**Обхождане на граф – упражнение**

**Зад 1**. Да се напише програма, която **обхожда в ширина граф** зададен със **списъци на съседите.**

**Даден е следния код:**

static void Main()

{

Graph g = new Graph(new int[][] {

new int[] {3, 6}, // наследници на връх 0

new int[] {2, 3, 4, 5, 6}, // наследници на връх 1

new int[] {1, 4, 5}, // наследници на връх 2

new int[] {0, 1, 5}, // наследници на връх 3

new int[] {1, 2, 6}, // наследници на връх 4

new int[] {1, 2, 3}, // наследници на връх 5

new int[] {0, 1, 4} // наследници на връх 6

});

}

**Изход:**

Breadth-First Search (**BFS**) traversal: **0 3 6 1 5 4 2**

**Зад 2**. Да се напише програма, която **обхожда в дълбочина граф** зададен със **списъци на съседите.**

**Даден е следния код:**

static void Main()

{

Graph g = new Graph(new int[][] {

new int[] {3, 6}, // наследници на връх 0

new int[] {2, 3, 4, 5, 6}, // наследници на връх 1

new int[] {1, 4, 5}, // наследници на връх 2

new int[] {0, 1, 5}, // наследници на връх 3

new int[] {1, 2, 6}, // наследници на връх 4

new int[] {1, 2, 3}, // наследници на връх 5

new int[] {0, 1, 4} // наследници на връх 6

});

}

**Изход:**

Depth-First Search (**DFS**) traversal (with **stack**): **0 6 4 2 5 1 3**

**Зад 3**. Да се напише програма, която **обхожда рекурсивно в дълбочина граф** зададен със **списъци на съседите.**

**Даден е следния код:**

static void Main()

{

Graph g = new Graph(new int[][] {

new int[] {3, 6}, // наследници на връх 0

new int[] {2, 3, 4, 5, 6}, // наследници на връх 1

new int[] {1, 4, 5}, // наследници на връх 2

new int[] {0, 1, 5}, // наследници на връх 3

new int[] {1, 2, 6}, // наследници на връх 4

new int[] {1, 2, 3}, // наследници на връх 5

new int[] {0, 1, 4} // наследници на връх 6

});

}

**Изход:**

Depth-First Search (**DFS**) traversal (**recursive**): **0 3 1 2 4 6 5**