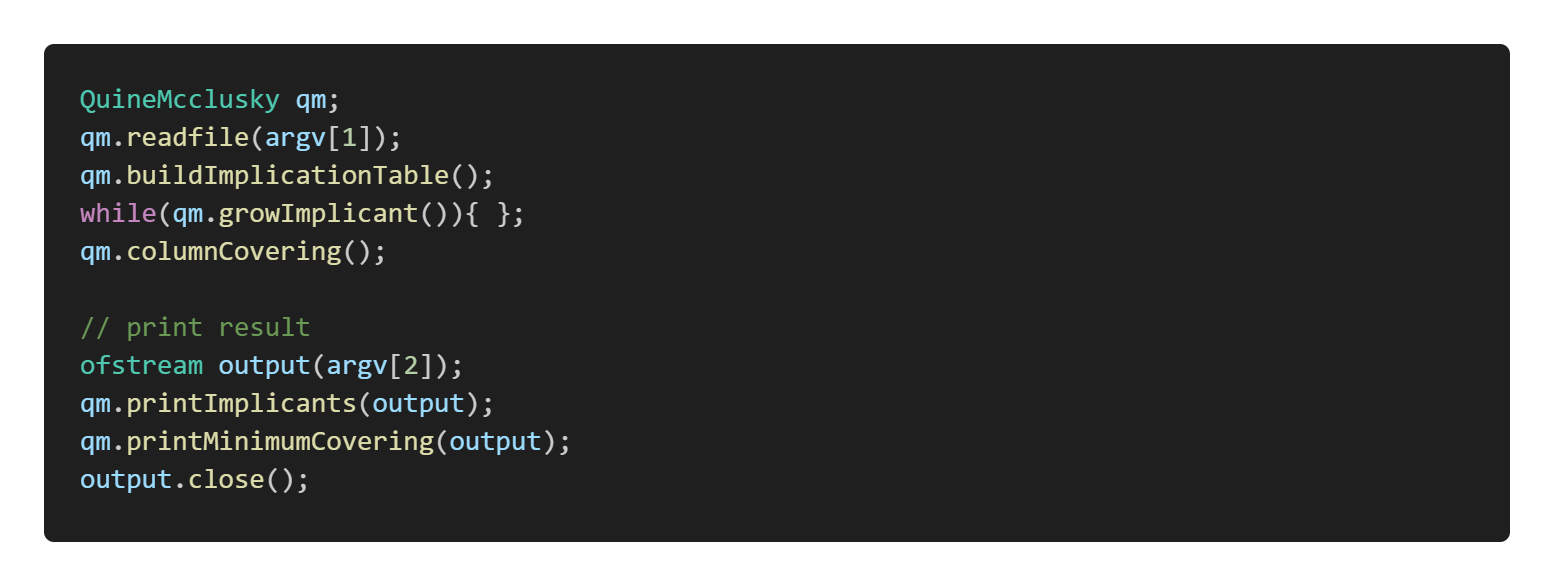
**Special Topics in Computer Aided Design**

**Lab1 Quine-Mccluskey Algorithm**

**312510224 林煜睿**

1. **Workflow**

Read input file Build Implication table Combine implicants and get *Prime Implicants* (Column Covering)Use *non-Prime Implicants* to do ***Petrick’s Method*** Print output



1. **Primary Implicant Generation**

First construct ***vector***<***list***<***Implicant***>>*implicationTable*, its index corresponding to the number of ones of implicant, the list stored inside it contains all implicants having same number of ones, where Data structure ***Implcant*** contains two values, ***string*** binary and ***int*** literal, which is its position in binary form and number of ones in its binary, repectively.

Then traverse *implicationTable* from index 0 to *implicationTable.size()*-2 (since index *implicationTable.size()*-1 has no implicants can combine with them). If current implicant can combine with implicant inside next layer, mark both of them combinable. Continuously executing until all implicants inside *implicationTable* are not combinable.

