云汉交龙战队 2023 赛季视觉部招新笔试题

注意事项

- 笔试共有 4 道题目，A 题为算法题，B 题为面向对象应用题，C 题为多线程应用题，D 题为 OpenCV 应用题。考虑到有些同学没有接触过多线程或 OpenCV，C、D 两题附有教程链接。
- 所有题目均可以采用 c++ 或 python 答题。默认标准为 c++17 和 python 3.9。
- 题目并无标准答案，只需在自己能力范围内尽可能多地答题即可。如无法用代码实现，在提交文件中给出解题思路或不完整的代码也可得到一定分数。
- 以下第三方库可以不加说明地使用：

第三方库名称	官网链接
Eigen	http://eigen.tuxfamily.org/index.php?title=Main_Page
OpenCV	https://opencv.org/

如使用其他第三方库，请在提交时给出相关说明和环境配置文件。祝各位码运昌盛。

A. 运动

> 本题以数据范围区分 3 个梯度，时间限制 2s，空间限制 256 MB。

题目描述

最近，小谢制作了一个新机器人模型，它甚至可以在 RoboMaster 的赛场上运动！

RoboMaster 超级对抗赛的赛场是一个 n 行 m 列的网格，定义 x 轴正方向为下，y 轴正方向为右，则第 x 行第 y 列坐标为 (x, y)，左上角坐标为 $(1, 1)$。赛场上一共有 k 个障碍物。第 i 个障碍物的坐标是 (x_i, y_i)。机器人不能走进有障碍物的网格，也不能越出赛场边界。

小谢的机器人在网格中既可以直行 1 格，也可以先右转 90 度再直行 1 格，在同一网格中最多右转一次。他的机器人有上、下、左、右四个朝向。起初，小谢的机器人位于 $(1, 1)$，朝向为右。他想让机器人恰好走过所有没有障碍物的网格一次。他是否可以实现他的想法？

输入格式

第一行包含 3 个整数 n, m, k，表示超级对抗赛赛场网格的行数和列数，以及障碍物的数量。

接下来的 k 行，每行两个整数 x_i, y_i 表示一个障碍物的坐标。

数据保证同一网格中最多只有一个障碍物，且 $(1, 1)$ 无障碍物。

输出格式

如果小谢能够让机器人恰好经过每个无障碍网格一次，输出 "Yes"（不含引号），否则输出 "No"。

样例

输入 #1

```
3 3 2
2 2
2 1
```

输出 #1

```
Yes
```

输入 #2

```
3 3 2
2 2
3 1
```

输出 #2

```
No
```

数据范围

梯度 1：$n, m, k \leq 10$

梯度 2：$n, m, k \leq 10^5$

梯度 3：$n, m \leq 10^9, k \leq 10^5$

B. 对象

> 在本题展现你对**面向对象**的理解。如果你不知道面向对象，你也可以用习惯的方法实现尽可能多的功能，本题以实现的操作种类数区分 3 个梯度。

题目描述

小谢正在用 X 排版软件为云汉交龙战队做招新推文。然而，X 软件各个组件之间的关系非常混乱，并且还充满了 bug。小谢决定让你为他开发一个更合理的排版框架。

开始时，排版界面没有任何组件。小谢会输入一些操作指令来添加或改动组件。你需要用字符输出结果，下方有结果示例。你只需实现基本功能，但不用太在意结果的美观程度。

操作指令不保证合法，非法情况按照你觉得最合适、最人性化的方式处理。

组件有 3 种类型：方角组件、圆角组件、花角组件。下方是它们的字符组成示例以及类型代号。

- 方角组件 `r`：

```
+---+
|   |
+---+
```

- 圆角组件 `c`：

```
o---o
|   |
o---o
```

- 花角组件 `l`：

```
*---*
|   |
*---*
```

每个组件有一个编号，编号需要展示在组件内部。

在同一层级的若干组件**按照出现在该层级的先后顺序由上到下**展示。组件长宽、空行方式和编号展示位置都由你决定。

- 创建指令

  `create x y -t`

  在编号为 `y` 的组件中创建编号为 `x`，类型为 `t` 的组件。排版界面最外层的编号为 0。

  输入 #1

```
create 1 0 -r
create 2 0 -l
create 3 1 -c
```

  输出 #1

```
+-------+
|   1   |
| o---o |
| | 3 | |
| o---o |
|       |
```

```
+--------+

*---*
| 2 |
*---*
```

