Báo Cáo Bài Tập Lớn

môn

Trí Tuệ Nhân Tạo

Nhóm 4

Cấn Duy Cát
Nguyễn Minh Trang
Nguyễn Mạnh Duy
Kiều Minh Đức
Nguyễn Mạnh Cường

01-04-2016

Mục lục

Lịch sử thay đổi

Người thay đổi	Ngày thay đổi	Lý do thay đổi	Phiên bản
Cat Can	01-04-2016	Tạo mẫu tài liệu	1.0
Duc Kieu	10-04-2016	Cập nhật phần 1 và phần 2	1.1
Cat Can	13-04-2016	Hoàn thành báo cáo	1.2

1. Giới thiệu

1.1 Mục đích

Tài liệu này dùng để phục vụ cho bài tập lớn: Fill-in-station trong môn Trí tuệ nhân tạo. Tài liệu sẽ miêu tả lại phát biểu bài toán, không gian trạng thái, các hàm mục tiêu và so sánh các hàm đánh giá chính.

1.2 Nội dung tài liệu

- **Phần 1: Không gian trạng thái**. Mô tả các state, initial state, các action, transition model, path cost và goal test.
- **Phần 2: Phát biểu bài toán**. Mô tả lại bài toán, các input, hướng giải và mục đích của bài toán.
- **Phần 3: Hàm mục tiêu**. Mô tả lại các hàm mục tiêu của bài toán.
- **Phần 4: So sánh hàm đánh giá**. So sánh hàm đánh giá. Nêu lại hai hàm đánh giá của thuật toán, cách cài đặt và so sánh hiệu năng của chúng với nhau (một hàm đánh giá do thầy giáo yêu cầu và một hàm đánh giá nâng cao của nhóm tự làm)

1.3 Tài liệu tham khảo

[1] http://www.cs.columbia.edu/~kathy/cs4701/Assignments/hw2-kt.html

2. Không gian trạng thái

2.1 Trạng thái

Ma trận gồm 3 hàng 3 cột được đánh số như hình vẽ:

1	2	3
4	5	6
7	8	9

Mỗi thao tác đặt một chữ cái vào ma trận sẽ tạo ra một trạng thái mới, do đó sẽ có các loại trạng thái có từ 0 đến 9 chữ cái. Với loại trạng thái n thì số lượng trạng thái sẽ là chỉnh hợp chập n của 9. Do đó không gian trạng thái sẽ có tổng cộng số trạng thái là:

$$N = \sum_{i=1}^{9} A_9^i = 986410 \text{ (trạng thái)}$$

Số trạng thái trên tương ứng với trường hợp toàn bộ các chữ cái đã cho là khác nhau đôi một. Với bộ input có nhiều hơn hoặc bằng một cặp từ giống nhau thì số lượng trạng thái trong không gian trạng thái sẽ giảm đi do các trạng thái trùng lặp nhau.

2.2 Trạng thái ban đầu

Trạng thái ban đầu là trạng thái ma trận chưa có chữ cái nào. Trạng thái ban đầu là duy nhất. Ta bắt đầu tìm kiếm từ trạng thái ban đầu bằng cách thêm các chữ cái lần lượt vào các vị trí trong ma trận.

2.3 Trạng thái đích

Trạng thái đích là trạng thái ma trận gồm đủ 9 chữ cái và các bộ 3 chữ cái nằm tại (1, 2, 3); (4, 5, 6); (7, 8, 9); (1, 4, 7); (2, 5, 8); (3, 6, 9); (1, 5, 9); (3, 5, 7) kết hợp với nhau theo thứ tự tạo thành một từ có nghĩa nằm trong từ điển.

Do bộ input là ngẫu nhiên nên số lượng trạng thái đích tương ứng với mỗi bộ input là khác nhau, có bộ input không có trạng thái đích (không có kết quả)

2.4 Trạng thái con

Với mỗi trạng thái thì việc thêm một chữ cái vào vị trí trống tiếp theo trong ma trận sẽ tạo ra một trạng thái con. Trạng thái ban đầu có nhiều trạng thái con nhất là 8 trạng thái con (trường hợp các chữ cái khác nhau đôi một) và trạng thái đích không có trạng thái con.

Với các trường hợp có 2 chữ cái giống nhau thì chỉ tính 1 trạng thái con.

2.5 Độ sâu cây tìm kiếm

Độ sâu cây tìm kiếm chính là số lượng chữ cái đã đặt vào ma trận. Độ sâu lớn nhất của cây tìm kiếm là 8. Trạng thái đích có độ sâu là 8. Trạng thái ban đầu coi độ sâu là 0.

3. Phát biểu bài toán

Ta xem xét bài toán như việc tìm đường đi trong một ma trận, trong đó ma trận được chuyển về dạng đường đi tuyến tính bằng cách chuyển nó thành một dãy các chữ cái bắt đầu từ hàng 1 rồi đến hàng 2 và cuối cùng là hàng 3. Tại mỗi bước di chuyển, thuật toán tìm kiếm sẽ lựa chọn chữ cái tiếp theo để đặt vào ma trận.

Hàm mục tiêu cần đảm bảo ràng buộc khi ta chọn chữ cái nằm ở vị trí cuối cùng của bất kỳ từ nào trong ma trận thì từ đó phải có nghĩa. Điều đó đồng nghĩa với việc khi thuật toán tìm kiếm tìm đến độ sâu 8 thì việc tìm kiếm hoàn thành.

4. Hàm mục tiêu

Hàm mục tiêu sẽ trả về toàn bộ các chữ cái có thể đặt vào vị trí tiếp theo còn trống trong ma trận. Để tối ưu hoá hàm mục tiêu, các chữ cái có tần suất với từ đứng trước bằng không và các chữ cái cuối cùng làm cho từ không có nghĩa sẽ bị loại.

Ngoài ra hàm mục tiêu sẽ loại đi các chữ cái trùng lặp đảm bảo rằng các trạng thái đã đi qua sẽ không quay lại lần thứ hai.

Hàm mục tiêu, được triển khai trực tiếp trong thân hàm chạy thuật toán.

Việc kiểm tra chọn chữ cái nằm ở vị trí cuối cùng của bất kỳ từ nào trong ma trận thì từ đó phải có nghĩa được thực hiện trong hàm heuristic. Khi từ không có nghĩa hàm heuristic trả về 0 tức là số điểm nhỏ nhất và trạng thái đó bị loại ngay khỏi hàng đợi.

5. So sánh hàm đánh giá

5.1 Phân tích hàm đánh giá

5.1.1 Hàm đánh giá thông thường

Hàm đánh giá thông thường sẽ trả về tổng giá trị tần suất cặp (bigram frequency) của tất cả các cặp chữ cái tạo bởi các từ đã có trong ma trận với chữ cái tiếp theo được lựa chọn. Sau khi triển khai hàm đánh giá thông thường thì chữ cái tiếp theo được lựa chọn sẽ là chữ cái có tổng giá trị tần suất cặp lớn nhất.

Với chữ cái đầu tiên, sẽ lựa chọn chữ cái có giá trị tần suất đứng đầu (initial frequency) cao nhất trong danh sách tần suất cặp.

5.1.2 Hàm đánh giá nâng cao

Với hàm đánh giá nâng cao, thay vì chỉ xét những chữ cái đã đặt vào ma trận thì hàm đánh giá sẽ xét thêm cả những chữ cái trong tương lai kèm theo cơ chế cộng điểm cho các chữ cái khi đặt vào sẽ làm cho khả năng tạo từ có nghĩa trong tương lai tăng lên.

5.2 So sánh

Thông tin chạy thử nghiệm:

- Hệ điều hành: windows 10 Home Single Language
- Chạy trên Command Line
- Python version: 2.7.11
- Cấu hình máy thử nghiệm:
  CPU Intel Core i5-3337U 1.80GHz
  RAM 4.00 GB (3.88 usable)
  64-bit Operating System.
- Hàm chạy thử nghiệm: main.py
- Thời gian: 20:00 – 20:30 ngày 13-04-2016
- Người chạy: Cấn Duy Cát

Kết quả chạy thử nghiệm:

Sau khi chạy thử 10 trên 1,000 bộ dữ liệu khác nhau cho mỗi lần (tương đương 10,000 bộ dữ liệu), thống kê kết quả chạy cho cả 2 thuật toán có được như sau:

	Lần chạy thử	1	2	3	4	5	6	7	8	9	10
Phần 2	Thời gian (ms)	19	20	26	26	29	21	20	19	20	21
	Số state (state)	795	835	833	867	856	860	829	810	855	842
Phần 3	Thời gian (ms)	16	17	22	22	21	17	17	17	17	18
	Số state (state)	223	235	237	236	232	238	236	232	236	239

(Hình ảnh các lần chạy thử nghiệm được đặt trong phụ lục kèm theo.)

Công thức thống kê:

- Thời gian chạy trung bình:

$$\bar{t} = \frac{1}{10}\sum_{i=1}^{10} t_i$$

- Số trạng thái trung bình:

$$\bar{n} = \frac{1}{10}\sum_{i=1}^{10} n_i$$

- EBF: nghiệm x của phương trình:

$$\bar{n} = x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 = \sum_{i=1}^{9} x^i$$

Với $\bar{n}$ là số trạng thái trung bình được tính trong phần trên

Giá trị EBF có thể tính gần đúng bằng công thức:

$$\text{EBF} = \bar{n}^{\frac{1}{9}}$$

Trong các phần tiếp theo giá trị EBF tính bằng công thức chính xác làm tròn đến 3 chữ số sau dấu phảy.

5.2.1 Thuật toán thông thường

Thời gian chạy trung bình: 22 ms

Số trạng thái trung bình: 838 trạng thái

EBF: 1.951

5.2.2 Thuật toán nâng cao

Thời gian chạy trung bình: 18 ms

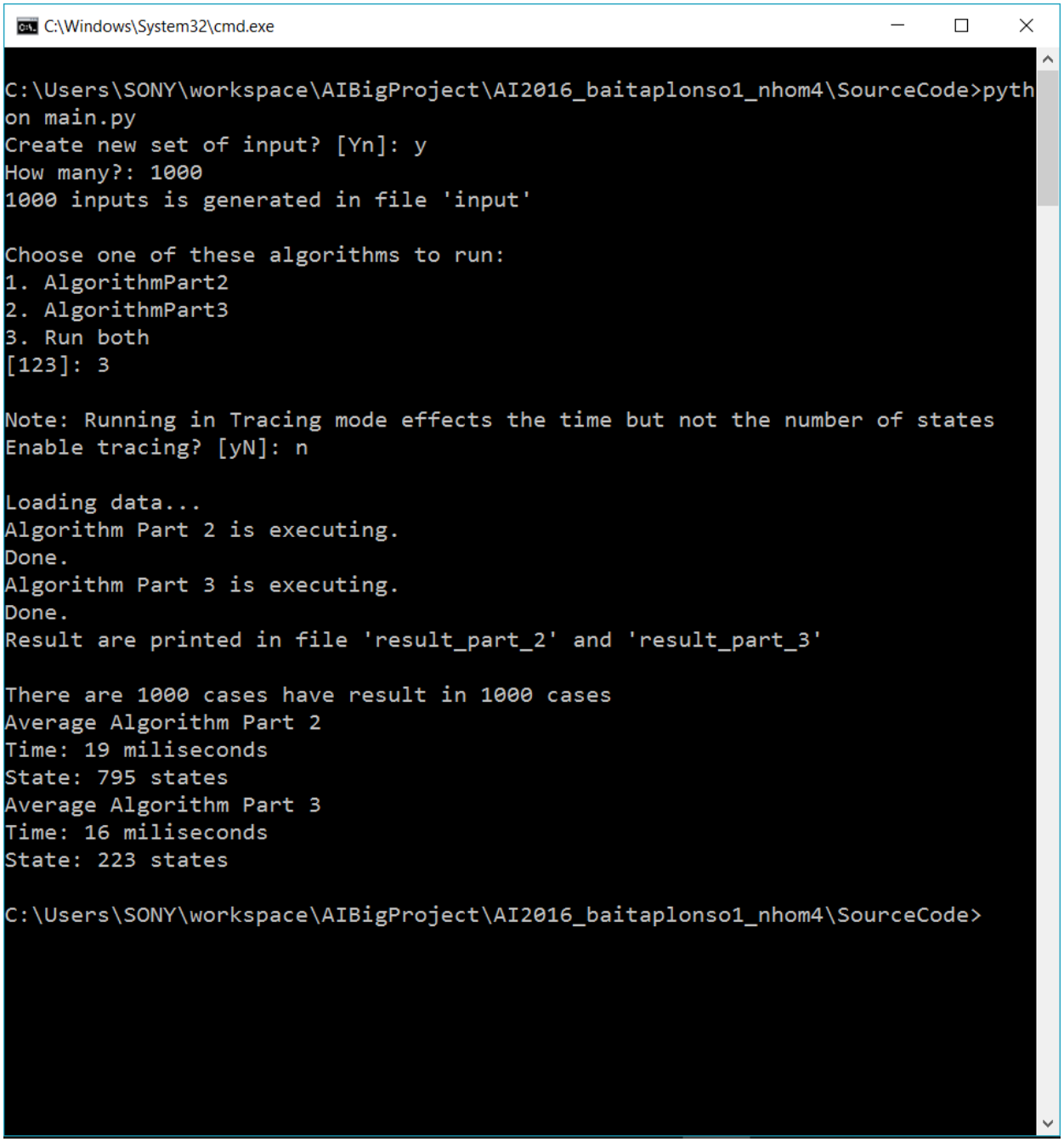
Số trạng thái trung bình: 234 trạng thái

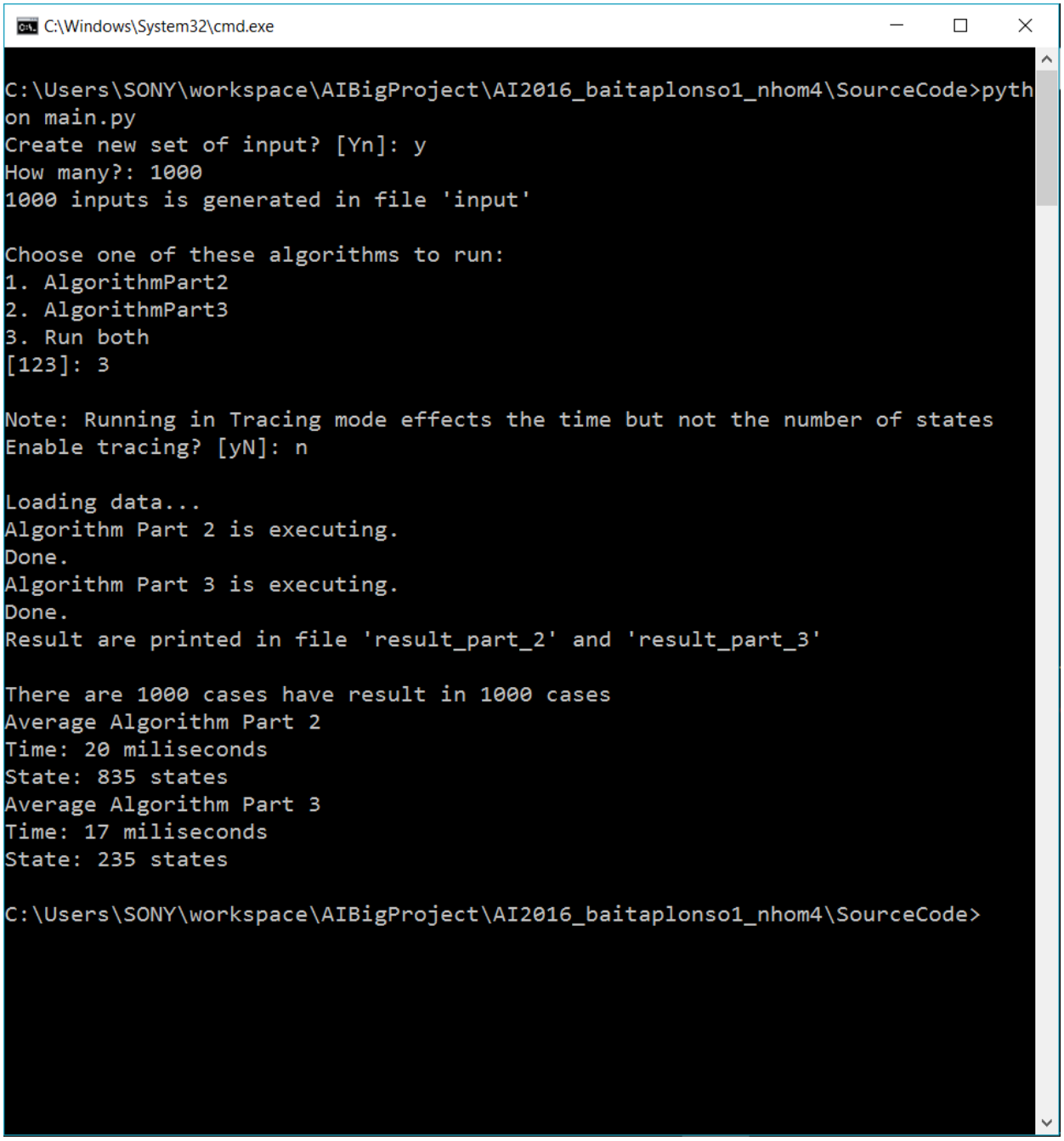
EBF: 1.656

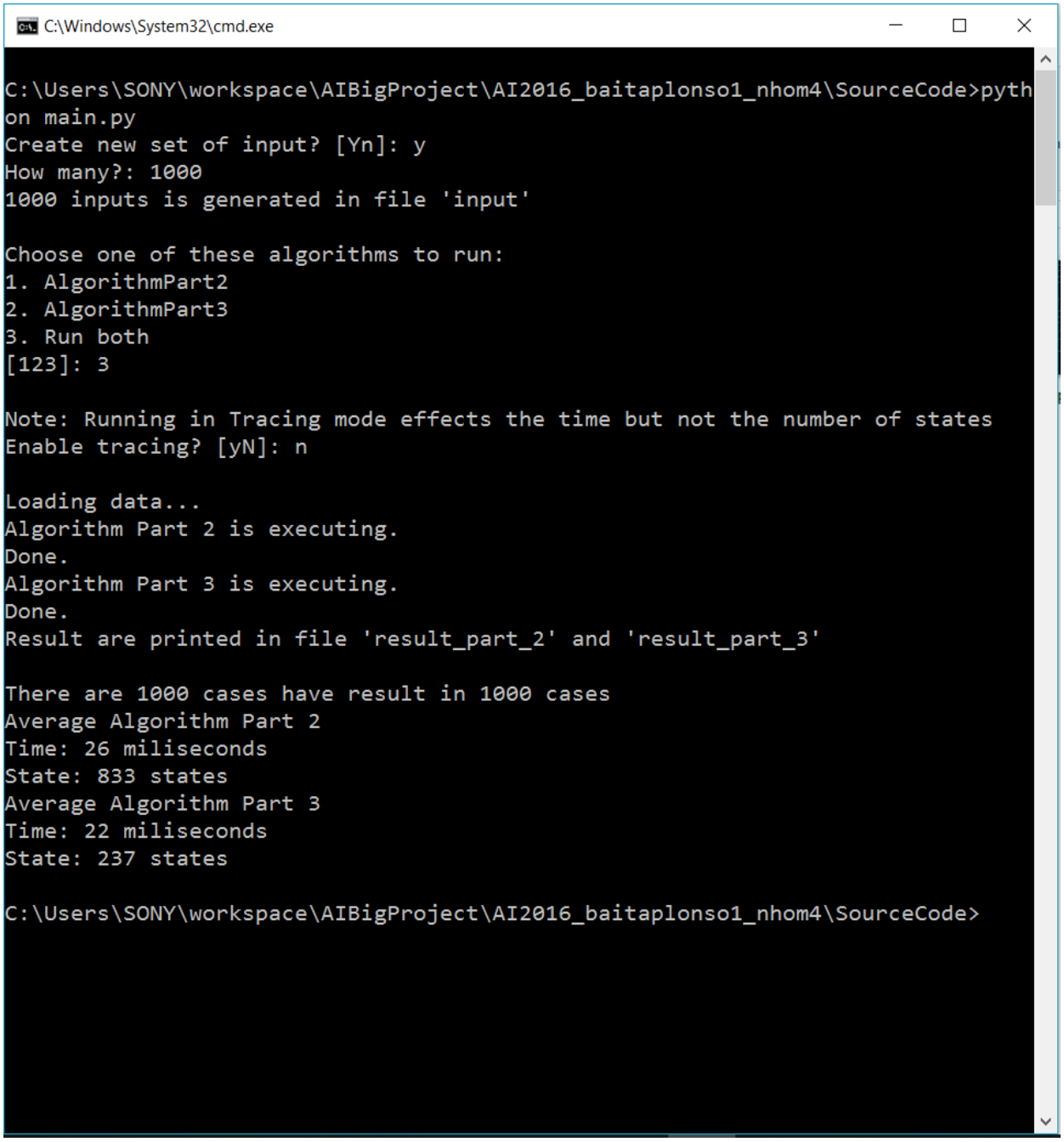
Kết luận:

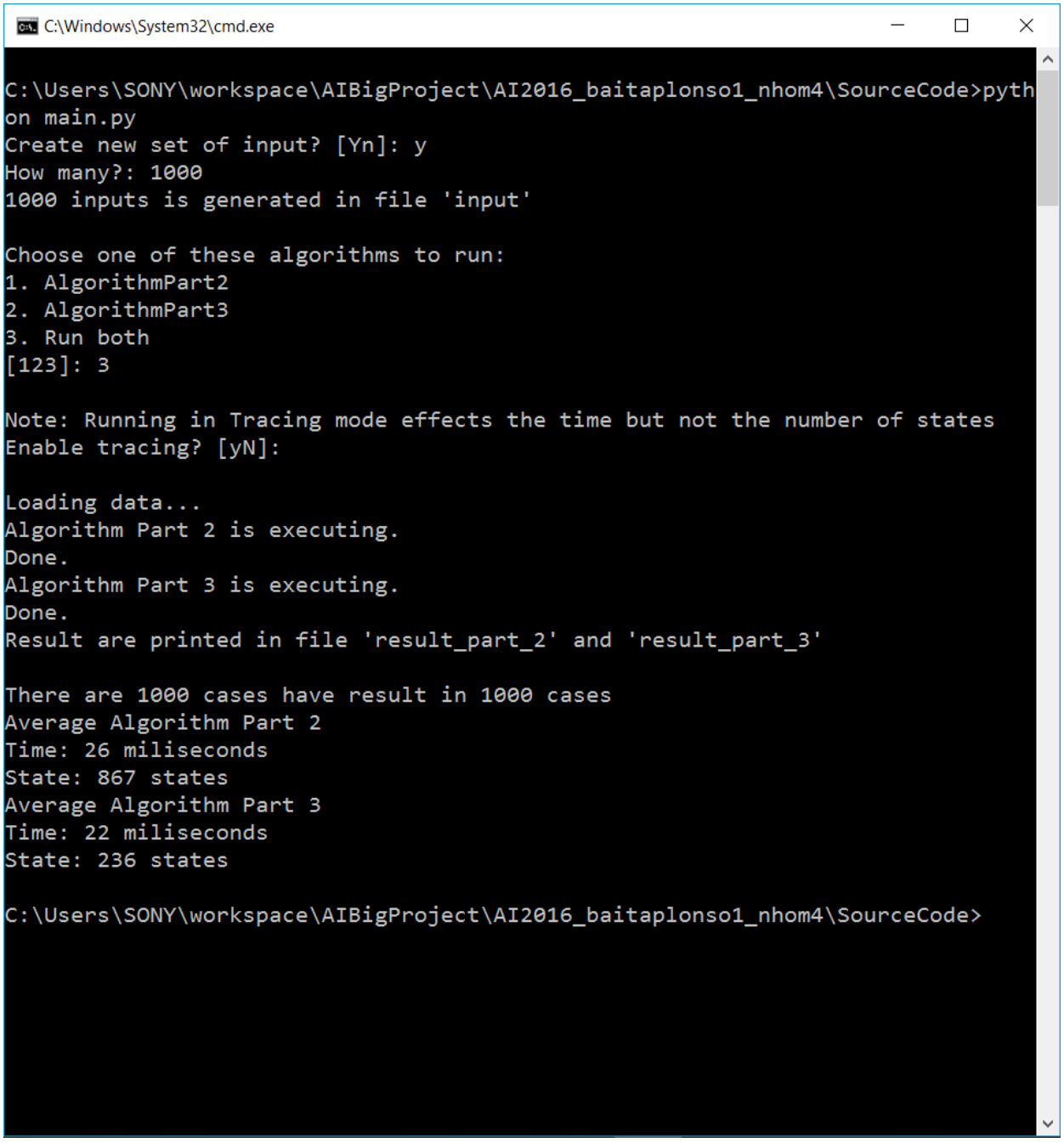
Qua thử nghiệm thì thuật toán nâng cao có hiệu năng tốt hơn thuật toán thông thường. Thời gian giảm 20%. Số trạng thái cần duyệt giảm 70%.

