Programming Assignment 1:

Quine-McCluskey Method (Due: 10/27)

Problem Descriptions:

Implement a two-level logic optimizer based on Quine-McCluskey method. The first step is to generate all prime implicants for the given function. The second step is to choose a minimum-cost cover and generate a sum-of-products expression of this function with **minimum number of prime implicant** (If there exist different solutions with the same minimum number of prime implicant, please choose the one which has minimum number of literals).

Input format:

You should allow input from a file. The following is an example.
f(A,B,C,D) = ∑ m(4,5,6,8,9,10,13) + ∑ d(0,7,15).

```
.i 4          /* the function f has 4 input variables: A,B,C,D. */
.m             /* on set */
4 5 6 8 9 10 13
.d             /* don't care set */
0 7 15
```

Output format:

(1) Generate all prime implicants to a file. The following is an example. If there are more than 20 primary implicants, report only the first 20 primary implicants with the least number of literals to reduce the file size. However, you still have to report the correct number of total primary implicants at ".p" field.

```
.i 4          /* the function f has 4 input variables: A,B,C,D. */
.m             /* on set */
4 5 6 8 9 10 13
.d             /* don't care set */
0 7 15
.p 7          /* there are 7 prime implicants */
0-00           /* A'C'D' */
-000           /* B'C'D' */
100-           /* AB'C' */
10-0           /* AB'D'*/
01--          /* A'B */
1-01           /* AC'D */
-1-1          /* BD */
```

(2) Generate the minimum sum-of-products expression to a file. The following is an example.

```
.mc 3          /* 3 prime implicants in minimum covering */
10-0           /* AB'D'*/
1-01           /* AC'D */
01--          /* A'B */
literal=8
```

Compile & Execute:

Compile command: $ g++ -std=c++11 <hw1.cpp>

Execute command: $ ./<execute file> <input file> <implicant1 file> <output file>

(Ex : $ ./a.out input1.txt implicant1.txt output1.txt)

Note that input and output file should be the arguments of program. Please make sure your code can be compiled and executed.

Grading:

Your grade depends on the correctness and runtime of your program. If several students generate minimum-cost results successfully, their grades are determined based on the ascending order of run time.

Program Submission:

1. Please use C++ language and your program must be written in only one source file.
2. Your source file must be named as "Student_ID_number_hw1.cpp" and please make sure that all characters of the filename are in lower case. For example, if your student number is 310510000, the name of your program file should be 310510000_hw1.cpp".
3. Do not print any words on terminal.
4. Please upload your source code, the output files for the benchmarks, and a simple report explaining the implementation details and your comments to the new E3 by deadline (Don't compress).
5. **Plagiarism is forbidden, otherwise you will get 0 point!!!**