Finally construct an ***unordered\_map***<***int***, ***vector***<***string***>> *mp*, where its key and value are on-set position and prime implicants in binary form. Traverse *mp*to find all essential prime implicants and non- essential prime implicants, and use essential prime implicants to eliminate covered on-set, and we get *remainOnset*.

1. **Cover Remaining On-set**

By using remaining on-set we get in previous step, I construct a ***implicantCoverage*** table, its represent the minterms that current prime implicant covers.

Ex:

If there exist 4 minterms, a prime implicant covers the first, second and fourth of minterms, I stored it in integer form: 1+2+4 = 7 = 4’b1011.

Also, I store all literal of implicants inside ***vector***<***int***> *literalsCount*.

Then use dynamic programming to find the minimum cover. I construct the *dp* vector to record the best solution of corresponding on-set position.

Ex:

*dp*[3] = *dp*[4’b0011] = the minimum number of implicants that covers the first and second minterms.

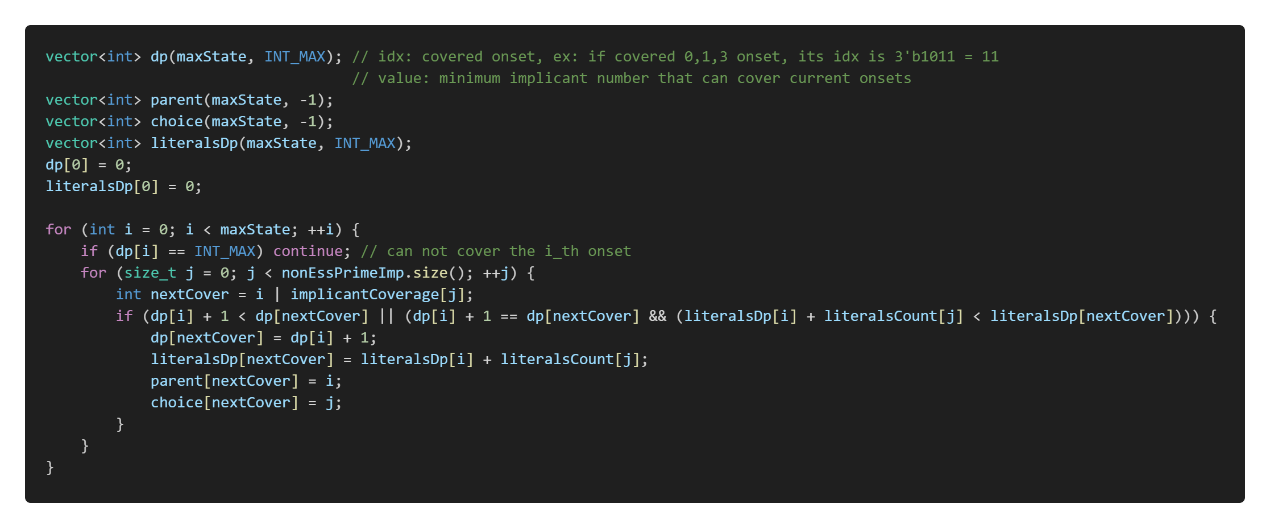
*dp*[12] = *dp*[4’b1100] = the minimum number of implicants that covers the third and fourth minterms.

I also create *parent* and *choice* vector to record the trace-information.

Ex:

If *parent*[5] = 3, since 5 = 4’b0101, 3 = 4’b0011, it means that to cover the first and third minterms(4’b0101), the best solution is generated from the implicant that covers the first and second minterms(4’b0011).

The *choice* vector records the implicant we choose to cover the current minterms.



Therefore, after executing the function, I can get minimum number of implicants to cover all minterms in *dp*[4’b1111] = *dp*[15]. Then trace back to find which implicant we choose to fulfill the answer by *parent* and *choice* vector.

