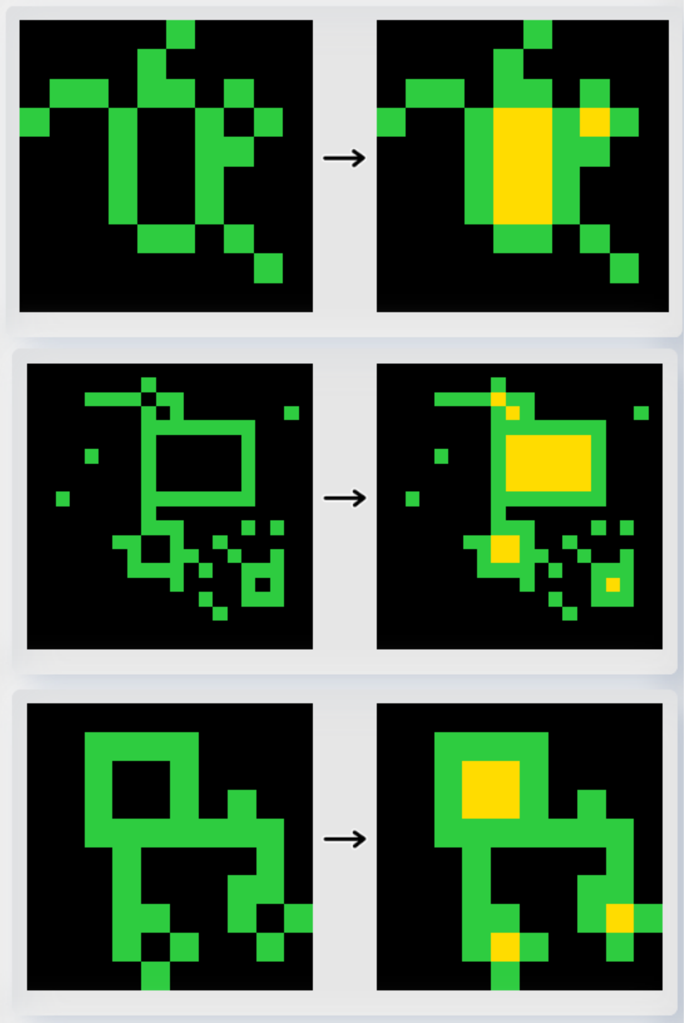
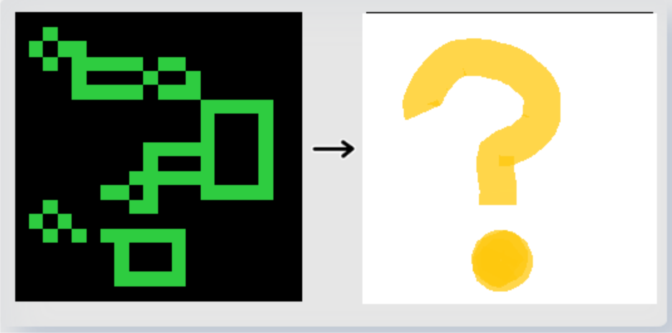
ANPL ARC

23/05/07

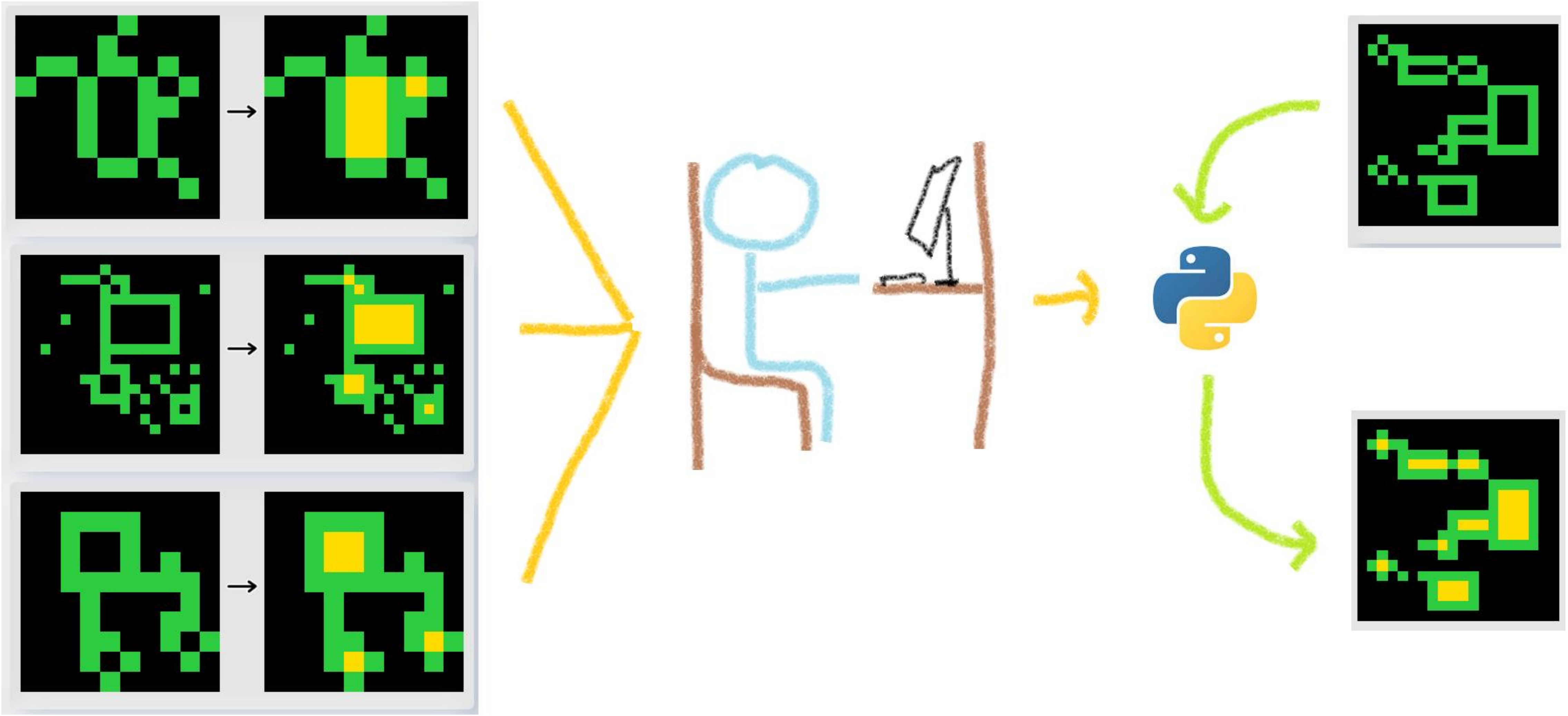
Abstraction and Reasoning Challenge



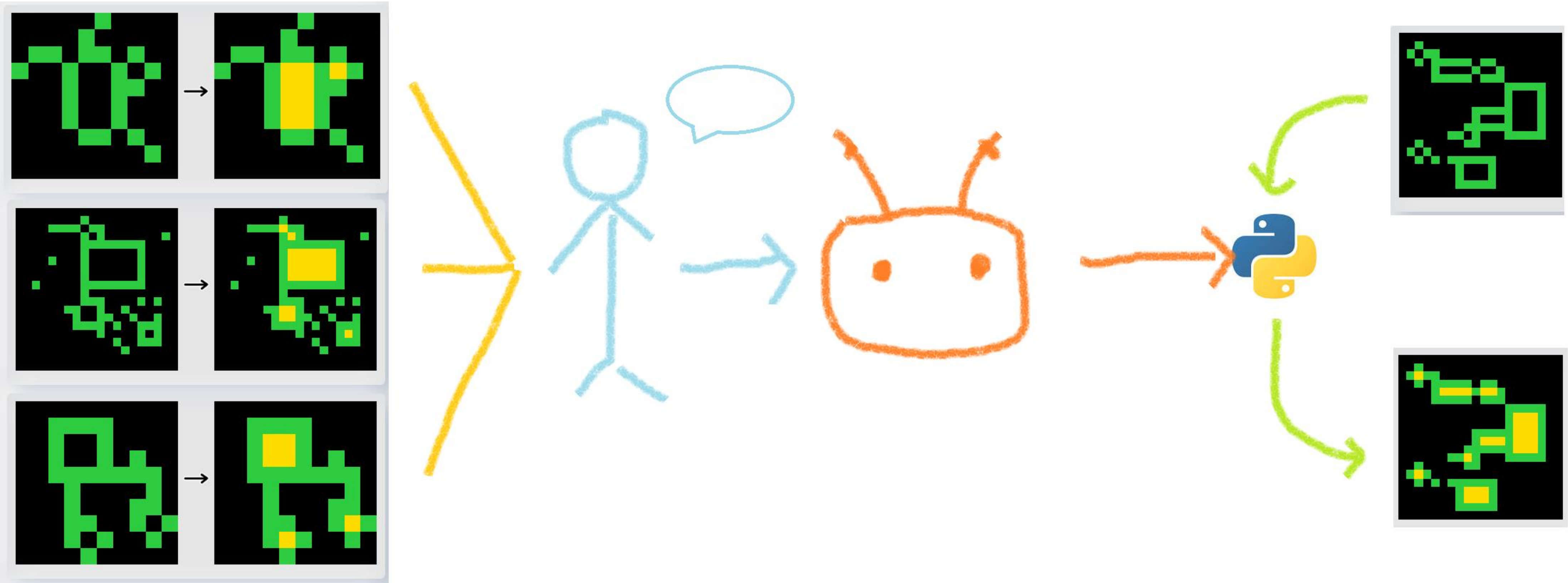
1/400



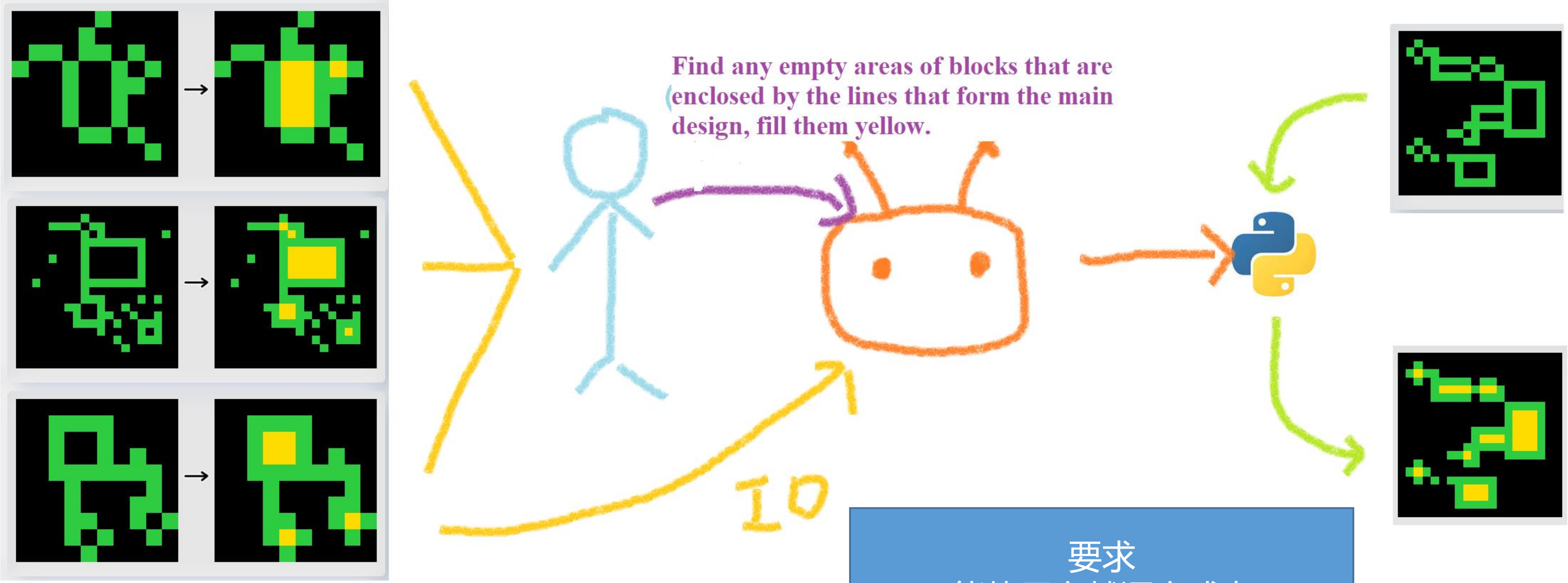
怎么解决ARC问题呢？



让AI帮助我们写代码！



Robot B: ChatGPT with nl and IO

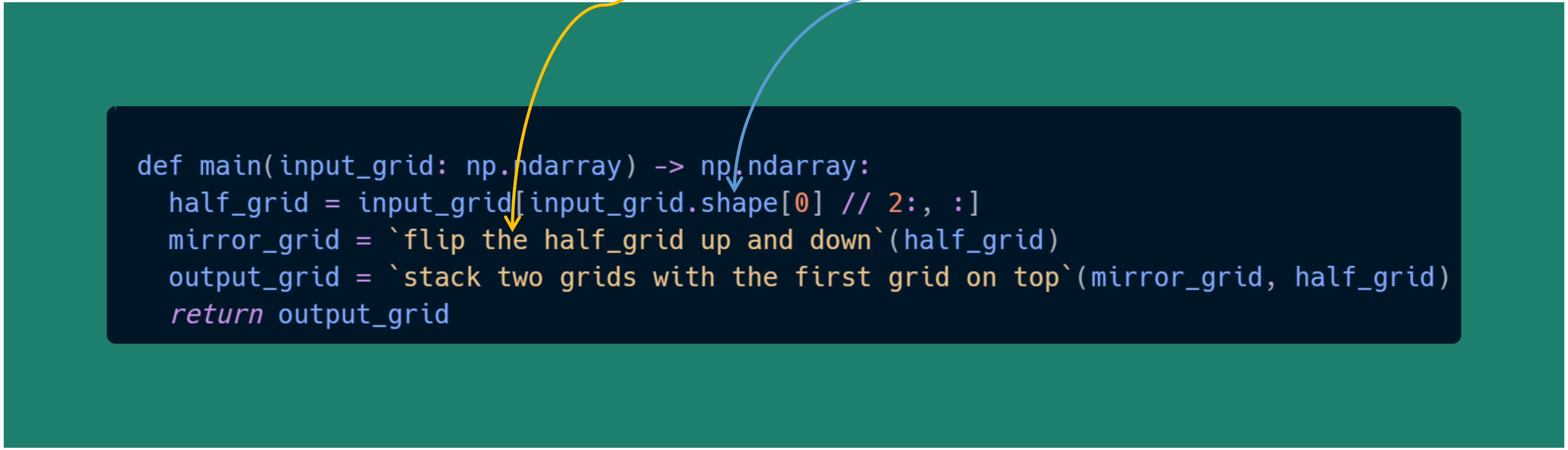


Find any empty areas of blocks that are enclosed by the lines that form the main design, fill them yellow.

要求
只能使用自然语言或者IO
不能直接给他正确的python代码

Robot A: A Natural Programming Language

- Pseudocode: Program with holes | Superset of Python
- hole 用两个反引号括住一段自然语言，robot会根据这段描述生成代码



```
def main(input_grid: np.ndarray) -> np.ndarray:
  half_grid = input_grid[input_grid.shape[0] // 2:, :]
  mirror_grid = `flip the half_grid up and down`(half_grid)
  output_grid = `stack two grids with the first grid on top`(mirror_grid, half_grid)
  return output_grid
```

Limitations

- 可以使用Python自带的控制流，if while for ...
- 只使用函数，不要使用类等功能
- 不要import其他库

- 非常需要定义数据结构怎么办？namedtuple
  - Shape = namedtuple("Shape", ["size", "color", "position", "clip_grid"])

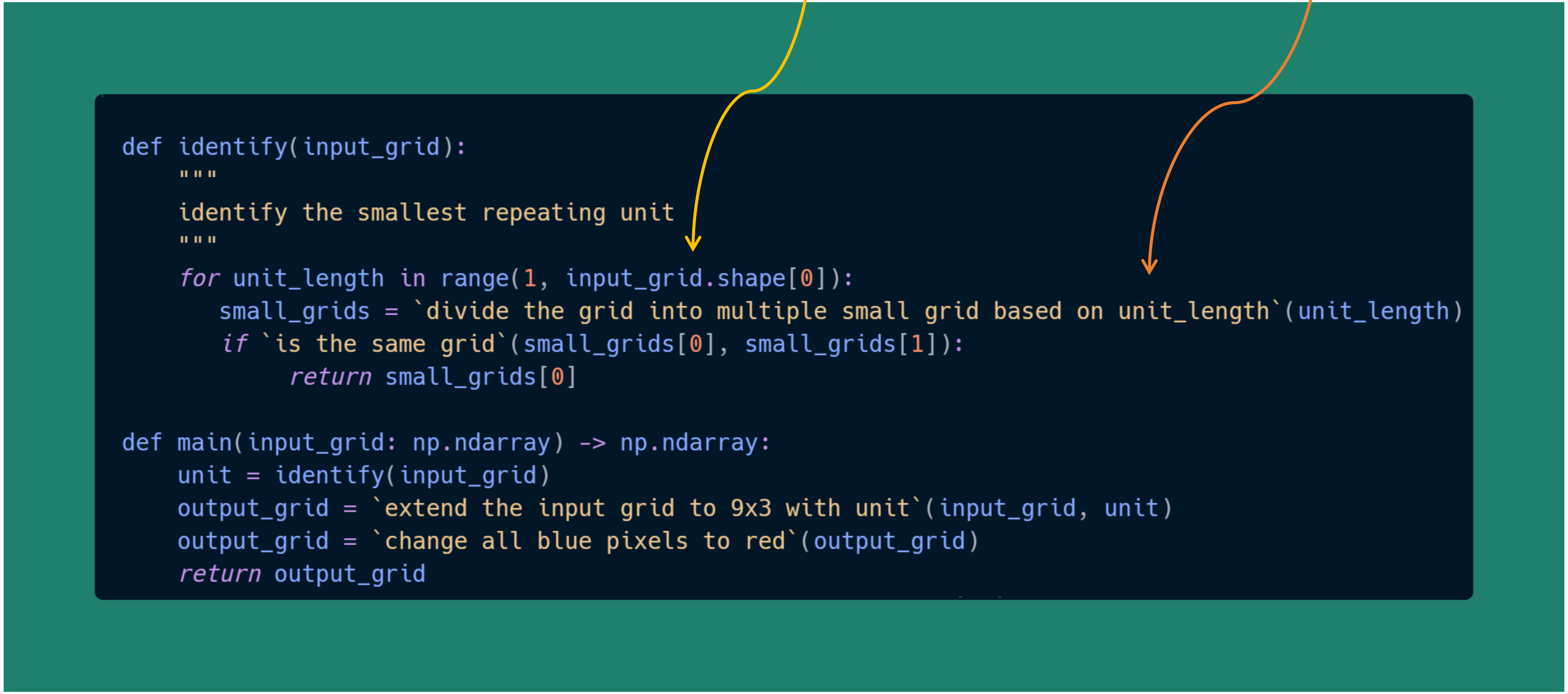
- 函数尽量是纯函数
  - 同一个输入，同一个输出

- 类似于算法题

Good Examples

合理使用python的控制流

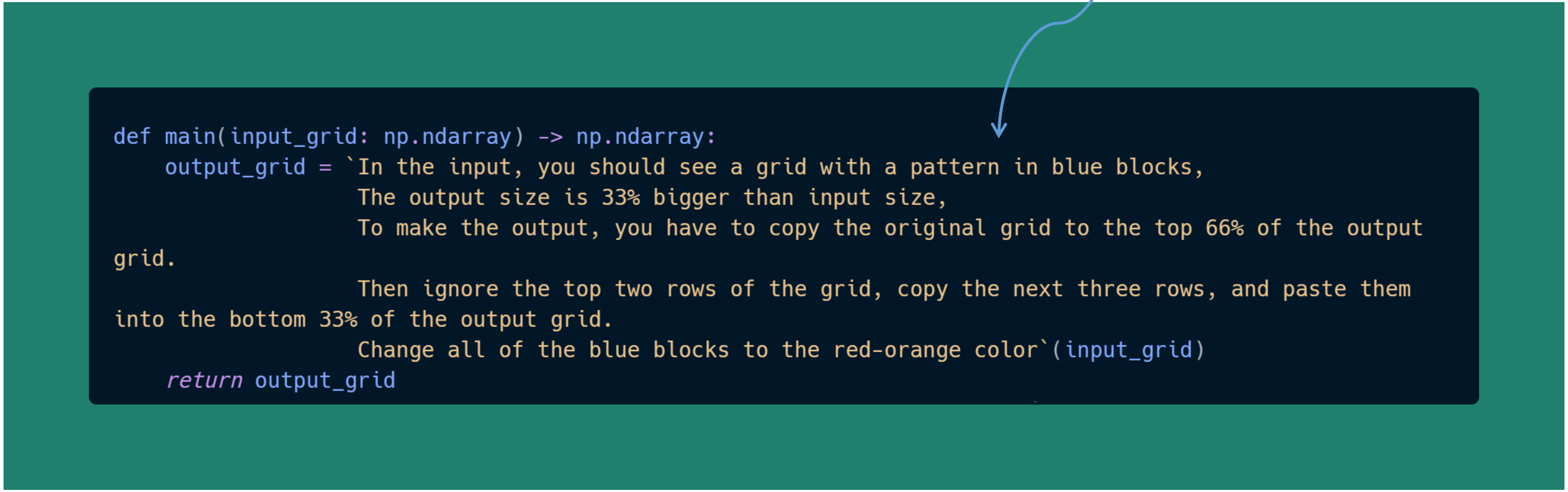
简单明确的描述



```
def identify(input_grid):
    """
    identify the smallest repeating unit
    """
    for unit_length in range(1, input_grid.shape[0]):
       small_grids = `divide the grid into multiple small grid based on unit_length`(unit_length)
       if `is the same grid`(small_grids[0], small_grids[1]):
            return small_grids[0]

def main(input_grid: np.ndarray) -> np.ndarray:
    unit = identify(input_grid)
    output_grid = `extend the input grid to 9x3 with unit`(input_grid, unit)
    output_grid = `change all blue pixels to red`(output_grid)
    return output_grid
```

Bad Examples

一长串话



```
def main(input_grid: np.ndarray) -> np.ndarray:
    output_grid = `In the input, you should see a grid with a pattern in blue blocks,
                   The output size is 33% bigger than input size,
                   To make the output, you have to copy the original grid to the top 66% of the output
grid.
                   Then ignore the top two rows of the grid, copy the next three rows, and paste them
into the bottom 33% of the output grid.
                   Change all of the blue blocks to the red-orange color`(input_grid)
    return output_grid
```

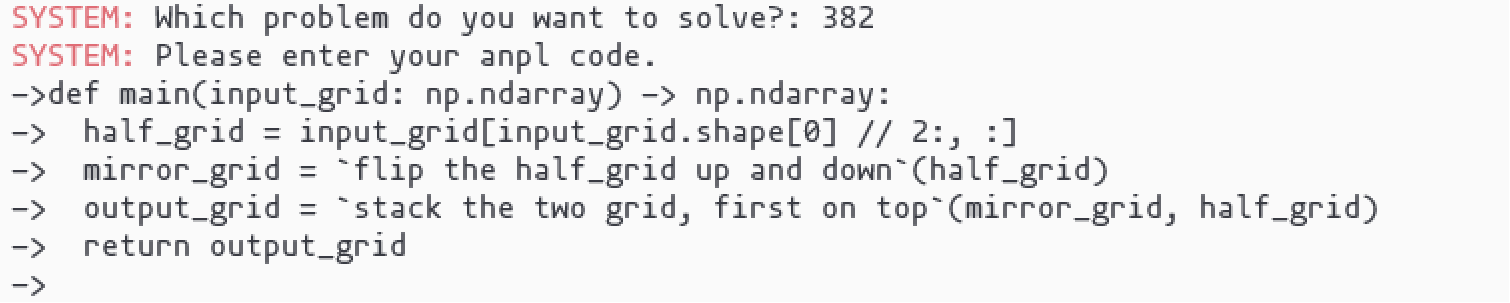
我们希望测试者是在用自然语言编程，
而不是用自然语言描述

一些你需要知道的事情~

- arc中一共有10种颜色，分别是
  - black, blue, red, green, yellow, grey, pink, orange, teal(淡蓝), maroon(暗红)
  - 10种颜色已经定义在了环境中，可以直接使用
- 环境中已经import numpy as np & from numpy import array
  - 所以可以直接调用np的函数
  - 输入grid时可以直接写 array([[1, 2, 3], [2, 3, 4]]) ...
- 合理的类型注释可以减少类型错误 - help ChatGPT

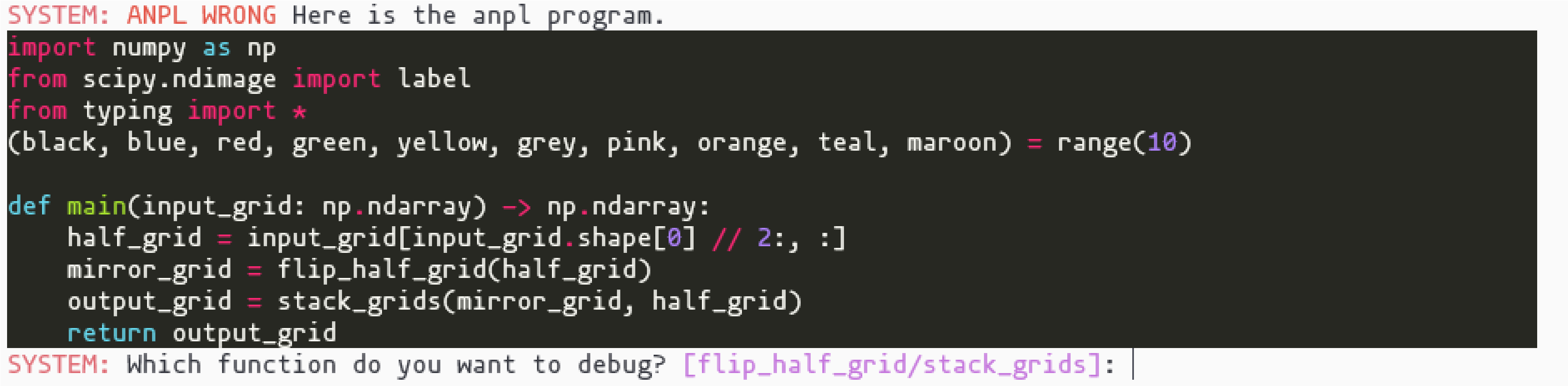
Workflow Start

- 选择任务并输入代码



```
SYSTEM: Which problem do you want to solve?: 382
SYSTEM: Please enter your anpl code.
->def main(input_grid: np.ndarray) -> np.ndarray:
->    half_grid = input_grid[input_grid.shape[0] // 2:, :]
->    mirror_grid = `flip the half_grid up and down`(half_grid)
->    output_grid = `stack the two grid, first on top`(mirror_grid, half_grid)
->    return output_grid
->
```

- 系统会自动生成并检查代码正确性, 如果不正确，进入debug

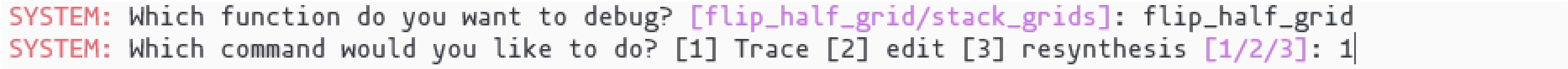


```
SYSTEM: ANPL WRONG Here is the anpl program.
import numpy as np
from scipy.ndimage import label
from typing import *
(black, blue, red, green, yellow, grey, pink, orange, teal, maroon) = range(10)

def main(input_grid: np.ndarray) -> np.ndarray:
    half_grid = input_grid[input_grid.shape[0] // 2:, :]
    mirror_grid = flip_half_grid(half_grid)
    output_grid = stack_grids(mirror_grid, half_grid)
    return output_grid
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]:
```

Workflow Debug-loop

- Debug是一个循环，反复执行debug命令。每次命令结束后会自动检查程序的正确性。如果正确，自动退出，否则再次进入循环。

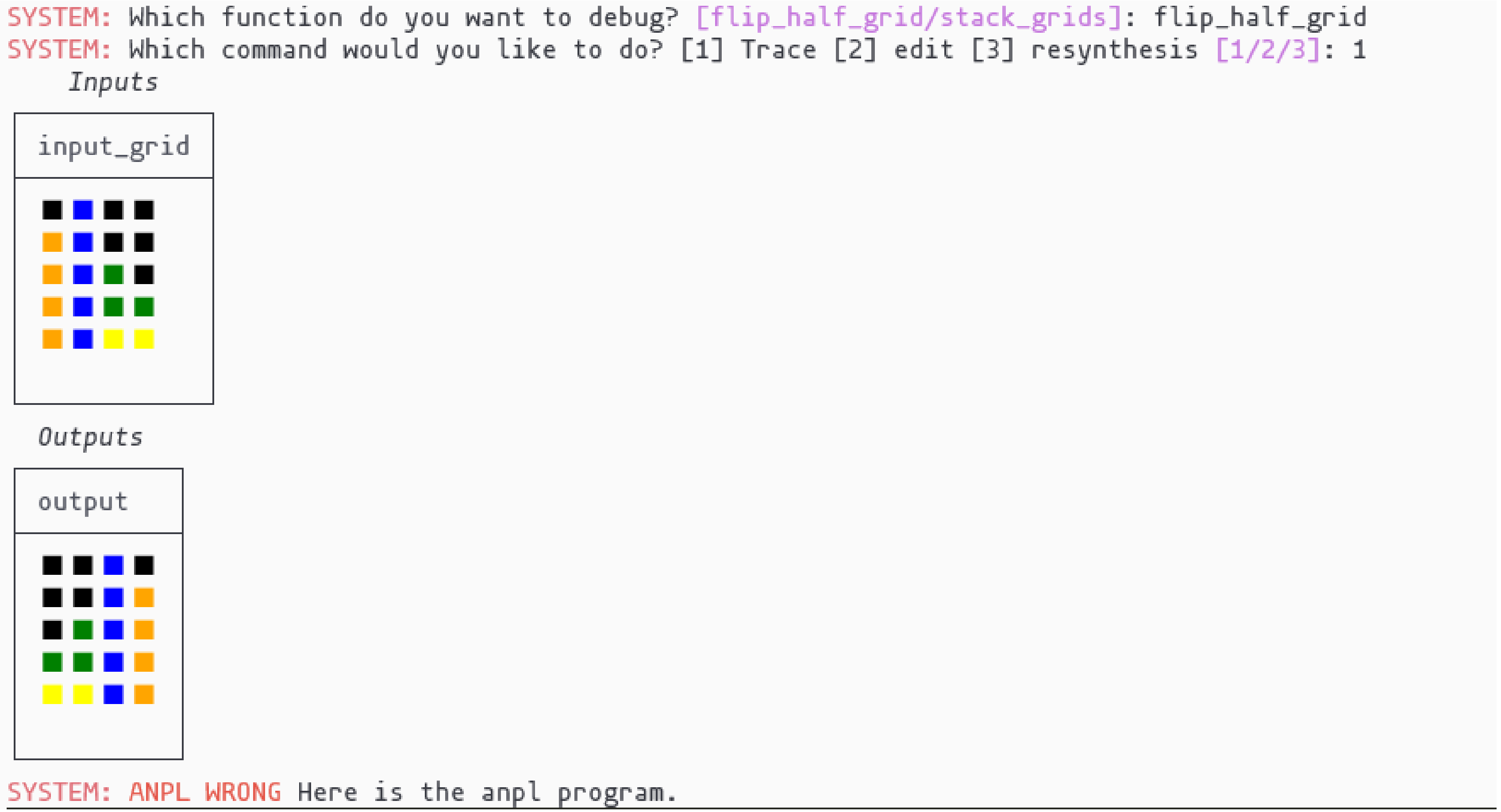
- Debug需要选择 命令 以及 函数名



```
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]: flip_half_grid
SYSTEM: Which command would you like to do? [1] Trace [2] edit [3] resynthesis [1/2/3]: 1
```

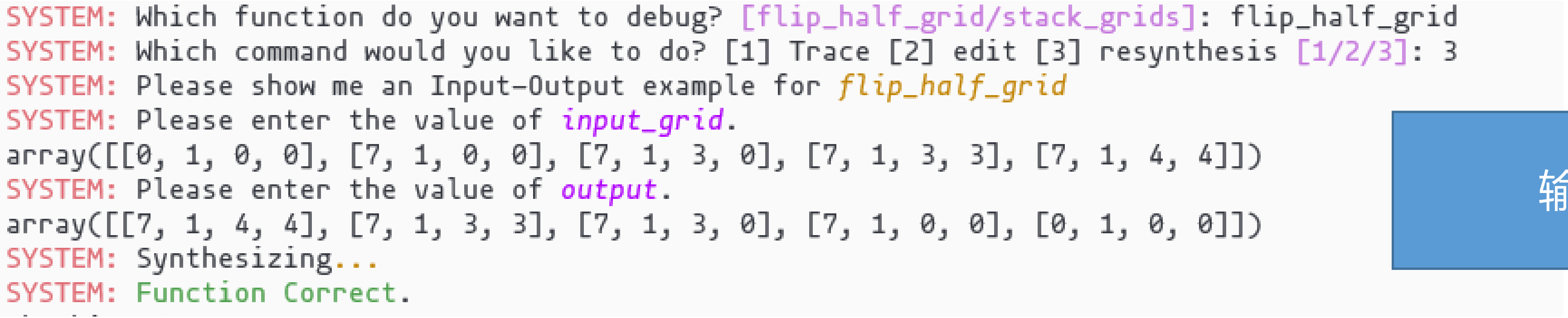
- 三种命令分别是 trace edit resynthesis
  - trace 查看指定函数的IO
  - edit 直接编辑指定函数的代码，继续编写anpl
  - resynthesis 给一组IO，命令系统生成多个函数，并测试是否满足此IO

Trace



```
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]: flip_half_grid
SYSTEM: Which command would you like to do? [1] Trace [2] edit [3] resynthesis [1/2/3]: 1
    Inputs

 input_grid

 ⬛🟦⬛⬛
 🟧🟦⬛⬛
 🟧🟦🟩⬛
 🟧🟦🟩🟩
 🟧🟦🟨🟨

  Outputs

 output

 ⬛⬛🟦⬛
 ⬛⬛🟦🟧
 ⬛🟩🟦🟧
 🟩🟩🟦🟧
 🟨🟨🟦🟧

SYSTEM: ANPL WRONG Here is the anpl program.
```

resynthesis



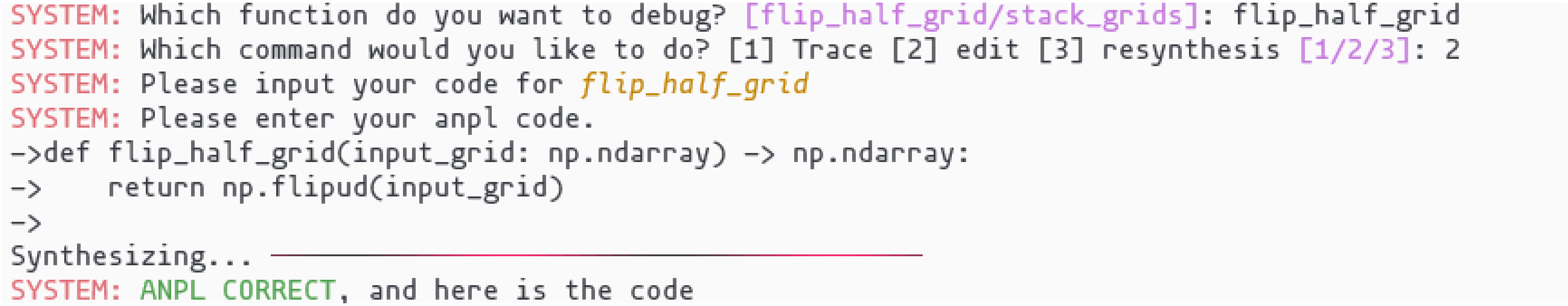
```
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]: flip_half_grid
SYSTEM: Which command would you like to do? [1] Trace [2] edit [3] resynthesis [1/2/3]: 3
SYSTEM: Please show me an Input-Output example for flip_half_grid
SYSTEM: Please enter the value of input_grid.
array([[0, 1, 0, 0], [7, 1, 0, 0], [7, 1, 3, 0], [7, 1, 3, 3], [7, 1, 4, 4]])
SYSTEM: Please enter the value of output.
array([[7, 1, 4, 4], [7, 1, 3, 3], [7, 1, 3, 0], [7, 1, 0, 0], [0, 1, 0, 0]])
SYSTEM: Synthesizing...
SYSTEM: Function Correct.
```

