**HW [6], [2/20/2019] MCS 253P**

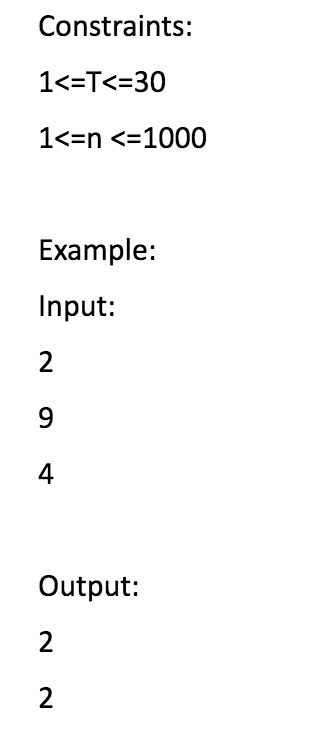
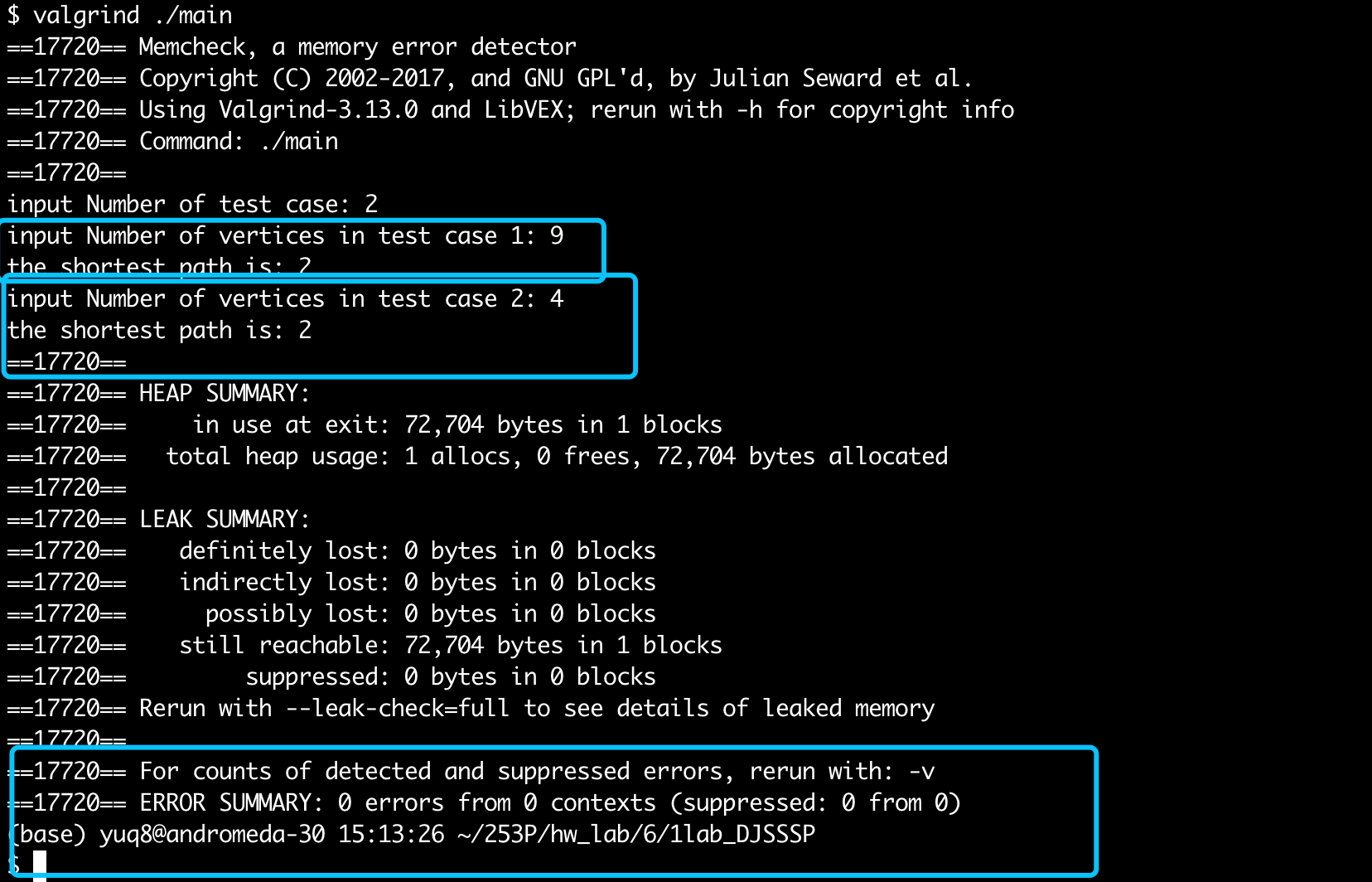
**Name: Yu Qin**

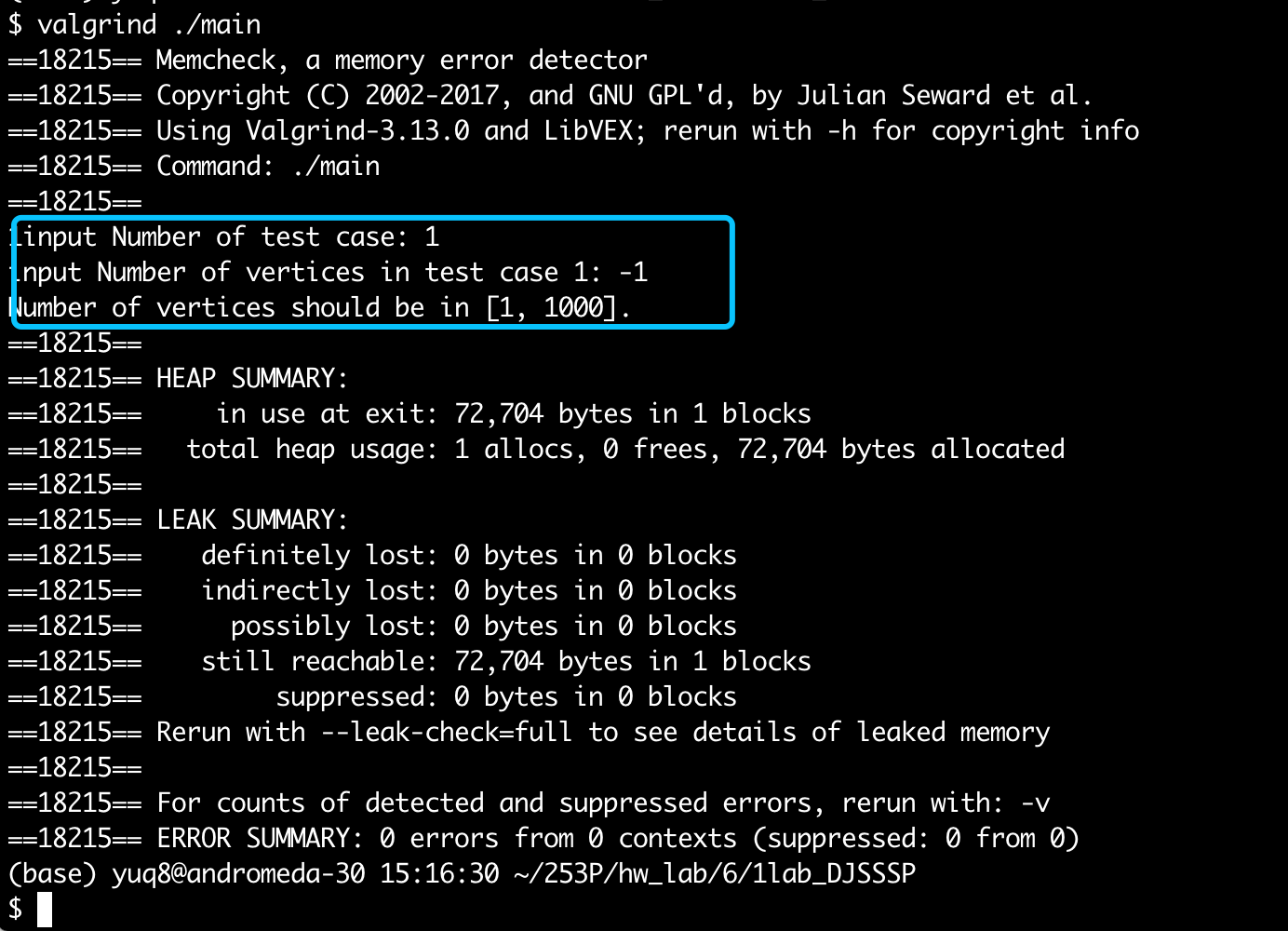
**Results Screenshots**

===========================start of the write-up====================

# **Lab 1: SSSP**

test case 1,2,3:





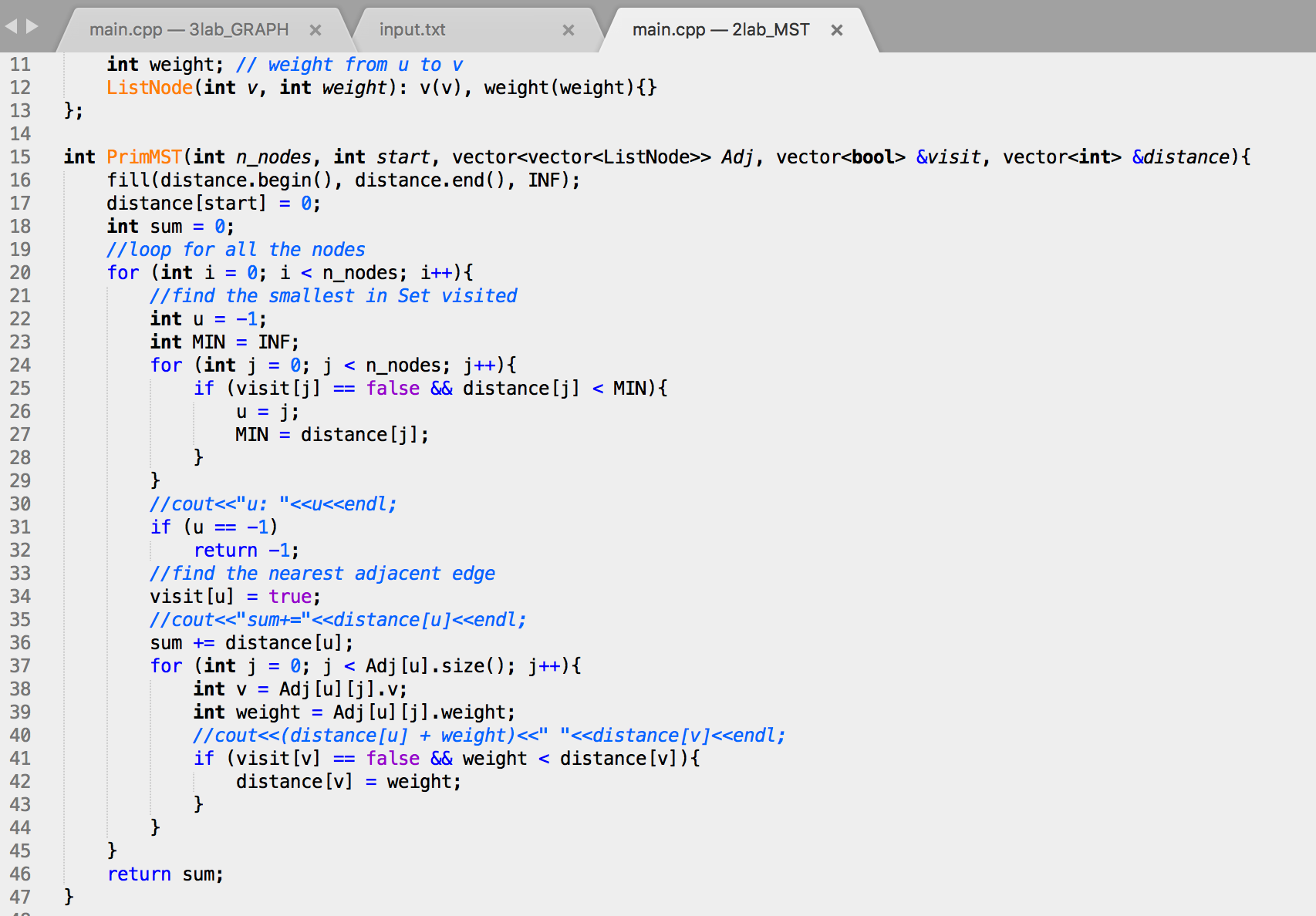
Conclusion: right result, 0 error in memory leak.

Little Notes:

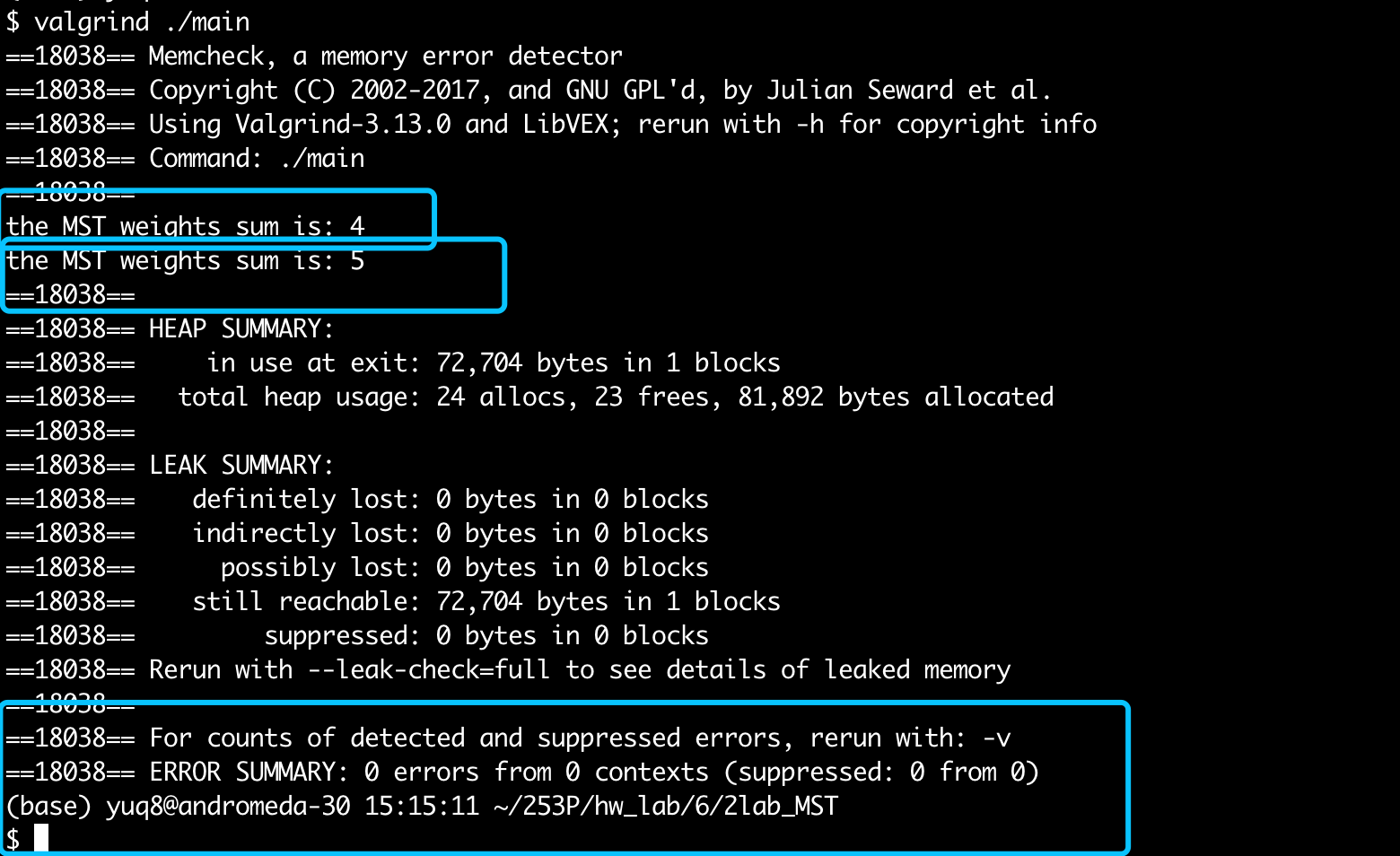
1. There are many ways to implement.

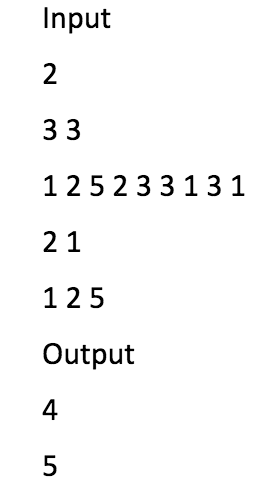
# Lab 2: MST

Code:



Test cases 1, 2:





Conclusion: right result, 0 error in memory leak.

Little Notes:

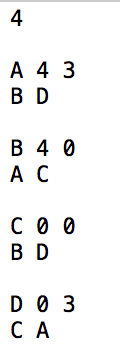
1. Can be implemented in Prim and Krustal.

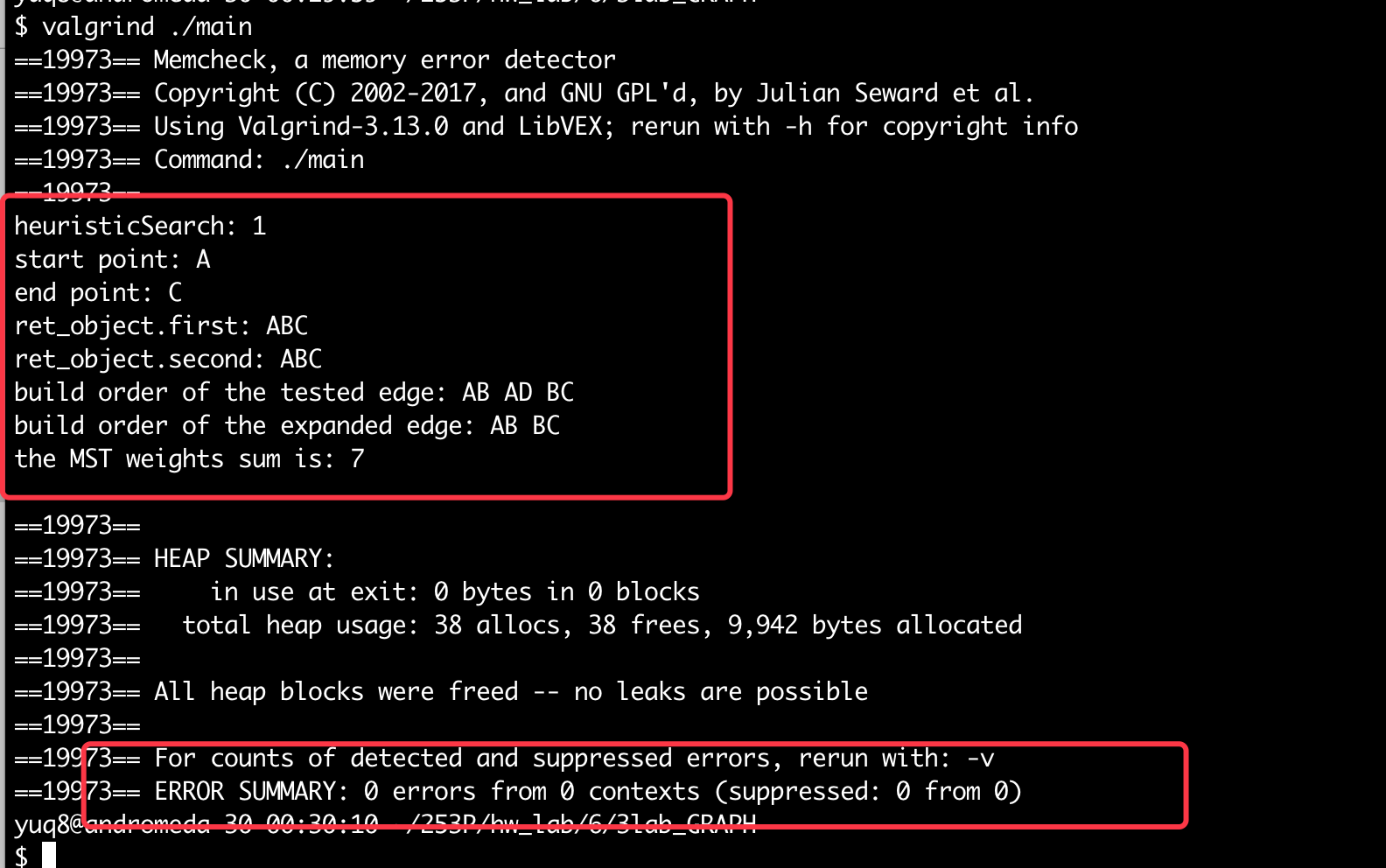
# Lab3:

