SPM HW1 GroupPart

I. 軟體專案開發的過程中，開發成員須經常召開會議，由於專案 80%的時間花在溝通，溝通的成效決定專案的成敗，故會議是軟體專案開發中重要且不可或缺的一環。本課程將每次作業模擬為真正的專案開發，希望同學將作業中的每題視為專案開發中的項目，透過小組會議分配作業的分工，使同學能對軟體專案的進行與管理有更深入的了解。

A. 如附件

B. 經討論後我們決定選擇 Microsoft Project

原因：

根據我們查詢到的資料發現 Azure DevOps 更像工程師團隊在開發專案時所使用的工具，與許多開發工具整合，例如：Visual Studio, Git, Jenkins，方便程式的自動化部屬、版本控管，並提供許多不同的開發方式包含 agile 開發、Scrum 開發。

另一方面 Microsoft Project 比較像 PM 所使用的工具，它包含更大方向的：項目規劃、排程、資源管理、甘特圖、項目追蹤等功能，雖然它與其他開發工具的整合有限，也比較只能使用傳統的 waterfall 方式，但我們覺得初入這個領域可以先從相對單純的工具開始使用，等未來我們更熟悉專案軟體流程後可以再學習更專注於細節的 Azure DevOps。

II. 請觀察資料夾 Program 中的三支程式，利用 3~4 種開源(OSS)工具，分別算出它們的圈複雜度，並附上過程的截圖與工具說明。如果得到的圈複雜度數值不同，請探討其可能原因。三人一組的組別使用 3 種開源工具即可，四人一組的組別則須使用 4 種開源工具。

以下為四位組員使用四種開源工具之結果：

- 張子姍：lizard (https://github.com/terryyin/lizard)

使用 Linux 的 lizard 工具，lizard 是一個輕量級的跨語言程式碼度量工具，支援 C++、C、Java、Python 等多種語言。它可以計算函數的 Cyclomatic Complexity、程式碼行數等。

安裝：*pip install lizard*

```
vpp@vpp-client:~/2024_SPM_HW1/Program$ sudo pip install lizard
[sudo] password for vpp:
Collecting lizard
  Downloading lizard-1.17.10-py2.py3-none-any.whl (66 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.0/66.0 KB 1.3 MB/s eta 0:00:00
Installing collected packages: lizard
Successfully installed lizard-1.17.10
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package ma
nager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

輸出結果的 CCN 為 Cyclomatic Complexity Number 的縮寫

1. 計算 B2_sequence.cpp 的圈複雜度: CC = 7

```
vpp@vpp-client:~/2024_SPM_HW1/Program$ lizard B2_sequence.cpp
================================================
  NLOC    CCN   token  PARAM  length  location
------------------------------------------------
      33      7    192      0      40 main@4-43@B2_sequence.cpp
1 file analyzed.
==============================================================
NLOC    Avg.NLOC  AvgCCN  Avg.token  function_cnt    file
--------------------------------------------------------------
     36      33.0     7.0      192.0         1     B2_sequence.cpp

=================================================================================================================
No thresholds exceeded (cyclomatic_complexity > 15 or length > 1000 or nloc > 1000000 or parameter_count > 100)
=================================================================================================================
Total nloc   Avg.NLOC  AvgCCN  Avg.token   Fun Cnt  Warning cnt   Fun Rt   nloc Rt
-----------------------------------------------------------------------------------------------
        36      33.0     7.0      192.0         1            0      0.00    0.00
```

2. 計算 Cryptanalysis.cpp 的圈複雜度: CC = 11

```
vpp@vpp-client:~/2024_SPM_HW1/Program$ lizard Cryptanalysis.cpp
================================================
  NLOC    CCN   token  PARAM  length  location
------------------------------------------------
      33     11    239      0      43 main@4-46@Cryptanalysis.cpp
1 file analyzed.
==============================================================
NLOC    Avg.NLOC  AvgCCN  Avg.token  function_cnt    file
--------------------------------------------------------------
     35      33.0    11.0      239.0         1     Cryptanalysis.cpp

=================================================================================================================
No thresholds exceeded (cyclomatic_complexity > 15 or length > 1000 or nloc > 1000000 or parameter_count > 100)
=================================================================================================================
Total nloc   Avg.NLOC  AvgCCN  Avg.token   Fun Cnt  Warning cnt   Fun Rt   nloc Rt
-----------------------------------------------------------------------------------------------
        35      33.0    11.0      239.0         1            0      0.00    0.00
```

3. 計算 Fibonaccimal_Base.cpp 的圈複雜度: CC = 7

```
vpp@vpp-client:~/2024_SPM_HW1/Program$ lizard Fibonaccimal_Base.cpp
================================================
  NLOC    CCN   token  PARAM  length  location
------------------------------------------------
      29      7    207      0      40 main@4-43@Fibonaccimal_Base.cpp
1 file analyzed.
==============================================================
NLOC    Avg.NLOC  AvgCCN  Avg.token  function_cnt    file
--------------------------------------------------------------
     31      29.0     7.0      207.0         1     Fibonaccimal_Base.cpp

=================================================================================================================
No thresholds exceeded (cyclomatic_complexity > 15 or length > 1000 or nloc > 1000000 or parameter_count > 100)
=================================================================================================================
Total nloc   Avg.NLOC  AvgCCN  Avg.token   Fun Cnt  Warning cnt   Fun Rt   nloc Rt
-----------------------------------------------------------------------------------------------
        31      29.0     7.0      207.0         1            0      0.00    0.00
```

- 王彥翔：cccc (https://github.com/sarnold/cccc)

使用 cccc 工具，cccc 可以用於 Linux, POSIX-style 與 Win32，並可以分析 C、

C++、Java。其中包 LOC、CC、comment Lines 等，最後以 html 形式生成報表。

安裝：參考作者在 github 上的說明

```

*sudo apt-get install software-properties-common*

*sudo add-apt-repository -y -s ppa:nerdboy/embedded*

*sudo apt-get install cccc*

```

使用方法：cccc <options> <filename>

使用 cccc 時會自動生成.cccc 資料夾並把相關分析文件放入其中，為了避免衝突我們使用--ourdir=<dirname>的選項為三個程式建立各自的資料夾

MVG: McCabe's Cyclomatic Complexity

1. 計算 B2_sequence.cpp 的圈複雜度: MVG = 7

Function prototype	LOC	MVG	COM	L_C	M_C
main( ) definition B2_sequence.cpp:4	32	7	0	******	******

2. 計算 Cryptanalysis.cpp 的圈複雜度: MVG = 11

Function prototype	LOC	MVG	COM	L_C	M_C
main( ) definition Cryptanalysis.cpp:4	32	11	0	******	******

3. 計算 Fibonaccimal_Base.cpp 的圈複雜度: MVG = 7

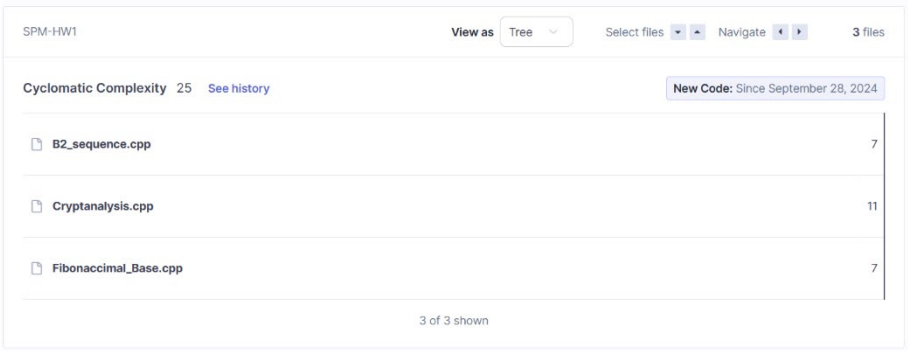
Function prototype	LOC	MVG	COM	L_C	M_C
main( ) definition Fibonaccimal_Base.cpp:4	28	7	0	******	******

- 李銘峰：SonarQube (https://docs.sonarsource.com/sonarqube/latest/)

使用 SonarQube，是一種本地分析工具，旨在檢測 30+ 種語言、框架和 IaC

平臺中的編碼問題。

1. 架設 SonarQube Server
   1.1. POSTGRES DB
   ```
   docker run -d --name sonarqube-db -e POSTGRES_USER=sonar -e POSTGRES_PASSWORD=sonar -e POSTGRES_DB=sonarqube postgres:alpine
   ```
   1.2. SonarQube Server
   ```
   docker run -d --name sonarqube -p 9000:9000 --link sonarqube-db:db -e SONAR_JDBC_URL=jdbc:postgresql://db:5432/sonarqube -e SONAR_JDBC_USERNAME=sonar -e SONAR_JDBC_PASSWORD=sonar sonarqube
   ```
2. 使用 sonar-scanner 掃描專案，取得參數。
   ```
   sonar-scanner \
     -Dsonar.projectKey=SPM-HW1 \
     -Dsonar.sources=. \
     -Dsonar.host.url=http://localhost:9000 \
     -Dsonar.token=sqp_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
   ```
3. 從 https://localhost:9000/ 取得分析結果。



- 趙承彥：Cyclo(https://github.com/sarnold/cyclo)

使用開源工具 Cyclo，Cyclo 透過靜態分析計算程式的圈複雜度，無需執行程式碼且支援多種語言，包括 C、C++、Python 和 Go。

git clone https://github.com/sarnold/cyclo.git

cd cyclo

```
ubuntu@ubuntu:~/cyclo$ ./mcstrip /home/ubuntu/B2_sequence.cpp | ./cyclo -c
intminn 7

Total complexity:    7
ubuntu@ubuntu:~/cyclo$ ./mcstrip /home/ubuntu/Cryptanalysis.cpp | ./cyclo -c
intmann 11

Total complexity:    11
ubuntu@ubuntu:~/cyclo$ ./mcstrip /home/ubuntu/Fibonaccimal_Base.cpp | ./cyclo -c
intmann 7
```

B2_sequence 的圈複雜度: CC = 7

Cryptanalysis 的圈複雜度: CC = 11

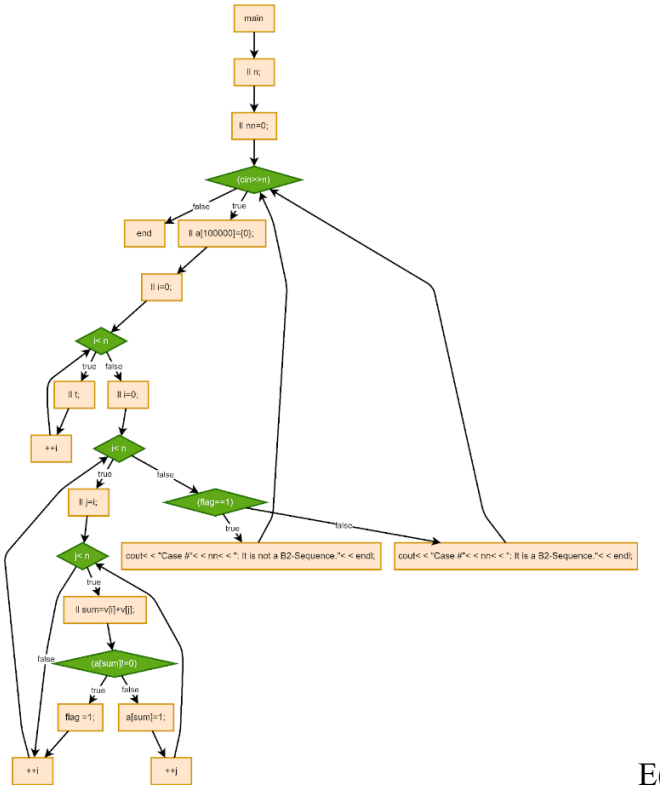
Fibonaccimal_Base 的圈複雜度: CC = 7

III. 承上題，依據上一題得出流程圖並參考 Chapter02 講義 p.71~p.73 的範例，分別計算三支程式的圈複雜度。請附上流程圖並列出詳細的公式、計算過程(多個 Module 的計算方法可參考 Chapter02 講義 p.82~p.83 Modified Cyclomatic Complexity Measures)。若算出來的圈複雜度與上題有所差異的話亦請探討其可能原因。

以下為使用不同工具所得流程圖：

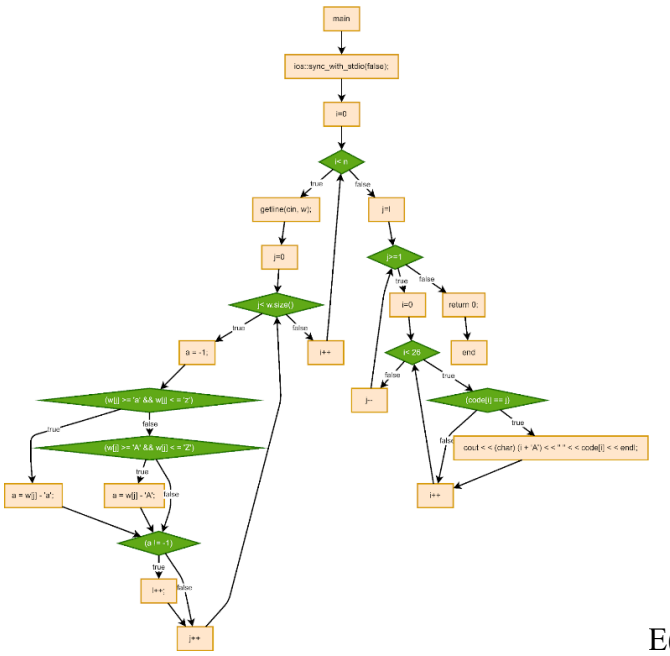
- 張子姍：https://debug996.com/draw/draw.html

1. B2_sequence.cpp:



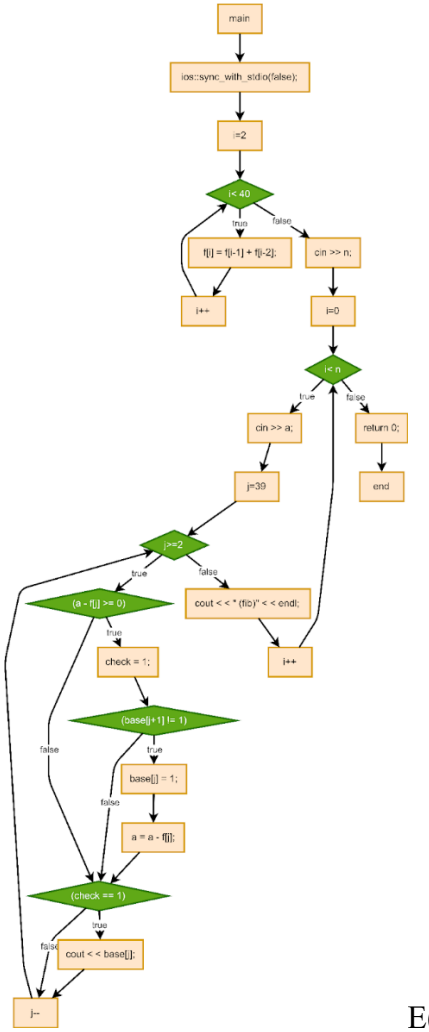
E(28)-N(23)+2*P(1) = 7

2. Cryptanalysis.cpp:



E(33)-N(26)+2*P(1) = 9

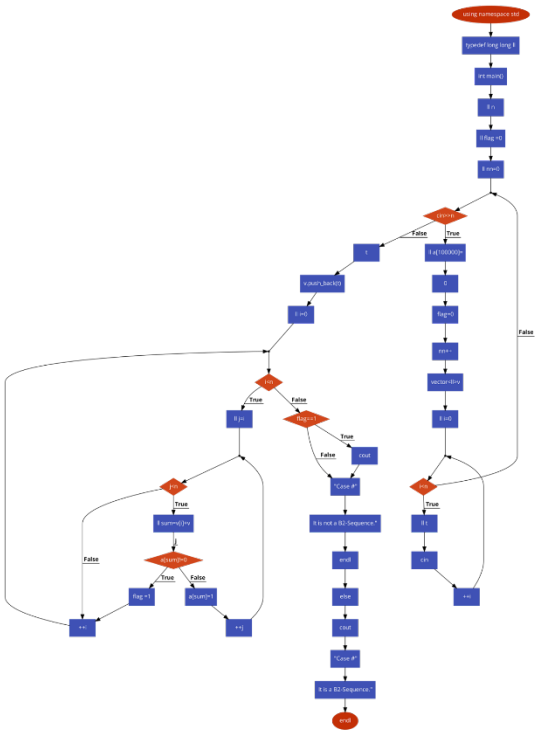
3. Fibonaccimal_Base.cpp:



E(29)-N(24)+2*P(1) = 7

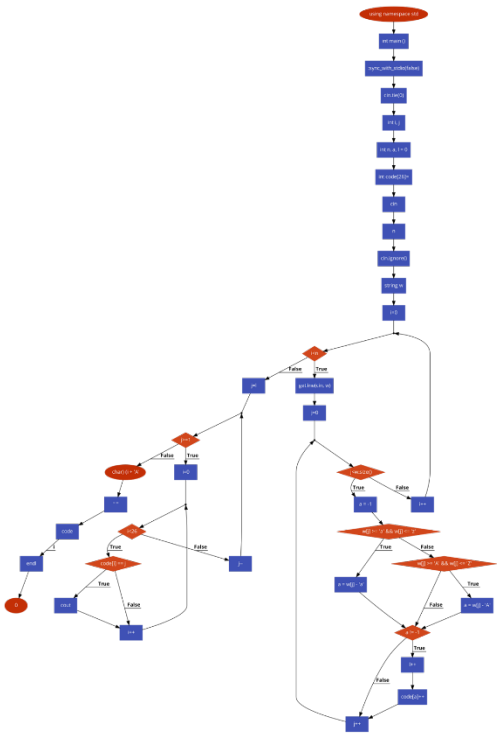
- 王彦翔：https://code2flow.com/

1. B2_sequence.cpp:



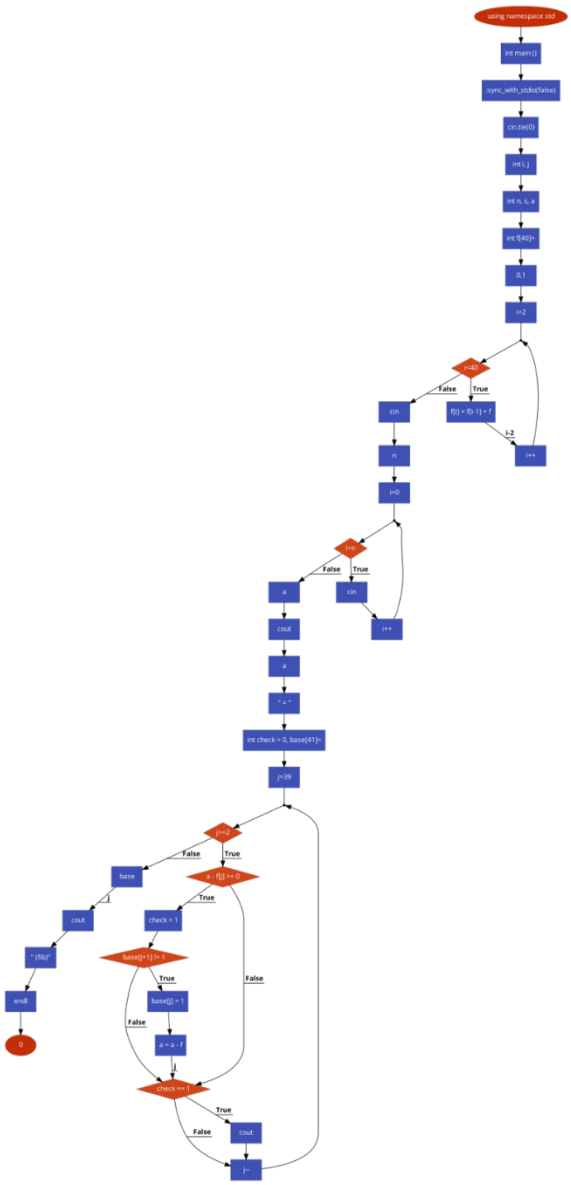