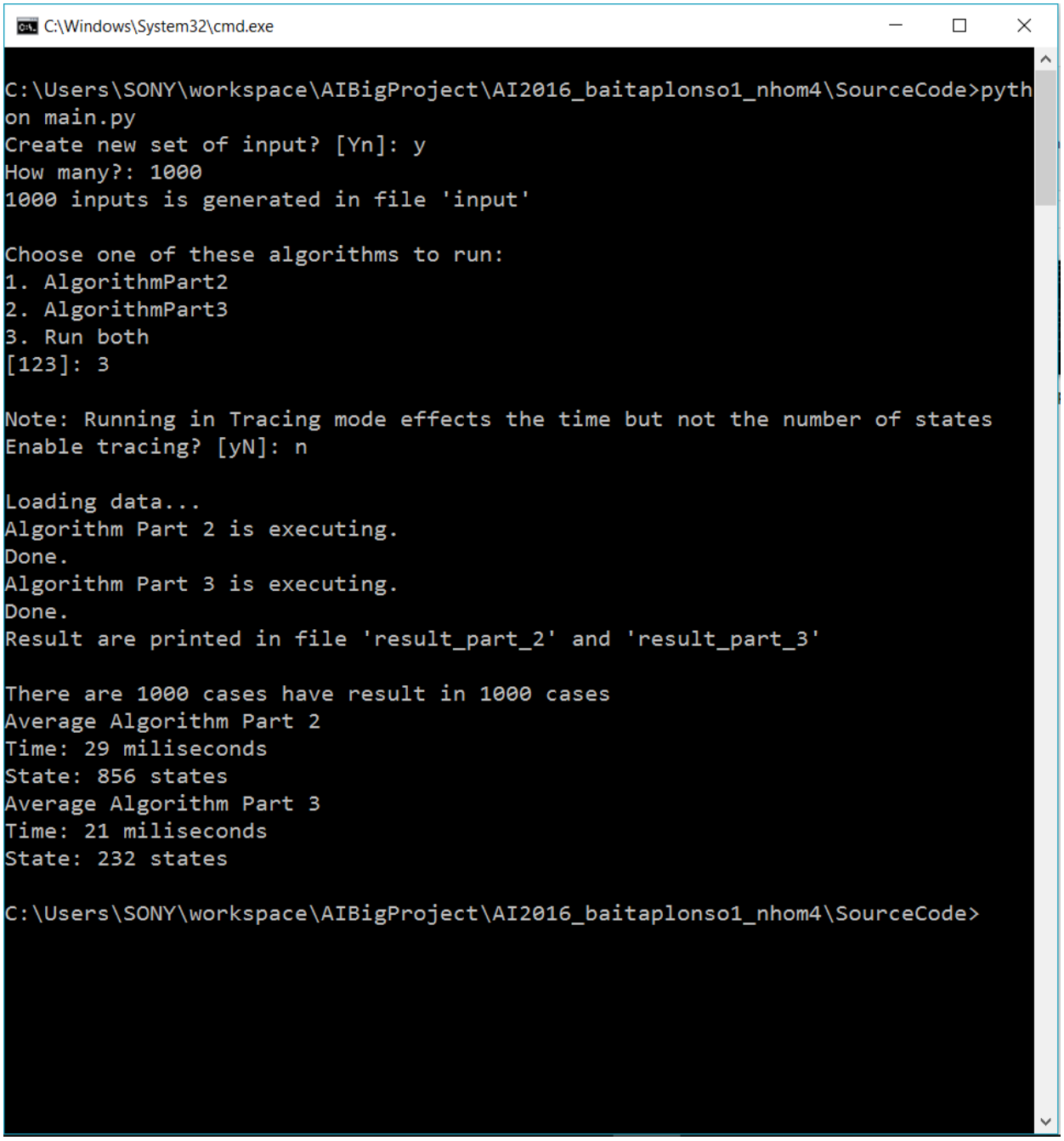
Phụ lục A: Hình ảnh chạy thử nghiệm

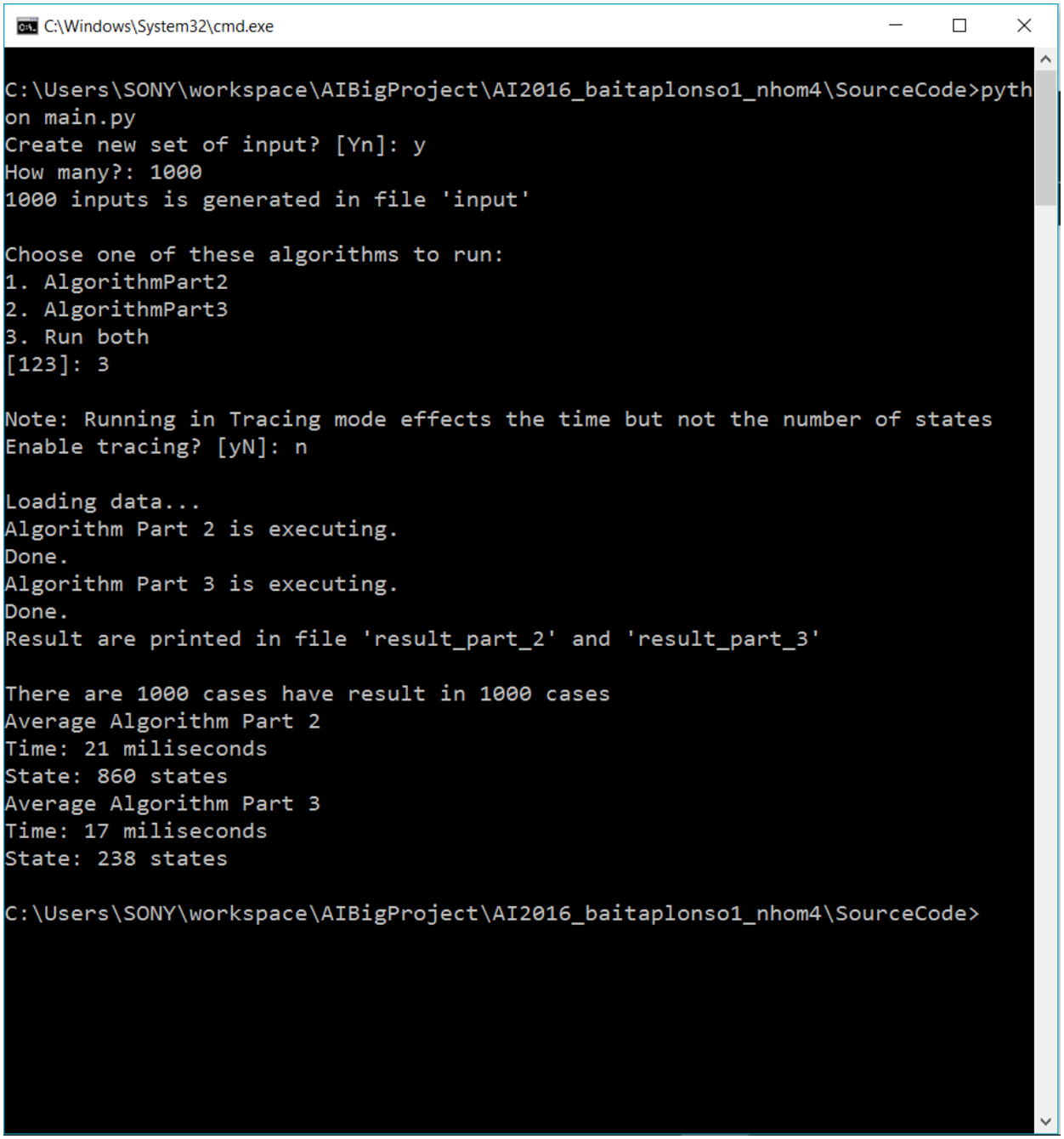


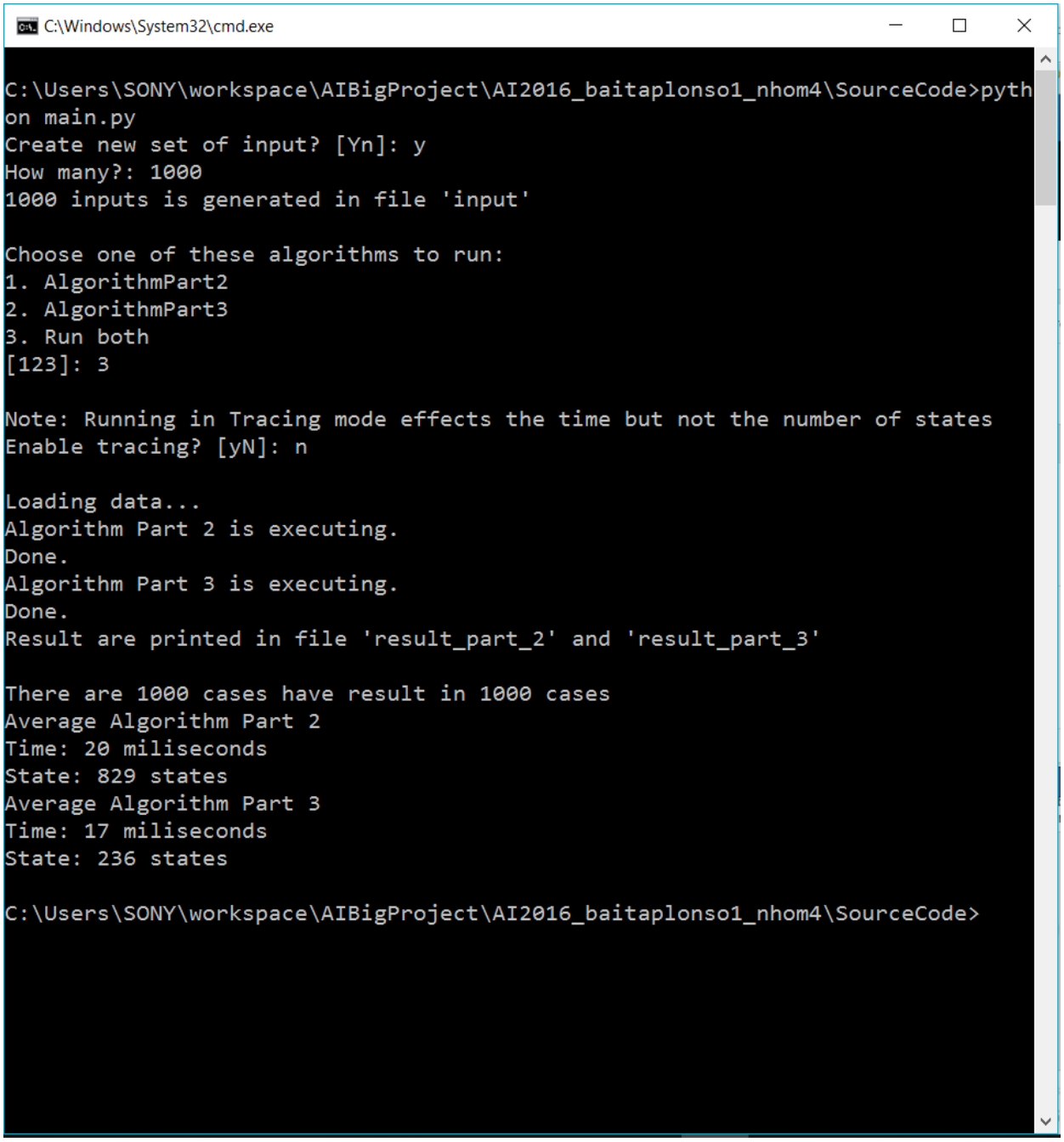

C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 20 miliseconds
State: 835 states
Average Algorithm Part 3
Time: 17 miliseconds
State: 235 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>

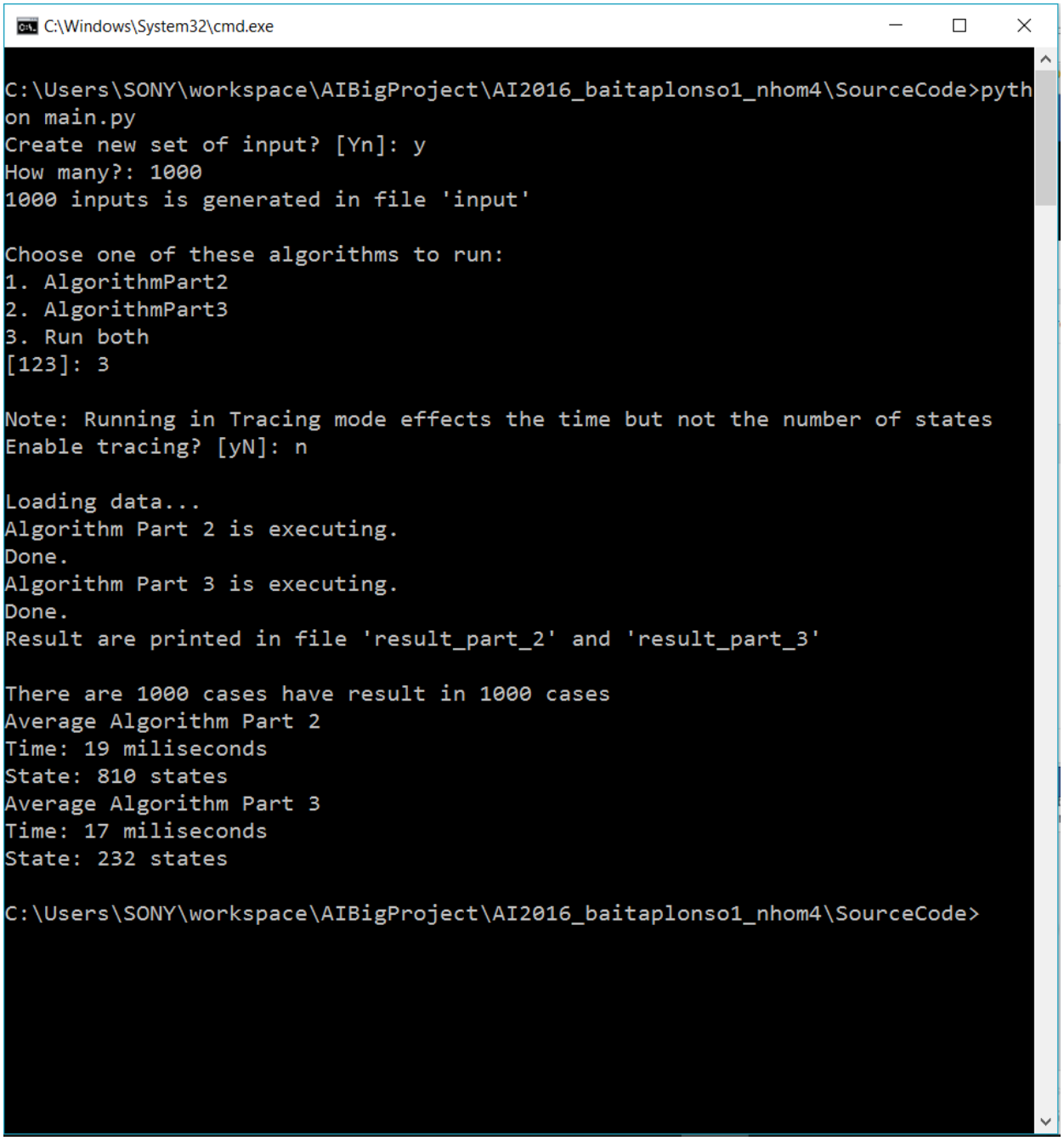

C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 26 miliseconds
State: 833 states
Average Algorithm Part 3
Time: 22 miliseconds
State: 237 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>

