa tarefa, foram utilizados dois registos, que vão servir de índice, de forma a criar dois ciclos , um interno e outro externo, e ao percorrer o ciclo interno vai-se verificando se o valor no registo do índice interno é menor do que o valor do registo do índice externo, e caso seja vai ser incrementado o valor de um registo que serve de contador.

**Alínea C:**

Nesta alínea pretende-se verificar se o puzzle que é fornecido é um puzzle que pode ser resolvido. Para isso, utilizei o código desenvolvido anteriormente, para obter a distancia de Manhattan e o total de inversões, e depois, a secção de código novo vai copiar o valor dos registos R1 e R2 para outros registos, e depois vai adicionar os valores desses registos para obter a soma. Em seguida, avalia a paridade do valor da soma indicando no caso de ser ímpar 1 ou par 0 no registo R3.

Segue-se os testes realizados, com os exemplos fornecidos:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Resolúvel** | **Ciclos de Relógio** | **Instruções** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Alínea D:**

Esta alínea consistia na mesma tarefa que a alínea C, no entanto possibilitando que fosse possível avaliar puzzles de outros tamanhos. Para isso, foi reservado em memória, antes do vetor um espaço com o valor do tamanho das linhas/colunas da matriz do puzzle.

Ao longo do código, foi substituido todas as instruções onde se avaliava o tamanho como sendo 4 para ler o valor desse espaço de memória.

Segue-se os testes realizados, com os exemplos fornecidos:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome** | **Dimensão** | **Resolúvel** | **Ciclos de Relógio** | **Instruções** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |