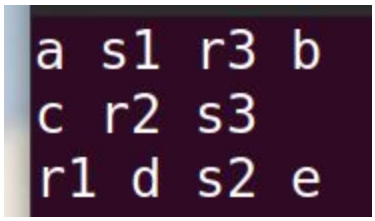
Part 1

Team member:

Yifan Wu yifanwu@csu.fullerton.edu

snapshot:

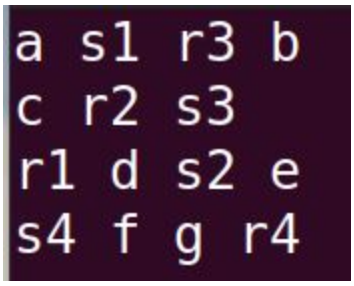
1. enter input to input.txt. Noted: do not use s0 or r0, use s1 as first send event.



2. run the program

```
student@tuffix-vm:~/Desktop/474proj1/part1$ javac yifan_1.java
student@tuffix-vm:~/Desktop/474proj1/part1$ java yifan_1
origin is:
a s1 r3 b
c r2 s3 null
r1 d s2 e
result is:
1 2 8 9
1 6 7 0
3 4 5 6
```

3. change input.txt



4. run the program

```
student@tuffix-vm:~/Desktop/474proj1/part1$ java yifan_1
origin is:
a s1 r3 b
c r2 s3 null
r1 d s2 e
s4 f g r4
result is:
1 2 8 9
1 6 7 0
3 4 5 6
1 2 3 4
```

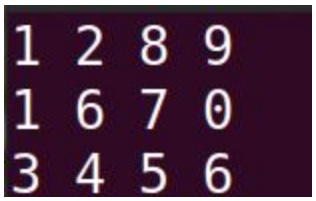
Pseudocode for part1:

```
//read input.txt and store the data to “origin” 2D array
//create “result” 2D array with same size of “origin”
traverse input.txt to get the number of max columns and rows N and M
String[N][M] origin
int[N][M] result

create send[] array, use send[0] to indicate the number of send events
for(each row of origin)
   for(each column of origin)
      if(origin[][] contains “r”)
         break, and go to next row
      else if(origin[][] contains “s”)
         store value to send[]

while(the number of send events is greater than 0)
   for(each row of origin)
      for(each column of origin)
         if(origin[i][j] contains “r”)
            //for receive_x event, find location of corresponding send_x event
            for(each row of origin)
               for(each column of origin)
                  get the column of send_x
            //get send_x value
            find the value of send_x from send[] array
            //write result to result[][]
            result[i][j] = max{ valueOf(send_x)+1, columnOf(send_x)+1 }
         else if(origin[i][j] contains “r”)
            //write to result[i][j] and send[] array
            result[i][j] = result[i][j-1] + 1
            send[] = result[i][j]
            increment send[0] as increasing size
         else
            //write to result[i][j] as it’s internal event
            result[i][j] = result[i][j-1] + 1

output result[][] as result
```

Part 2

snapshot:

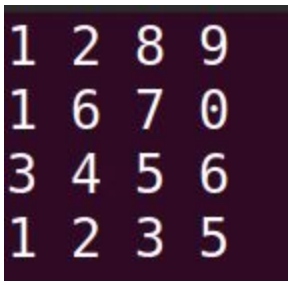
1. enter input to input.txt. Noted: do not use s0 or r0, use s1 as first send event.



2. run the program

```
student@tuffix-vm:~/Desktop/474proj1/part2$ java yifan_2
origin is:
1 2 8 9
1 6 7 0
3 4 5 6
result is:
a s1 r3 b
c r2 s3 null
r1 d s2 e
```

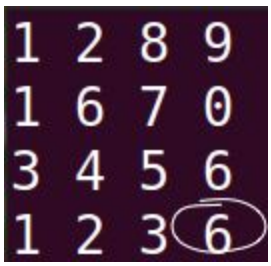
3. change input.txt



4. run the program

```
student@tuffix-vm:~/Desktop/474proj1/part2$ java yifan_2
origin is:
1 2 8 9
1 6 7 0
3 4 5 6
1 2 3 5
result is:
a s1 r3 b
c r2 s3 null
r1 s4 s2 d
e s1 f r4
```

5. change input.txt



6. run the program

```
student@tuffix-vm:~/Desktop/474proj1/part2$ java yifan_2
INCORRECT
```

Pseudocode for part2:

```
//read input.txt and store the data to “origin” 2D array
//create “result” 2D array with same size of “origin”
traverse input.txt to get the number of max columns and rows N and M
String[N][M] origin
String[N][M] result
create receive[] array, use receive[0] to indicate the number of receive events

//first step, traverse origin[][] and find gap number, then store value to receive[], increment receive[0], and store the gap number to result[][]
// if first column of origin is not ‘1’, then it must be gap number and receive events
for(each row i)
   if(origin[i][0] != 1)
      receive[1] = origin[i][0]
      receive[0]++
      result[i][0] = “r1”
// check next columns
for(each column of origin)
   for(each row of origin)
      if(origin[i][j] is a gap number)
      receive[x] = origin[i][j]
      receive[0]++
      result[i][j] = “r_x”

//second step, verify and compute value
while(receive[0] is greater than 0)
   for(each value in the receive[] array)
      boolean find = false // if there is a send_x event corresponding to a receive_x, true
      //if find send_x
      for(each row in origin)
         for(each column in origin)
            if(find send_x)
               find = true
               result[i][j] = send_x
               receive[0] decrement
      //if not find send_x
      if(find==false)
         print(“INCORRECT”)
         return program

//third step, fill internal letter to result[][]
for(each row in origin)
   for(each column in origin)
      if(it’s not receive or send event)
         result[i][j]  = internal event letter

output result[][] as result
```