|  |  |  |
| --- | --- | --- |
| Function | Time Analysis | Space Analysis |
| findLetter(char L) | Theta(n) | Theta(1) |
| printIDtable() | Theta(n) | Theta(1) |
| fillGraph() | For n times  For m times //edges  findLetter 4N  Theta(4n^2 \* m) //n vertices  //m edges | n //IDtable.resize()  Theta(n) |
| printGraph() | Theta(n^2) | Theta(1) |
| maxCover(std::vector<char> &minPerm) | Theta(n!) | Theta(n) //minCover = minPerm.size() |
| cover(std::vector<char> order) | Theta(n^2) | Theta(1) |
| Main() | For g times //numGraphs  fillGraph() //n^4\*m  maxCover() //n!  Theta(g((4n^2\*m) + n! )) | n // answer(numVertices);  n //new Graph(numVertices);  n //fillGraph()  n //maxCover()  Theta(n) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Reason for the test | Actual Input Data | Expected Output Data | Actual Output |
| 1 | Test for a single vertex with no edges | 1 1 a 0 | a 0 | a 0 |
| 2 | Test star graph with four edges | 1 5 a 1 e b 1 e c 1 e d 1 e e 4 a b c d | a b e c d 2 | a b e c d 2 |
| 3 | Test for multiple graphs | 2 1 a 0 1 5 a 1 e b 1 e c 1 e d 1 e e 4 a b c d | a 0 a b c d e 2 | a 0 a b c d e 2 |
| 4 | Full graph (each vertex has an edge to each vertex) for 5 vertices | 1 5 a 4 b c d e b 4 a c d e c 4 a b d e d 4 a b c e e 4 a b c d | a b c d e 4 | a b c d e 4 |
| ~~5~~ 26! Was too much for my computer to calculate so I am replacing this test with test 4, a smaller full graph | ~~Full graph (each vertex has an edge to each vertex) for all possible letters~~ | ~~1 26 a 25 b c d e f g h i j k l m n o p q r s t u v w x y z b 25 a c d e f g h i j k l m n o p q r s t u v w x y z c 25 a b d e f g h i j k l m n o p q r s t u v w x y z d 25 a b c e f g h i j k l m n o p q r s t u v w x y z e 25 a b c d f g h i j k l m n o p q r s t u v w x y z f 25 a b c d e g h i j k l m n o p q r s t u v w x y z g 25 a b c d e f h i j k l m n o p q r s t u v w x y z h 25 a b c d e f g i j k l m n o p q r s t u v w x y z i 25 a b c d e f g h j k l m n o p q r s t u v w x y z j 25 a b c d e f g h i k l m n o p q r s t u v w x y z k 25 a b c d e f g h i j l m n o p q r s t u v w x y z l 25 a b c d e f g h i j k m n o p q r s t u v w x y z m 25 a b c d e f g h i j k l n o p q r s t u v w x y z n 25 a b c d e f g h i j k l m o p q r s t u v w x y z o 25 a b c d e f g h i j k l m n p q r s t u v w x y z p 25 a b c d e f g h i j k l m n o q r s t u v w x y z q 25 a b c d e f g h i j k l m n o p r s t u v w x y z r 25 a b c d e f g h i j k l m n o p q s t u v w x y z s 25 a b c d e f g h i j k l m n o p q r t u v w x y z t 25 a b c d e f g h i j k l m n o p q r s u v w x y z u 25 a b c d e f g h i j k l m n o p q r s t v w x y z v 25 a b c d e f g h i j k l m n o p q r s t u w x y z w 25 a b c d e f g h i j k l m n o p q r s t u v x y z x 25 a b c d e f g h i j k l m n o p q r s t u v w y z y 25 a b c d e f g h i j k l m n o p q r s t u v w x z z 25 a b c d e f g h i j k l m n o p q r s t u v w x y~~ | ~~a b c d e f g h i j k l m n o p q r s t u v w x y z 25~~ | ~~a b c d e f g h i j k l m n o p q r s t u v w x y z 25~~ |