**Code：**

****

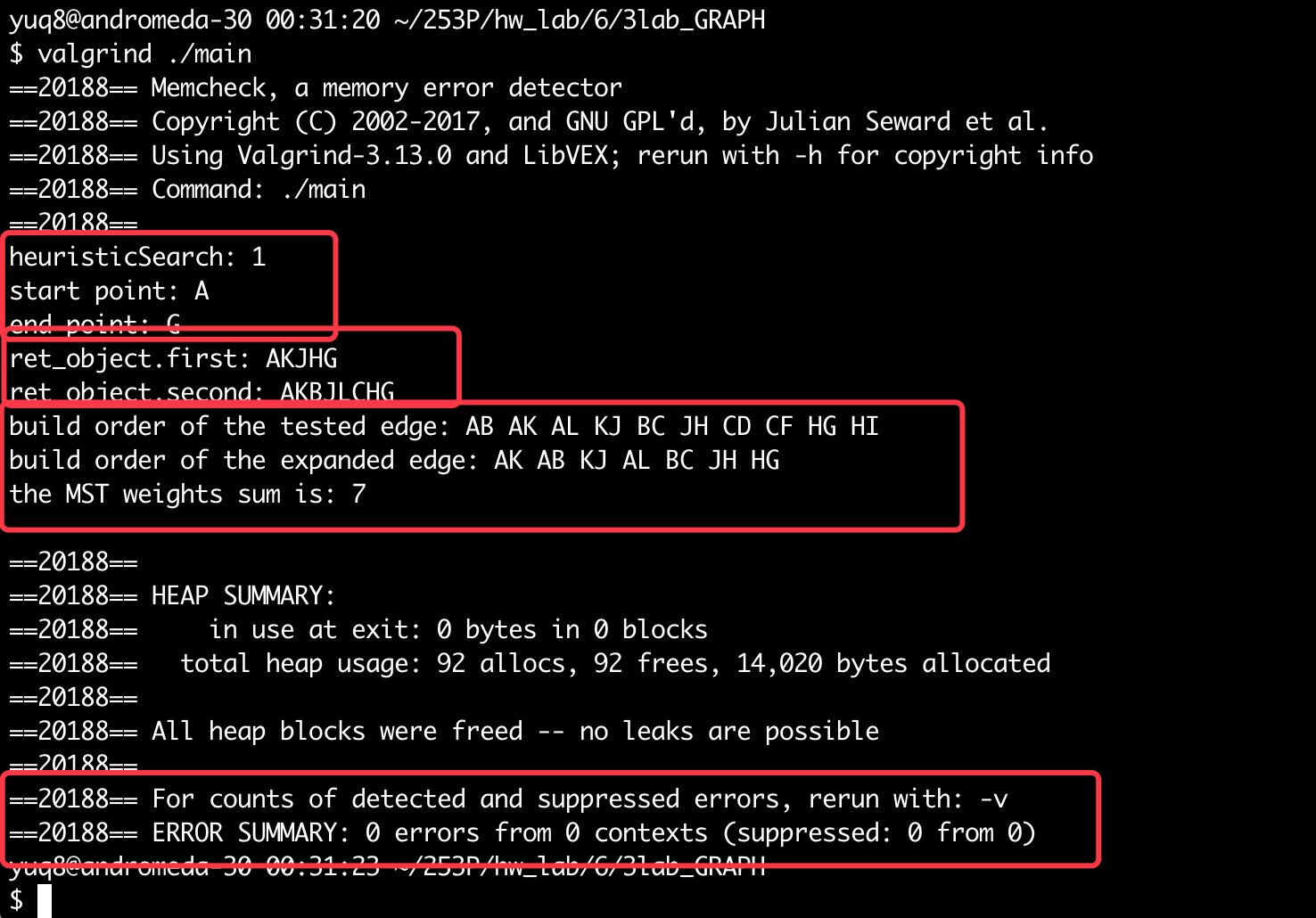
**test cases 1: with easy input**

****

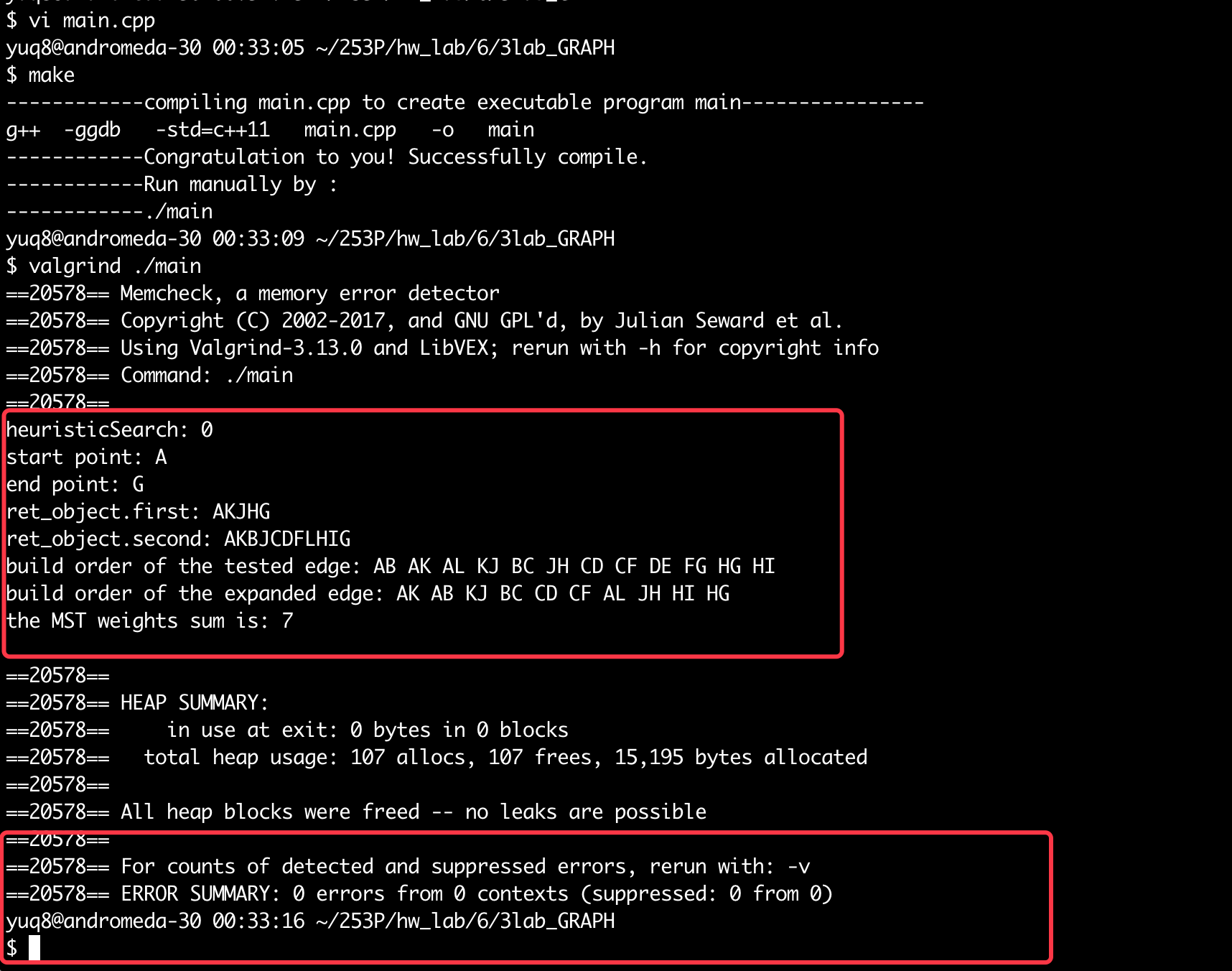
****

**test cases 2 with standard given input:**

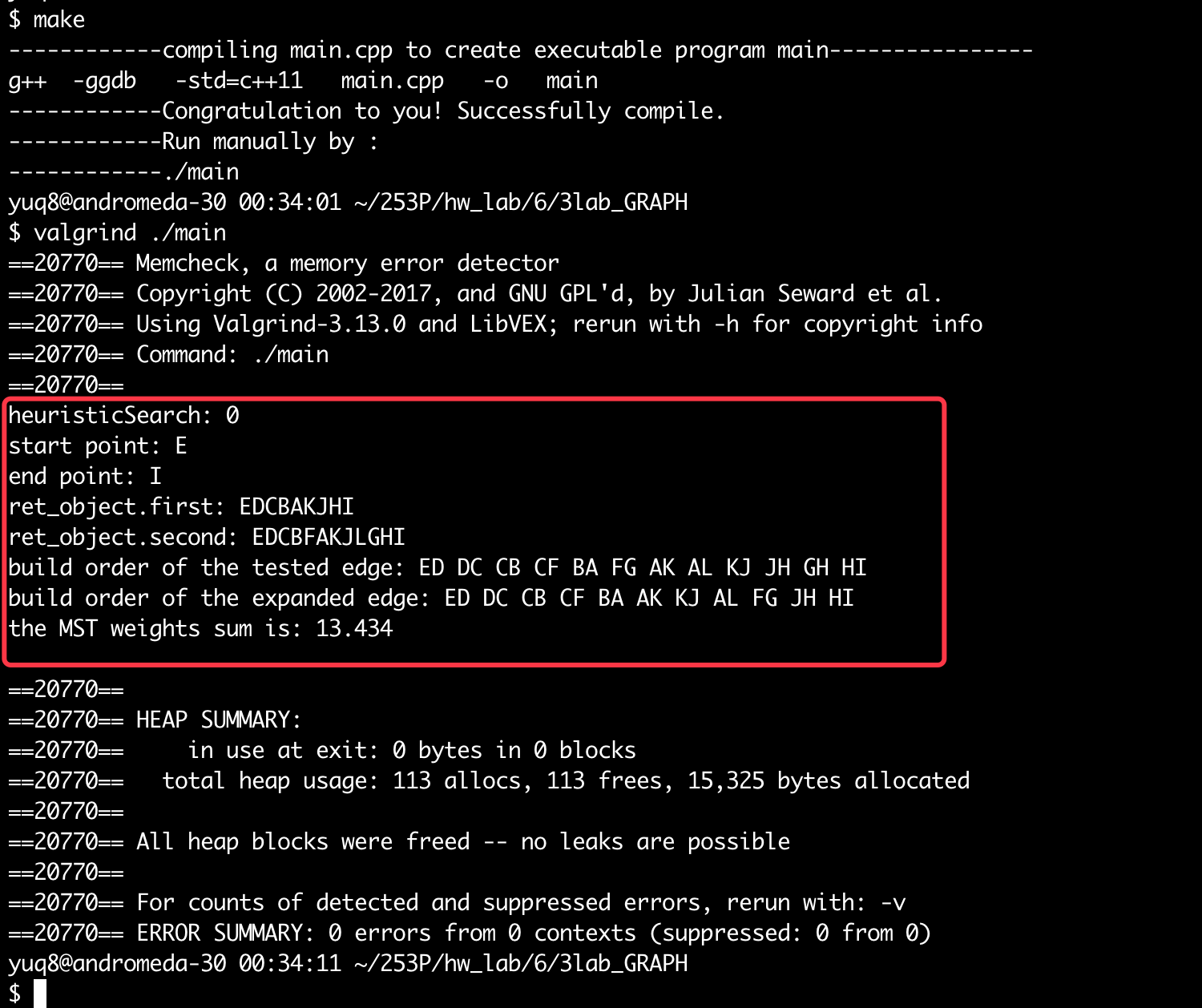
**with heuri:**

****

**without heuri:**

****

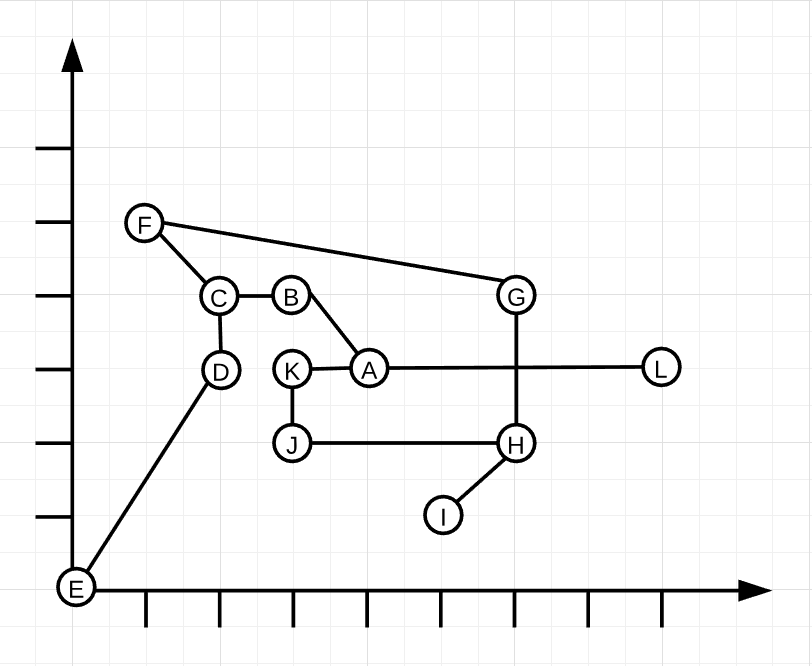
test 4:



**Part II (50pts) - [REQUIRES COMPLETION OF PART I FOR CREDIT]**

Copy the image of the graph for the standard input twice.

* for one, label the edges in order (1, 2, 3, ...) as they were established during Dijkstra's
* for the other, label the edges in order (1, 2, 3, ...) as they were established during A\*

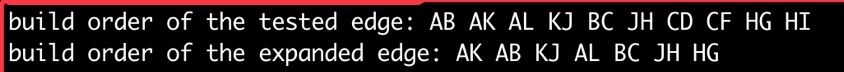


As shown in the code, the order are following. But since I’m not sure whether you mean to, I give both the refresh edge and expanded node:

For without heur from A to G, the order of 1, 2, 3… is :



For with heur from A to G:



Since it is the same order as above, so I let alone the just drawing step.

# **Lab 5.3: All about Coins——special column**

**General Problem Description**

About coins, there are many kinds of detailed categories. For example:

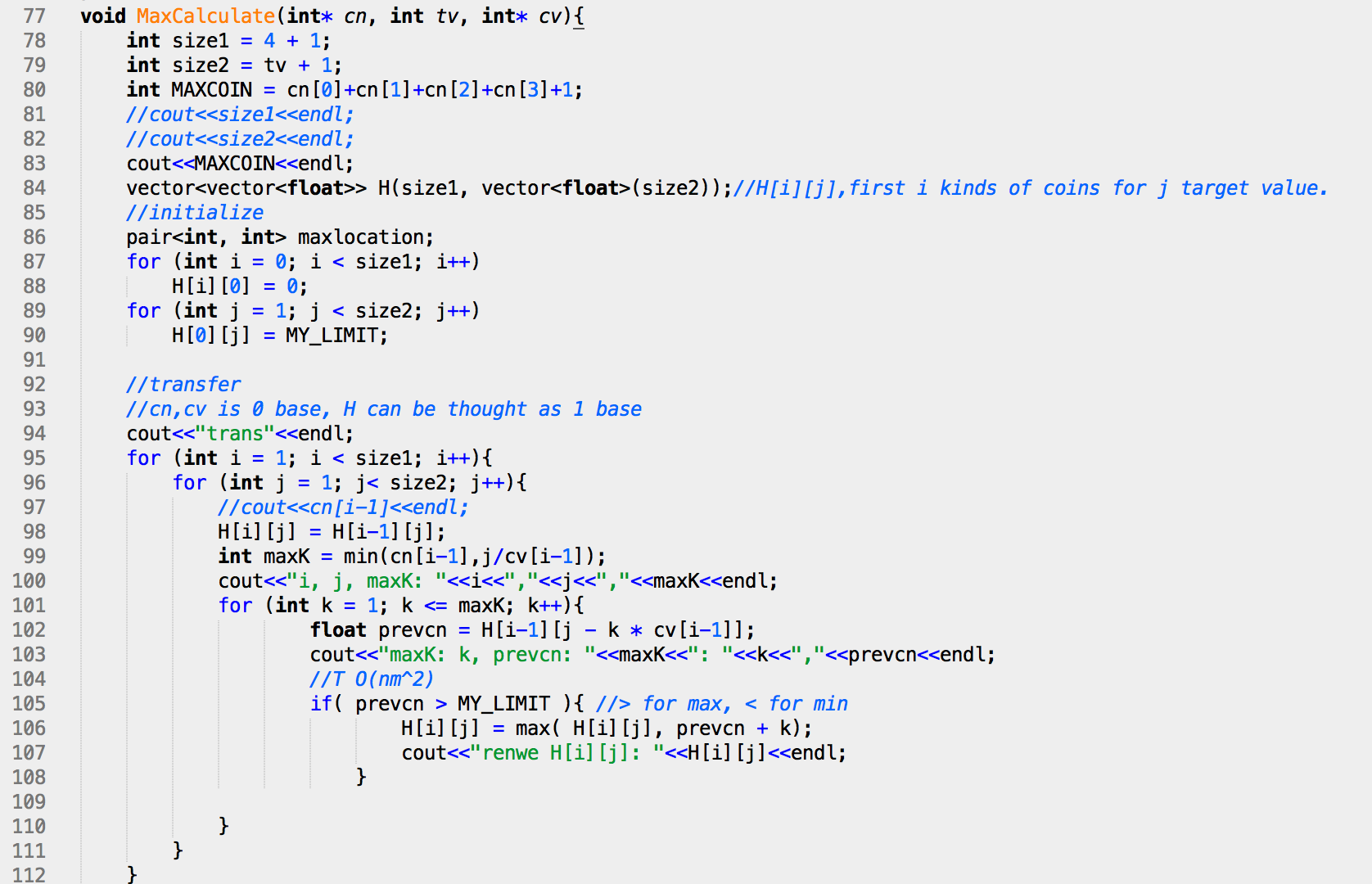
|  |  |  |
| --- | --- | --- |
|  | Unlimited coins number | Limited coins number |
| Max coins |  | Lab5.3 |
| Min coins | [1], LC 322[2] | [1] |
| Solution nums |  |  |
| Require sth. |  |  |

[1] <https://blog.csdn.net/suwei19870312/article/details/9296415>

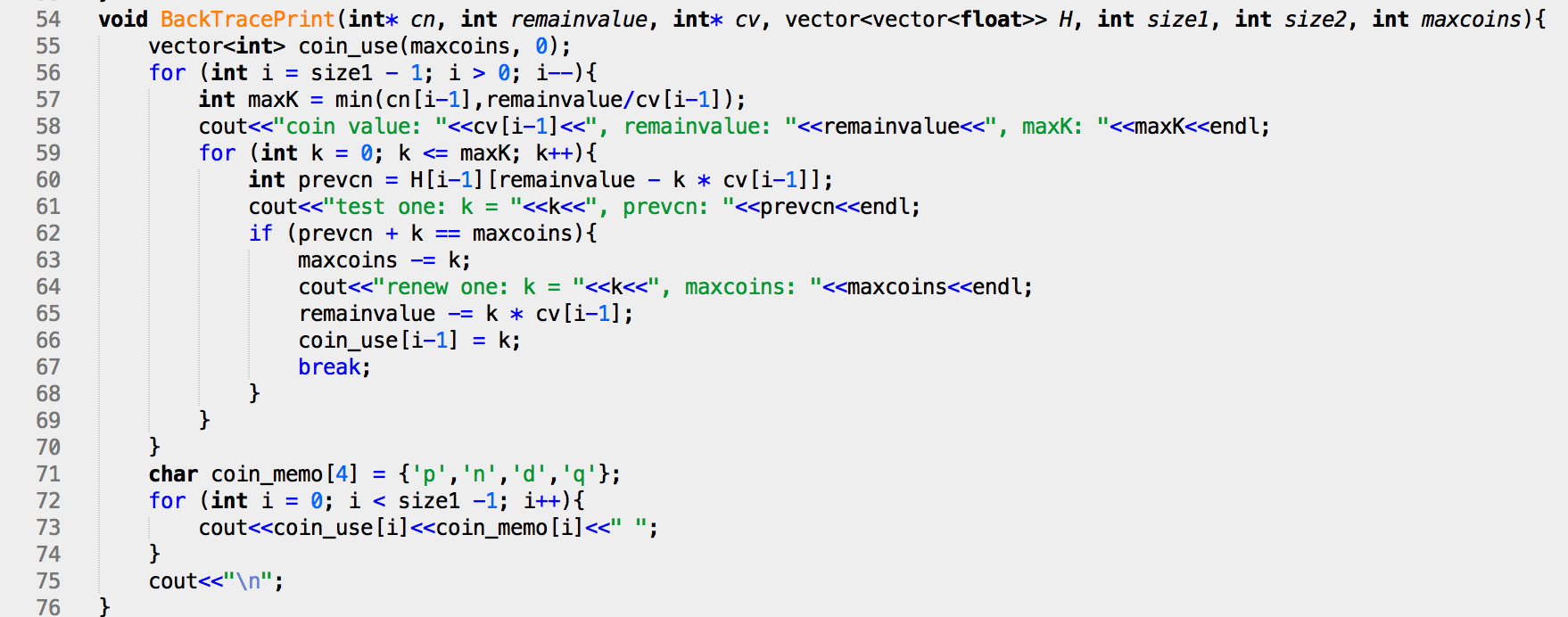
[2] <https://www.laioffer.com/zh/videos/2018-04-23-322-coin-change/>

