**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**
**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**
**KHOA CÔNG NGHỆ THÔNG TIN**


# Báo cáo Lab01

# Search Strategies

**|Giảng viên hướng dẫn|**

**Dr. Nguyễn Ngọc Thảo**

**Dr. Nguyễn Hải Minh**

**|Sinh viên thực hiện|**

**Phan Tấn Đạt 18127078**

**Môn học: Cơ sở trí tuệ nhân tạo (18CLC3)**

Thành phố Hồ Chí Minh – 2020

## 1. Đánh giá công việc:

| Công việc | Hoàn thành | Chưa hoàn thành |
|---|---|---|
| Thực hiện Breadth-first search theo yêu cầu | ✓ | |
| Thực hiện Uniform-cost search theo yêu cầu | ✓ | |
| Thực hiện Iterative deepening search theo yêu cầu (uses depth-first tree search as core component and avoids loops by checking a new node against the current path) | ✓ | |
| Thực hiện Greedy-best first search theo yêu cầu (using the Manhattan distance as heuristic) | ✓ | |
| Thực hiện Graph-search A* theo yêu cầu (using the same heuristic as above) | ✓ | |
| Xuất ra kiểu dữ liệu của kết quả các thuật toán theo yêu cầu | ✓ | |
| In ra console kết quả | ✓ | |
| Đọc dữ liệu từ file trong folder INPUT | ✓ | |
| Lưu kết quả ra file trong folder OUTPUT | ✓ | |

## 2. Mô tả bài làm:

Sử dụng ngôn ngữ Python 3 để thực hiện các thuật toán tìm kiếm trên một graph được tạo ra bằng các đọc 1 file định dang ".txt"

```
import glob

def choose_input_files(dir):
    # ..\INPUT\
    file_list = glob.glob(dir + "\*.txt")
    if file_list:
        for i in range(len(file_list)):
            print("<" + str(i) + ">\t" + file_list[i])
        while (1):
            get_choice = input("Input a GIVEN number to choose data file: ")
            choice = int(get_choice)
            if choice >= 0 and choice < len(file_list):
                return file_list[choice]
    else:
        print("No input file was found in given directory!!!!")
        return None
```

function `choose_input_files(dir)` trong "file_tools.py" sẽ tìm các file định dang ".txt" trong folder INPUT để người dùng lựa chọn theo tên file trên console, hình minh họa:

```
Reloaded modules: Breadth_first_search, Classes, Frontier_processing,
Uniform_cost_search, Greedy_best_first_search, A_star_graph_search,
Iterative_deepening_search, file_tools
<0> ..\INPUT\data.txt
<1> ..\INPUT\input.txt
<2> ..\INPUT\input_old.txt
<3> ..\INPUT\maze00.txt
<4> ..\INPUT\maze01.txt

Input a GIVEN number to choose data file: 1
```

Trong "file_tools.py" chứa định nghĩa 2 function **def ImportData**(file_dir: str): và **def OutputData**(file_dir: str, algorithm_name: str, data): là 2 function để đọc, xuất file. Ngoài ra còn có định nghĩa của function **def printResult**(algorithm_name, data): để in kết quả ra console

Mỗi đỉnh của graph được lưu thành 1 object class Node định nghĩa trong file "Classes.py"

Một mê cung(graph) chứa các đỉnh Node được lưu thành object class Maze định nghĩa trong file "Classes.py"

Các thuật toán tìm kiếm được đặt ở trong các file ".py" có tên tương ứng và trả ra dạng kết quả theo yêu cầu

Compile file "main.py" để thực hiện load dữ liệu vào graph và chạy các thuật toán tìm kiếm:

```
from Breadth_first_search import Breadth_first_search
from Uniform_cost_search import Uniform_cost_search
from Greedy_best_first_search import Greedy_best_first_search
from A_star_graph_search import A_star_graph_search
from Iterative_deepening_search import Iterative_deepening_search

from Classes import Maze
from file_tools import OutputData, ImportData, choose_input_files, printResult
import sys
# ----------------------------------------
if __name__ == "__main__":
    file_name = choose_input_files("..\INPUT")

    if file_name is not None:
        input_list = ImportData(file_name)
        if len(input_list) < 3:
            print("No data was imported")
            print(input_list)
            sys.exit()
        size = int(input_list.pop(0))
        goal = int(input_list.pop(-1))
        if (goal < 0) or (goal > int(size * size)):
            print("\n[Warning]: Goal doesn't exist in Maze!\n->This might result in
long runtime and uncompleted result!!\n")

        board = Maze(size,input_list,goal)

        # start = input("Enter the number of starting point: ")
        # start = int(start)
        start = 0

        print("Starting point:\t", start)
```

```
        print("Goal:\t\t\t", goal)

        algorithms = [(Breadth_first_search),
                      (Uniform_cost_search),
                      (Iterative_deepening_search),
                      (Greedy_best_first_search),
                      (A_star_graph_search)]
        for method in algorithms:
            result = method(board, start, goal)
            #print(method.__name__ + " completed\n")
            OutputData("..\OUTPUT\ ", method.__name__, result)
            printResult(method.__name__,result)
```