- 删除指令

  `del x`

  删除 `x` 以及其所属组件。

  输入 #2

  ```
  create 1 0 -r
  create 2 0 -l
  create 3 1 -c
  del 1
  create 1 0 -l
  ```

  输出 #2

  ```
  *---*
  | 2 |
  *---*

  *---*
  | 1 |
  *---*
  ```

- 移动指令

  `mv x y`

  移动编号为 `x` 的组件到编号为 `y` 的组件中。

  输入 #3

  ```
  create 1 0 -r
  create 2 0 -l
  create 3 1 -c
  mv 2 1
  ```

  输出 #3

  ```

  ```

```
+-------+
|   1   |
| o---o |
| | 3 | |
| o---o |
|       |
| *---* |
| | 2 | |
| *---* |
|       |
+-------+
```

- 改变类型指令

  `set x -t`

  将编号为 `x` 的组件类型修改为 `t`。

  输入 #4

  ```
  create 1 0 -r
  set 1 -l
  ```

  输出 #4

  ```
  *---*
  | 1 |
  *---*
  ```

- 撤销指令

  `undo`

  撤回前一步操作。

  输入 #5

  ```
  create 1 0 -r
  create 2 0 -l
  create 3 1 -c
  undo
  ```

  输出 #5

  ```

  ```

```
+---+
| 1 |
+---+

*---*
| 2 |
*---*
```

- 重做指令

  `redo`

  重做最近 `undo` 的指令。

  输入 #6

  ```
  create 1 0 -r
  create 2 0 -l
  create 3 1 -c
  undo
  redo
  ```

  输出 #6

  ```
  +-------+
  |   1   |
  | o---o |
  | | 3 | |
  | o---o |
  |       |
  +-------+

  *---*
  | 2 |
  *---*
  ```

数据范围

梯度 1：实现 `create` 指令。

梯度 2：实现 `create`，`del`，`mv`，`set` 指令。

梯度 3：实现所有 6 个指令。

你提交的代码会经过人工测试。本题将根据代码的完成情况、运行效率和可拓展性进行评分。

C. 自瞄

题目描述

小谢通过学习 Harry-hhj 的视觉开源代码，在团队中协作完成了识别装甲板神经网络。图像通过这一神经网络后，就会给出一个 0 到 99 之间的整数坐标结果，接下来便可以对它进行运动建模。

为了分离 CPU 和 GPU 任务，提高代码并行度，小谢将**神经网络识别器**和**运动预测器**分为两个线程独立运行。然而由于相机帧率抖动等原因，识别器和预测器两者的同步性需要保证。简单来说，就是识别器将识别结果**发布**到一个队列中，而预测器会**订阅**这个队列。小谢想要你帮忙完成发布订阅框架，要求：当队列中没有内容时，订阅者休眠，发布者唤醒并发布识别结果到队列中；队列中有内容时，发布者休眠，订阅者唤醒并弹出和输出识别结果。

小谢提供的文件夹如下：

```
c
├── c.cpp
└── c.py
```

你需要选择其中一个源文件，可以在其中看到未完成的发布者和订阅者的简单框架。然后你需要在注释 `START CODE HERE` 和 `END CODE HERE` 之间填写代码，最终实现发布者和订阅者的互动，并依次输出发布者的所有结果。

提示

如果你没有接触过多线程，可以通过以下教程学习：

c++: https://sjtu-robomaster-team.github.io/c++-thread/

python: https://www.runoob.com/python3/python3-multithreading.html

D. 陀螺

> 本题以实现的功能区分 3 个梯度。

题目描述

随着小谢的又一版自瞄横空出世，RoboMaster 的赛场上也出现了“反自瞄”，其中的代表之一就是“小陀螺”。机器人在小陀螺模式下，云台和底盘处于分离状态，在底盘绕运动中心高速旋转的同时云台保持稳定不动。小谢邀请你来帮他识别出旋转中的装甲板，并计算陀螺旋转的角速度，方便之后进行建立新的预测手段。