E(44)-N(39)+2*P(1) = 7

2. Cryptanalysis.cpp:



E(46)-N(39)+2*P(1) = 9

3. Fibonaccimal_Base.cpp:



E(43)-N(38)+2*P(1) = 7

討論：

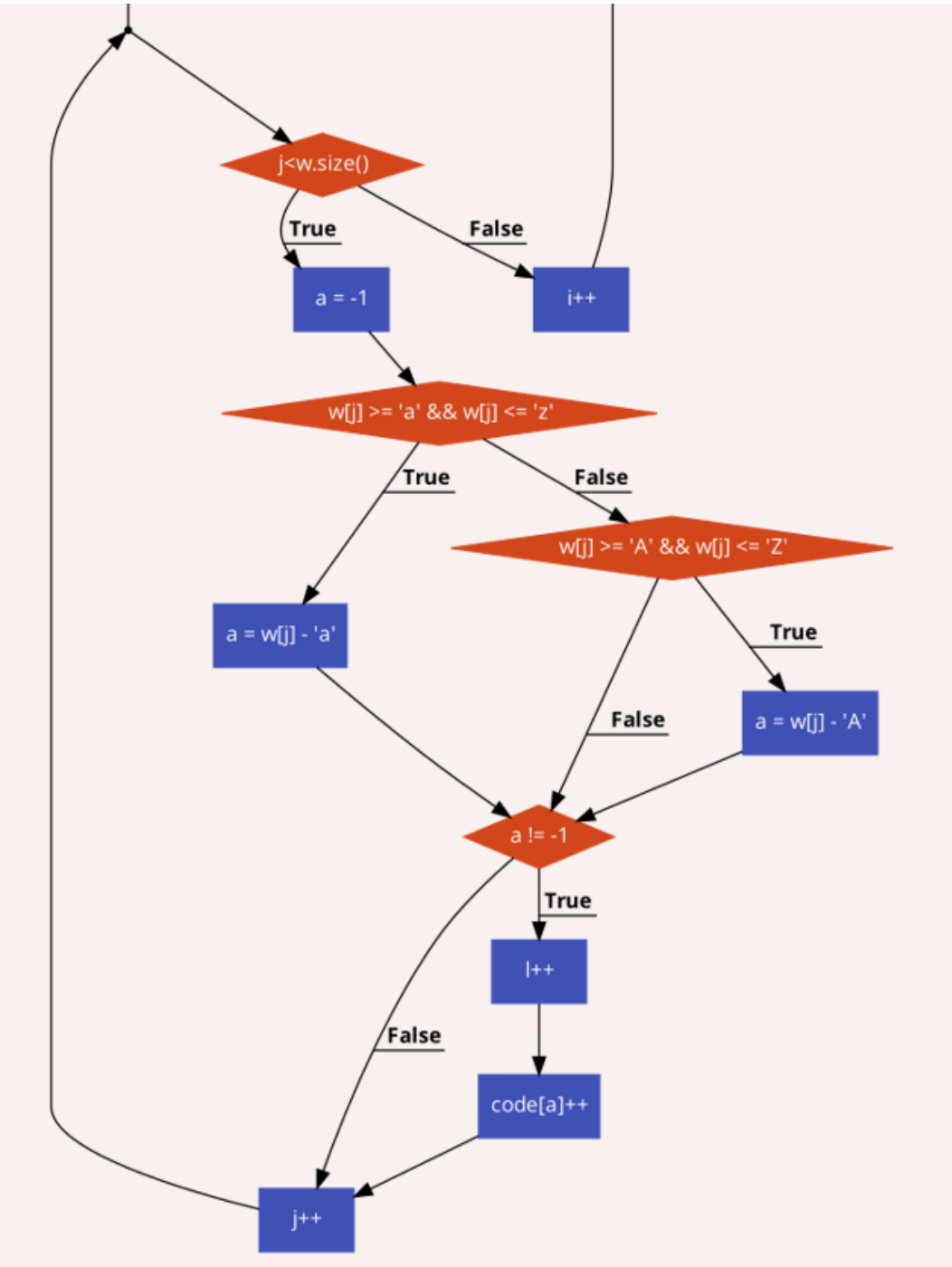
1. 四位組員分別用各自的分析軟體跑出來的結果皆相同。

2. 根據繪製出來的流程圖並搭配 e-n+2p 得出來的結果發現 Cryptanalysis 算出來的結果與分析程式算出來的結果不同，一開始我們根據講義中 ch2p57 有提到複雜度>10 可能算出來的結果不一定準確，推測應該是因為 Cyptanalysis 已經達到一定複雜度使得不同分析軟體的結果與流程圖的結果不同。後來翻閱分析程式的算法會把&&的前後條件拆分成 2 個 node 而流程圖並沒有，因此重新繪製流程圖後成功得出複雜度也是 11。至於 B2 和 Fib

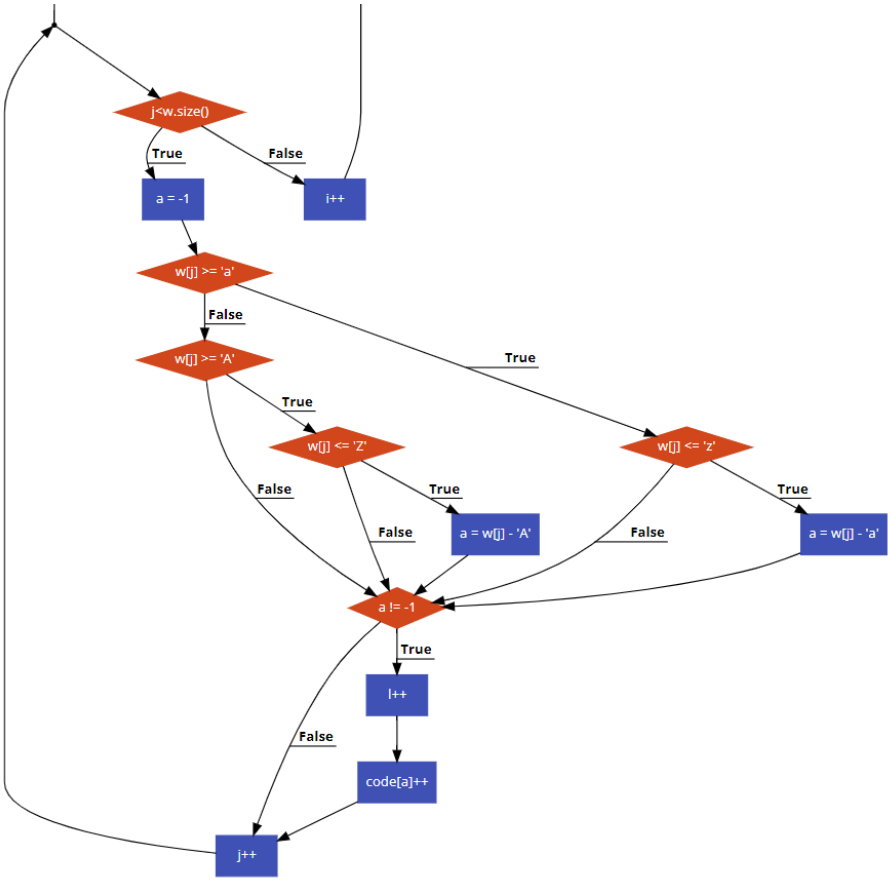
無此問題是因為程式當中並沒有&&或||。

- McCabe's Cyclomatic Complexity(MVG)
