CSCE 625
Programing Project #2: Implementation (No written report and video portion for project 2)
Due: 4/19 by Midnight

Problem Statement:

This is a simplification of the problem that was discussed in class for the brainstorming and quick look:

- There is a set of requests for missions. The set is static and the number does not change. The number of requests can be from 0 to 60
- Each mission can be met with a model of sUAS that is associated for that mission. For each mission there is a list of acceptable sUAS models. There are 7 mission types. The list of acceptable sUAS models has at least 1 sUAS and up to 14 models.
- There is a set of available pilots. The set is static and the number does not change. The number of pilots will be between 2 and 13.
- A pilot may be qualified to fly 1 or more models of sUAS.
- A pilot can fly up to 3 missions. They will use the same sUAS for all missions.
- There is a set of sUAS that are available. The set consists of 1 up to and including 14 sUAS models. There may be more than one instance of a model. Thus the set of sUAS could be: 5 Mavics, 1 P4, 1 AR180, 5 PHK, 2 Solo, 1 Falcon
- Find a mapping of missions, sUAS, and pilots such that all the missions requests are satisfied without violating any constraints; if impossible, then flag and terminate without crashing.
- Extra credit: each pilot has a model of sUAS that they prefer. Find a mapping that maximizes the number of pilots getting to use their favorite sUAS model.

Implementation:
• Programming Language: you are free to choose from (C++, Python, Java, Matlab) for algorithm implementation, but we recommend the gecode toolkit.
• Toolkit: we recommend the gecode (http://www.gecode.org/index.html) implemented in C++ for solving CSPs. The documentation of gecode is available here (http://www.gecode.org/documentation.html).

• Input file format: The input file format is at end of document

Program Output:

You will be given 3 different input files. The program should produce an list of missions, model, and pilots that satisfies the constraints. If the program cannot satisfy the constraints, then it should flag that condition and terminate without crashing.

Output file format: The output file should be a .txt (e.g., mapping_1.txt) in which each line contains 4 elements (mission number, sUAS model, pilot name, pilot's favorite or not) representing the mapping of each mission.

For example, mapping_1.txt might be:
M1 Mavic Adams Yes
M2 P4 Broder No
M3 PHK Gringrich Yes
M4 P4 Peters No
...
M60 PHK Smith No

What to Turn in:
• You need to turn in your submission (source code and output files with required format) in a .zip file using the eCAMPUS. Please include any files that are needed to compile and run the program. For the programs which use C++, a Makefile will be very helpful.
• Include a README file in your submission which provides any instructions necessary for compiling and running your program in terminal. It is your responsibility to make sure the TA can compile and test your program. You can assume that gecode package is already installed.

Grading Rubric:

The TA will examine your code and output.

• [15 points] Code correctly expressed the domain and selected in gecode or designed the appropriate constraints in the code.
• [30 points] The output mapping (of missions, sUAS, and pilots) satisfies all the basic constraints (all the constraints except the pilot's favorite model) and complete all the missions. TA is writing a script to check the constraints if you claim that all the constraints are satisfied in your mapping.
• [5 points] Output mapping is "pretty print". That is, the output mapping clearly shows the mission number, the UAS model, the name of the pilot and whether the UAS model is pilot's favorite in a way that is easy to read.

• [Bonus 10 points] Additional code to produce an output mapping that satisfies all constraints while maximizing the number of pilots with their favorite UAS model.

Plagiarism and Late Policy:
Plagiarism: This project is to be done individually. The software will be checked for plagiarism using Moss so be very careful. It is ok to discuss the project with each other on a general level, e.g., "I'm using a stack instead of a queue," but not at the "here is my function call" level. A "let's sketch out the main program together" can be fine if it is clear that the collaborating students did their own coding.

Late Policy: 20% per day after deadline.

Input file format (no %%% just there to explain):

```
%%% Vector of UAS model name (12) which will need for pretty printing
AR180
AR200
Disco
Falcon
Inspire
Mavic
M600
P3
P4
PHK
ScanEagle
Solo
%%%%
MT A 011000010000  % 12 numbers representing 12 UAS models as a vector; "1" means that model can satisfy that for the mission
MT B 000000010000
MT C 000000010001
MT D 111100010001
MT E  110000000010
MT F 110000000110
MT G 111110111001
%%% Pilot name, UAS models can fly, favorite UAS model
P Merrick  011000010000  010000000000
P Adams 011000010000 010000000000
```

P Gingrich 011000010000 010000000000
%%% Number of UAS available by model
NUAS 153000079211 % 12 characters—each is the number of a UAS model, no more than 9 of any model
%%% Mission name and Mission Type
M 1 A
M 2 B
M 3 B
M 4 D
M 5 E
M 6 A
M 7 C
M 8 F
M 9 A
M10 A
M 11 B
M 12 C
M 13 E
M 14 D
M 15 B