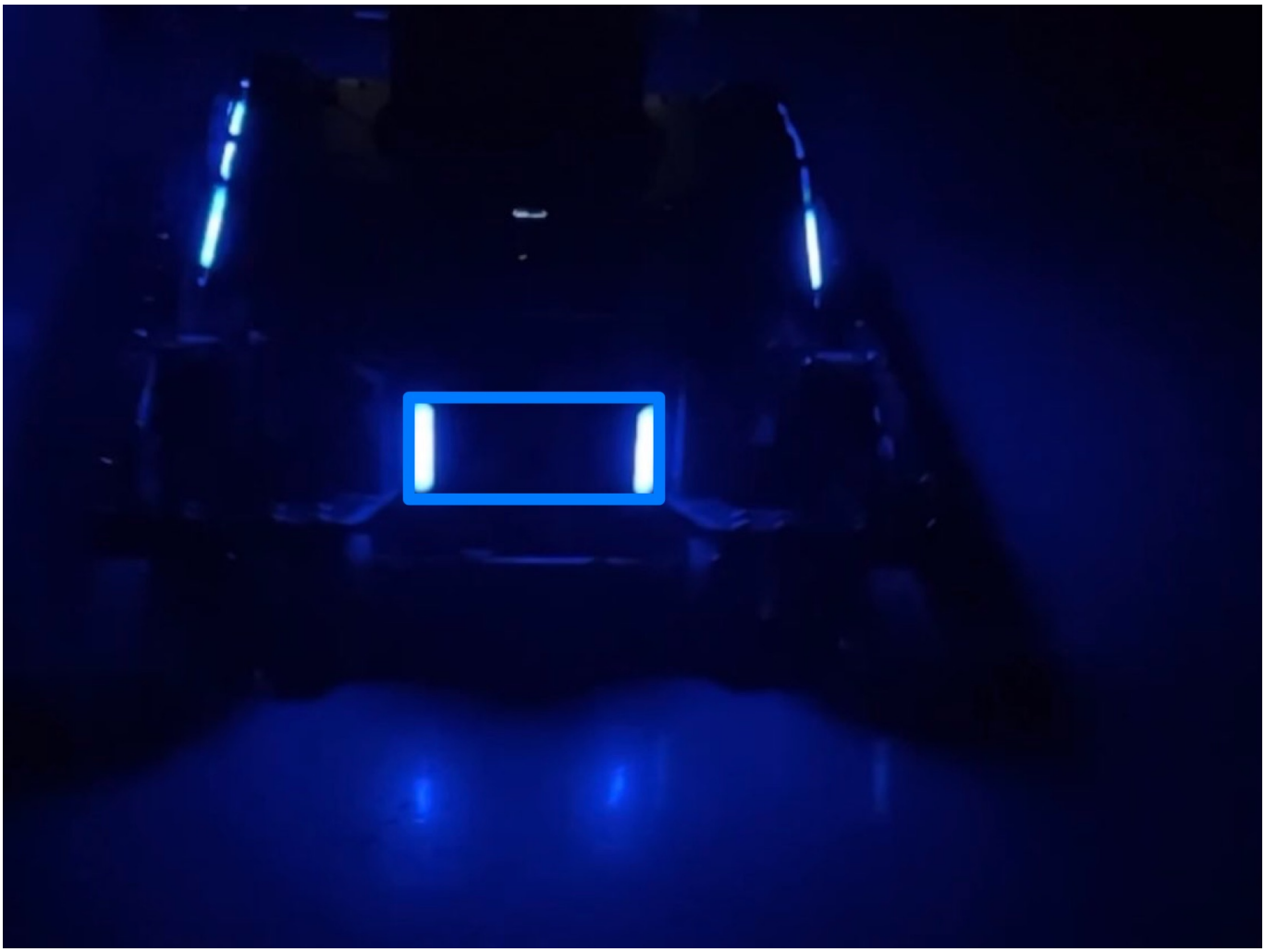
梯度

小谢提供的文件夹如下：

```
d
├── install_opencv.pdf  -> c++ opencv 安装教程
├── pineapple.jpg       -> 小菠萝机器人在有光环境下的照片，用于参考
├── d.jpg               -> 待识别的静止机器人图片，曝光极低
└── d.mp4               -> 待识别的陀螺机器人视频
```

梯度 1：在 `d.jpg` 中用外接矩形框标出中间装甲板的位置。

一种理想效果如下：



梯度 2：在 `d.mp4` 的每一帧用外界矩形框标出所有可见装甲板的位置，注意每帧中最多有 2 个可见装甲板。

梯度 3：设法计算 `d.mp4` 中机器人陀螺运动的角速度。注意，该视频帧率为 60 fps。

提示

本题需要用 opencv 库实现。

c++ opencv 安装方法：见文件夹中 `install_opencv.pdf`。建议使用 ubuntu 系统，如果系统不是 ubuntu，你也可以百度查找 opencv 安装方法。

基础操作教程：可见 https://sjtu-robomaster-team.github.io/vision-learning-1/。

python opencv 安装方法：在终端输入 `pip3 install opencv-python`。

基础操作教程：可参考 https://blog.csdn.net/bugang4663/article/details/111413850