So, In this lab, I write code mainly consisting of 2 parts：

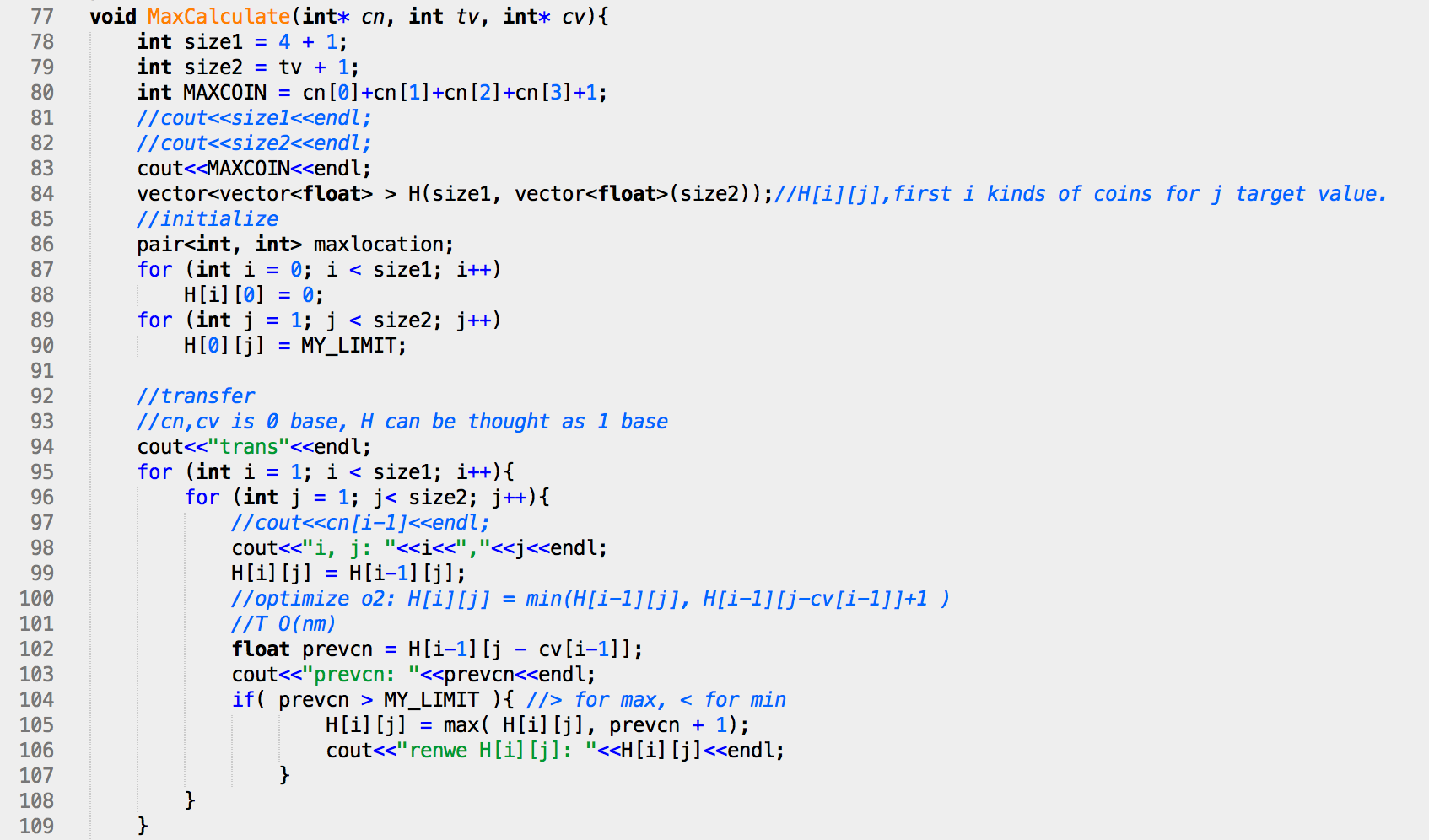
1. find the max in DP



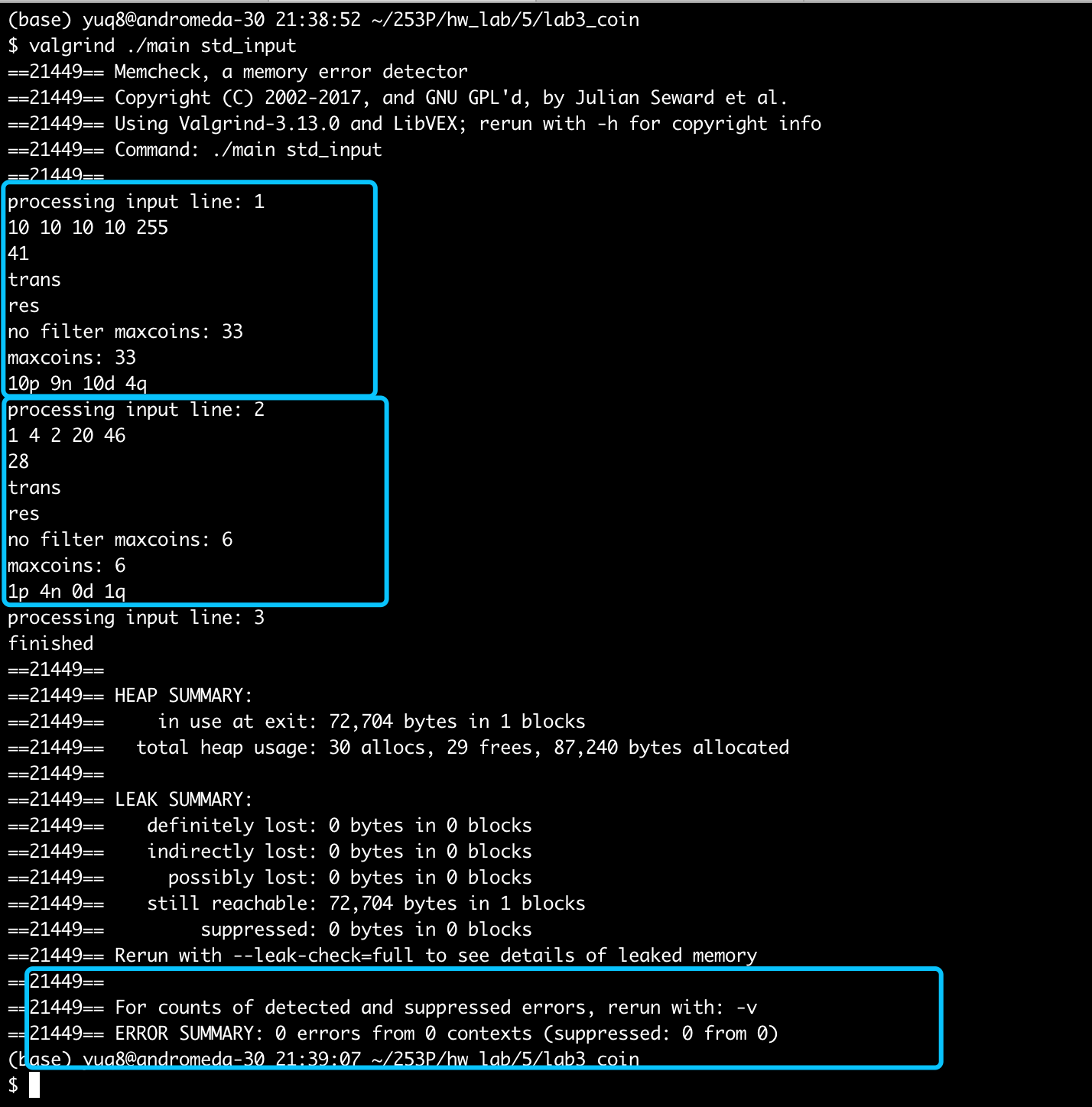
1. traceback:



and the first part can be optimized in Time to :



test case 1, 2:



What’s more, it can be optimized in space to O(m).

The result screenshot is：

And future similar codes are on going… to be continued.