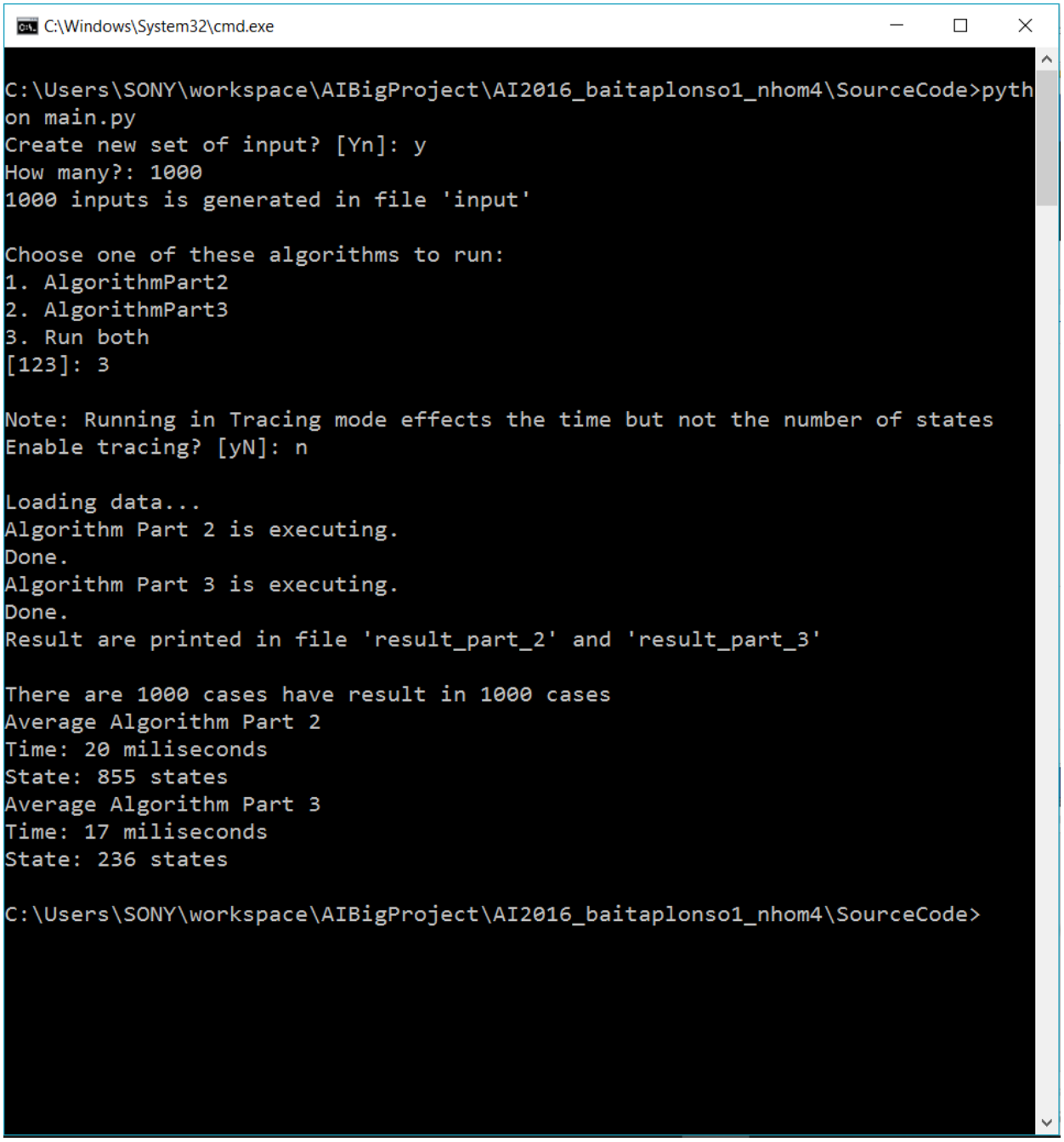

C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]:
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 26 miliseconds
State: 867 states
Average Algorithm Part 3
Time: 22 miliseconds
State: 236 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>

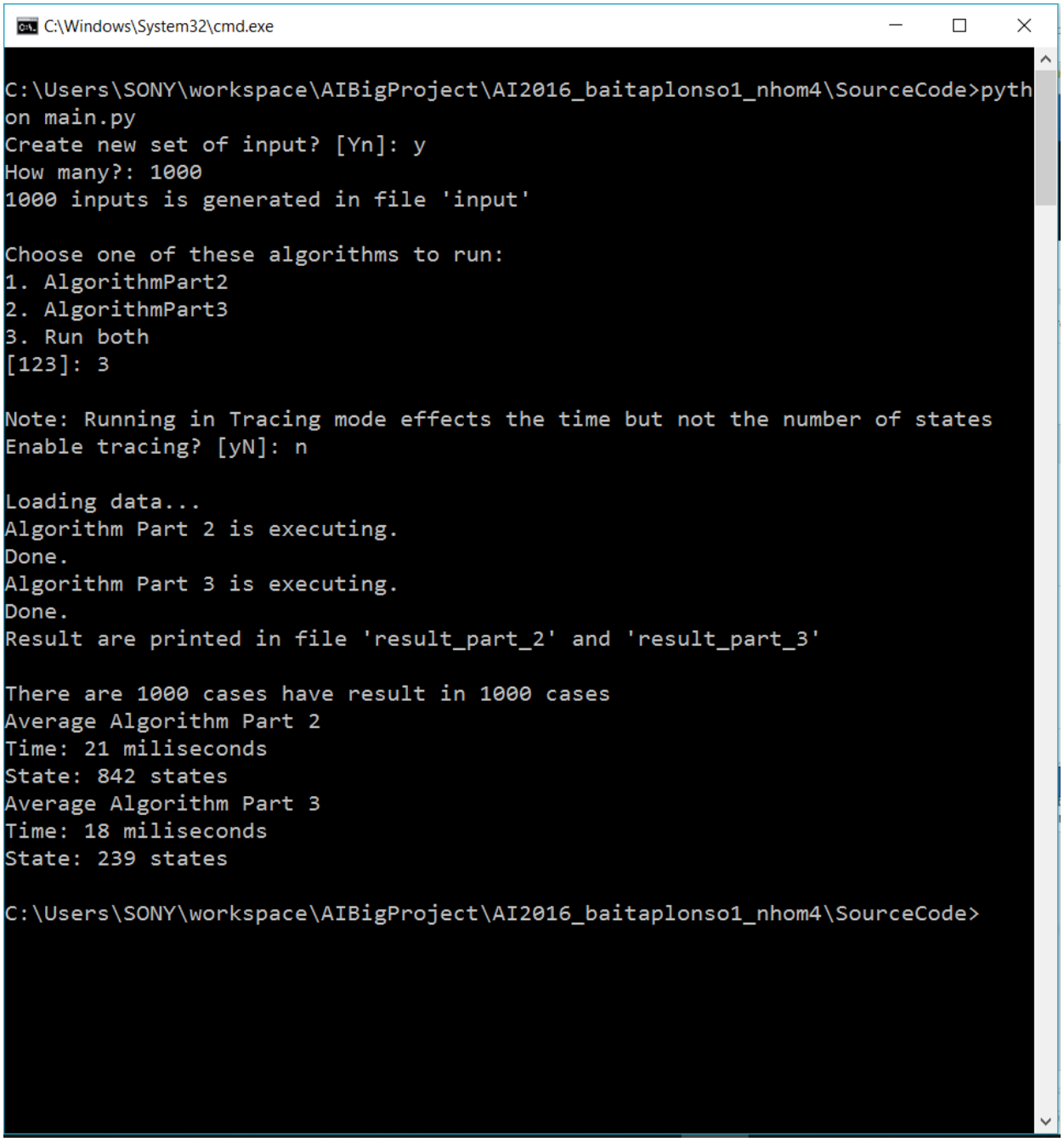

C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 29 miliseconds
State: 856 states
Average Algorithm Part 3
Time: 21 miliseconds
State: 232 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>


C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 21 miliseconds
State: 860 states
Average Algorithm Part 3
Time: 17 miliseconds
State: 238 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>


C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 20 miliseconds
State: 829 states
Average Algorithm Part 3
Time: 17 miliseconds
State: 236 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>


C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 19 miliseconds
State: 810 states
Average Algorithm Part 3
Time: 17 miliseconds
State: 232 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>


C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 20 miliseconds
State: 855 states
Average Algorithm Part 3
Time: 17 miliseconds
State: 236 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>


C:\Windows\System32\cmd.exe
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>pyth
on main.py
Create new set of input? [Yn]: y
How many?: 1000
1000 inputs is generated in file 'input'
Choose one of these algorithms to run:
1. AlgorithmPart2
2. AlgorithmPart3
3. Run both
[123]: 3
Note: Running in Tracing mode effects the time but not the number of states
Enable tracing? [yN]: n
Loading data...
Algorithm Part 2 is executing.
Done.
Algorithm Part 3 is executing.
Done.
Result are printed in file 'result_part_2' and 'result_part_3'
There are 1000 cases have result in 1000 cases
Average Algorithm Part 2
Time: 21 miliseconds
State: 842 states
Average Algorithm Part 3
Time: 18 miliseconds
State: 239 states
C:\Users\SONY\workspace\AIBigProject\AI2016_baitaplonso1_nhom4\SourceCode>