  The formal definition of cyclomatic complexity is that it is the count of linearly independent paths through a flow of control graph derived from a subprogram. A pragmatic approximation to this can be found by counting language keywords and operators which introduce extra decision outcomes. This can be shown to be quite accurate in most cases. In the case of C++, the count is incremented for each of the following tokens: 'if','while','for','switch','break','&&','||'

修改前：



修改後：



參考資料:

https://kaelzhang81.github.io/2017/06/18/%E8%AF%A6%E8%A7%A3%E5%9C%88%E5%A4%8D%E6%9D%82%E5%BA%A6/

IV. GQM 是一種專案開發方法，必須先針對專案設立目標 (Goal)，並針對團隊所設立的目標設想開發過程可能遇到的問題 (Question)，再針對該問題提出數項指標 (Metrics) 作為度量方法，請同學根據下面敘述:(1)共享雨傘軟體系統 (2)食物外送平台，閱讀後與組員討論並從中選擇一主題，並使用 GQM 開發方法完成下列 A、B、C 小題。

A. 共享雨傘軟體流程圖設計

設計說明：

我們團隊經討論後選擇共享雨傘軟體系統為主題，此共享雨傘軟體參考 Raingo app 設計並改進。一進入 app 先檢查是否已登入過帳號，根據此條

件進入主頁畫面或確認是否初次使用，若為初次使用則進入註冊與綁定支付工具頁面，非首次使用則進入登入畫面並進入主頁。

主頁中一共有三大功能：

1. 選單

包含使用說明、個人資料編輯、過去租借紀錄查詢、優惠券查詢、設定付款方式、聯絡客服稍後會詳細介紹。

聯絡客服：

依據我們觀察 Raingo app 的客服使用方式，都會跳轉到 Line 才能使用客服功能，但我們考慮到有些使用者不一定有 Line，例如:手機中只有微信，因此有 app 內建的客服功能應該可以更好地被使用者使用。