输入IO

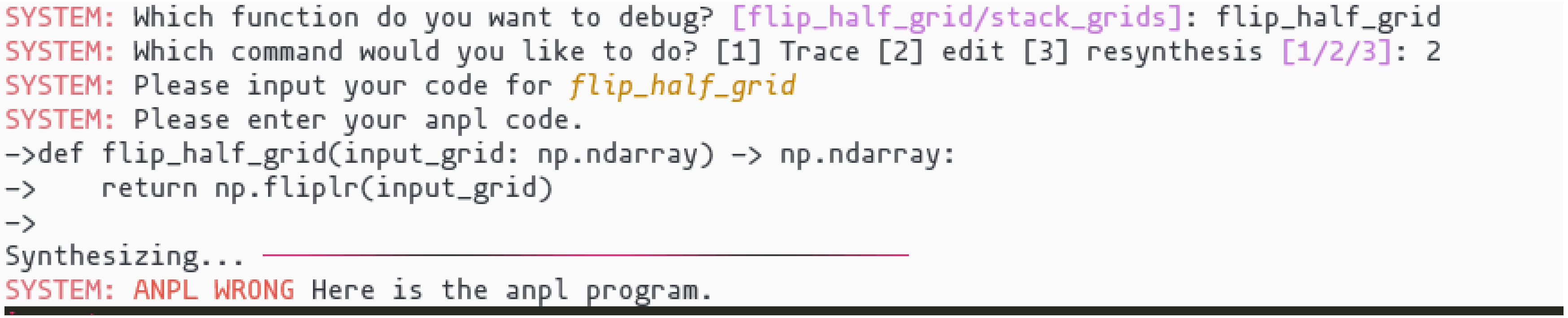
提示结果，如果无法生成正确结果，则退出循环

edit

输入ANPL Code

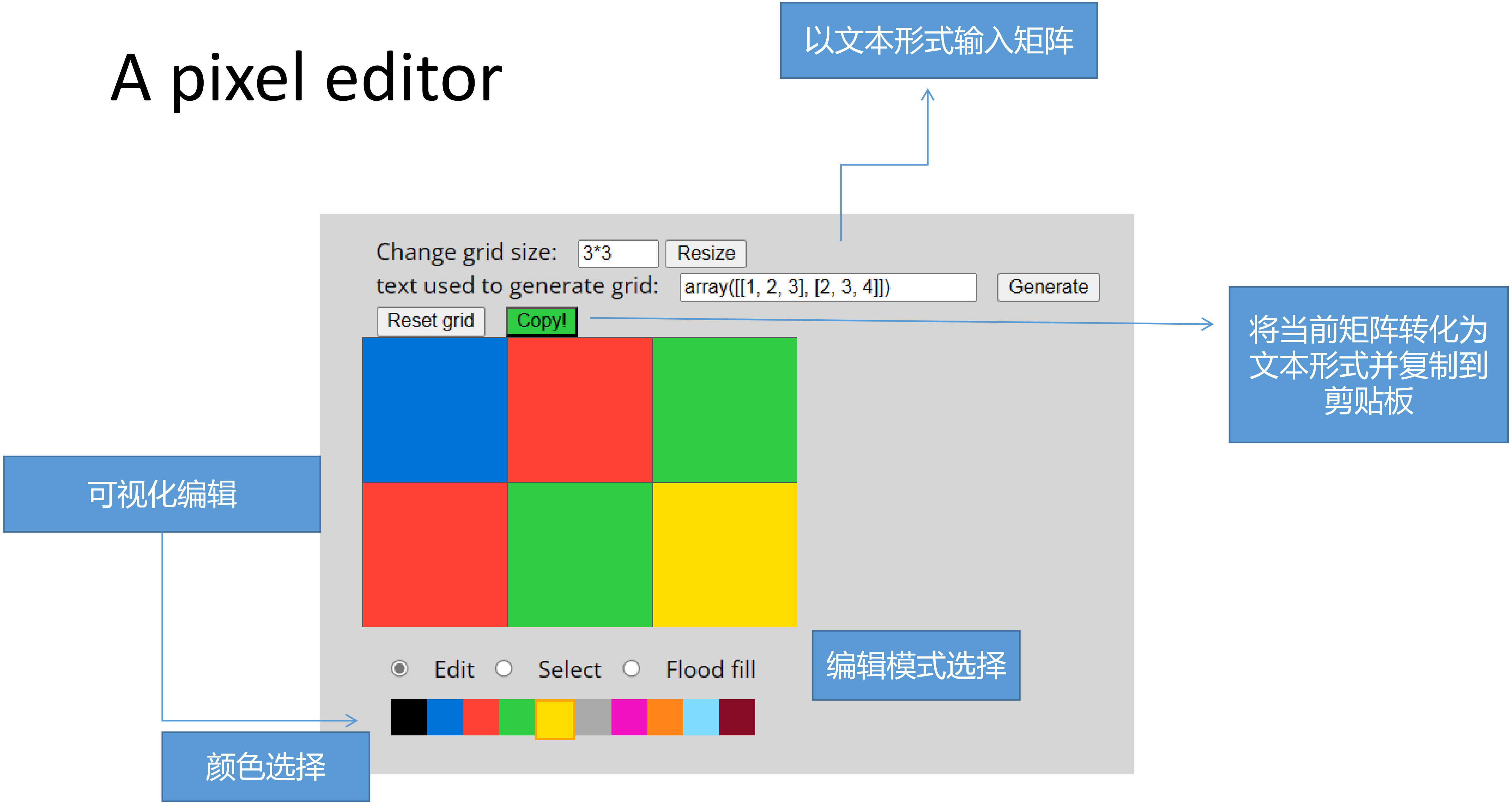


```
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]: flip_half_grid
SYSTEM: Which command would you like to do? [1] Trace [2] edit [3] resynthesis [1/2/3]: 2
SYSTEM: Please input your code for flip_half_grid
SYSTEM: Please enter your anpl code.
->def flip_half_grid(input_grid: np.ndarray) -> np.ndarray:
->    return np.flipud(input_grid)
->
Synthesizing...
SYSTEM: ANPL CORRECT, and here is the code
```



```
SYSTEM: Which function do you want to debug? [flip_half_grid/stack_grids]: flip_half_grid
SYSTEM: Which command would you like to do? [1] Trace [2] edit [3] resynthesis [1/2/3]: 2
SYSTEM: Please input your code for flip_half_grid
SYSTEM: Please enter your anpl code.
->def flip_half_grid(input_grid: np.ndarray) -> np.ndarray:
->    return np.fliplr(input_grid)
->
Synthesizing...
SYSTEM: ANPL WRONG Here is the anpl program.
```

A pixel editor



Change grid size: 3*3 Resize

text used to generate grid: array([[1, 2, 3], [2, 3, 4]]) Generate

Reset grid Copy!

● Edit ○ Select ○ Flood fill

以文本形式输入矩阵

将当前矩阵转化为文本形式并复制到剪贴板

可视化编辑

编辑模式选择

颜色选择

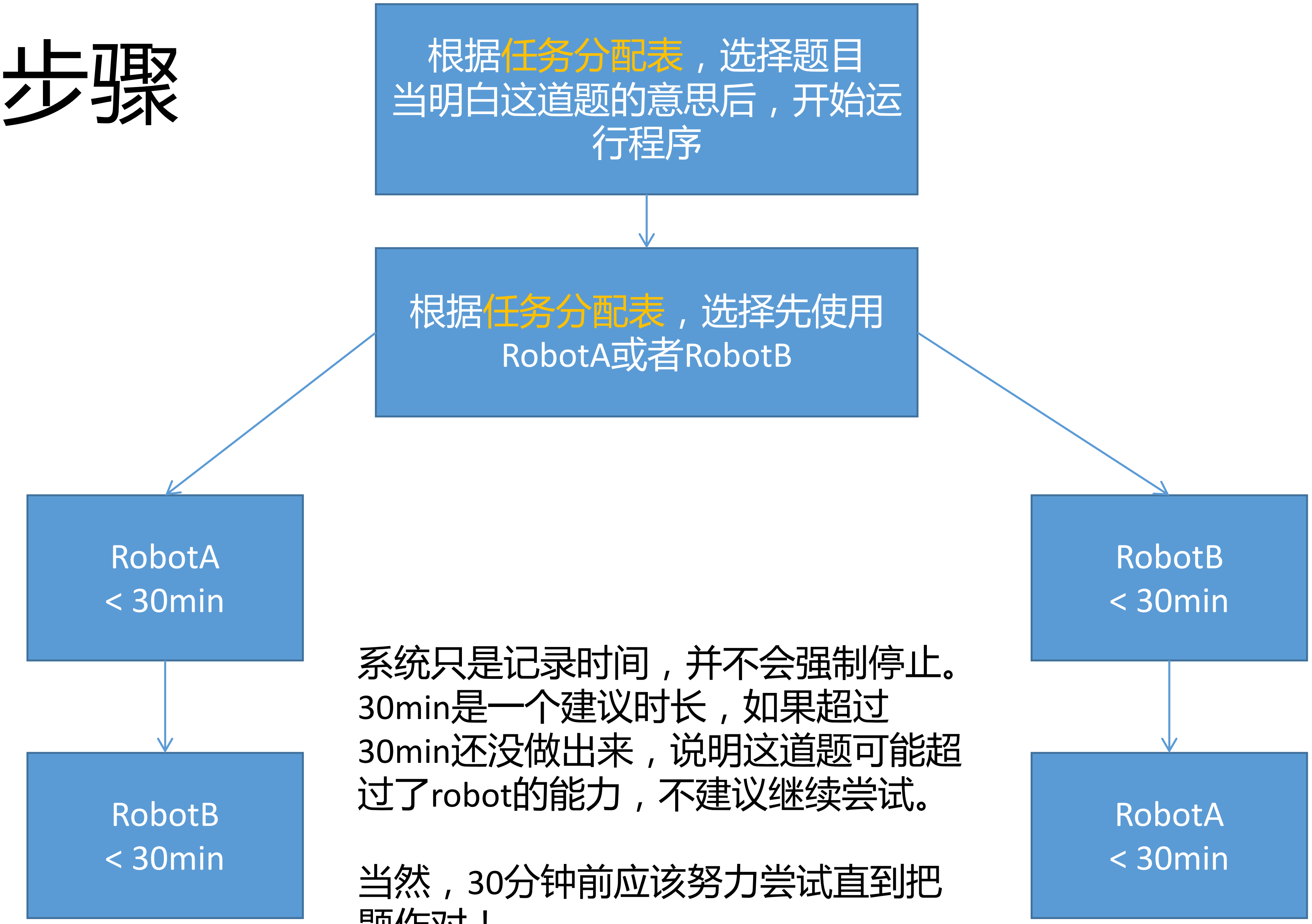
环境配置

- 一个非常简单的命令行工具
- pip install -r requirements.txt (openai, rich)

- 测试者需要
  - python >= 3.9
  - VPN (in terminal)
  - openai-key: 我们会发给大家，复制到key.txt即可

- python robotA.py | python robotB.py 启动不同的robot
- 将会发布在 http://62.234.201.16/nzy/anpl

实验步骤



根据任务分配表，选择题目当明白这道题的意思后，开始运行程序

根据任务分配表，选择先使用RobotA或者RobotB

RobotA < 30min

RobotB < 30min

RobotB < 30min

RobotA < 30min

系统只是记录时间，并不会强制停止。30min是一个建议时长，如果超过30min还没做出来，说明这道题可能超过了robot的能力，不建议继续尝试。

当然，30分钟前应该努力尝试直到把题作对！

题目分配

- 400 ARC + 100 编程 (?)
- ARC DDL: 5.12 24:00
- 按照序号分配题目
- 每天提交≥3道题，其余题目时间自由分配

数据收集 & 程序bug

- 测试记录存在log文件夹下
- 每天测试结束后，请将log文件夹下所有文件压缩并发送至
  - nanziyuan21s@ict.ac.cn

- 如果程序有bug，请立即微信联系！
  - 直接发送到微信群即可

- 可以通过gitlab issue/微信/邮箱交流其他问题~

Thanks