2. 借還傘

在借還傘的介面中，首先判斷使用者是否正在借用雨傘，若否則顯示「掃描 QR code」按鈕，使用者點擊後將開啟鏡頭供使用者掃描雨傘上的 QR code，成功掃描後即可進入付款方式選擇並選取可用優惠券，待全數確認完畢即可點擊「確認租借」；在前 5 分鐘使用者可以先確認租借的雨傘是否有問題，有問題可以點擊 app 中的問題回報功能回報狀況，並歸還有問題的雨傘，確認無誤則完成借傘流程。

若使用者有正在借用中的雨傘，系統會先確認此次借用時間是否已經超過 14 天，若是則跳轉到逾期買斷流程。若使用時間小於 14 天則系統會跳至顯示時長與費用的頁面，供使用者及時確認目前的費用。最後當使用者抵達還傘站點可以點擊「我要還傘」，系統將計算此次借用時間與費用，扣除借傘時使用者使用的優惠券從使用者當時選得的付款方式扣除款項，即還傘成功。

3. 站點查詢

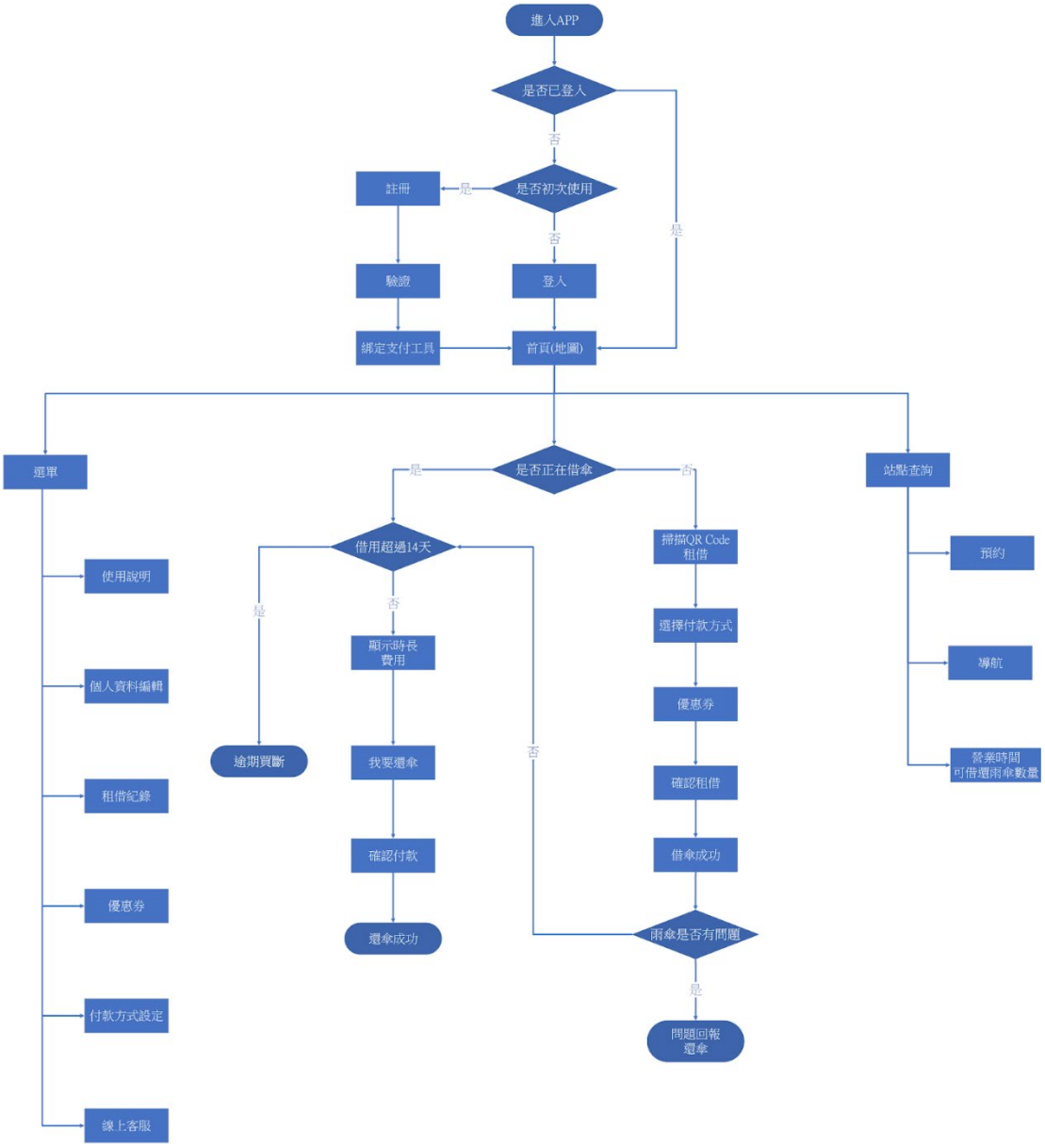
包含內建導航到對應站點、各站點營業時間與可借還傘確認、預約借傘稍後會詳細介紹。

預約借傘：

考慮到使用者目前的位置可能離欲借傘的站點有段距離，為消弭使用者借不到傘的擔心我們提供了預約借傘的功能，使用者可以在地圖上點選欲借傘之站點，並點選預約借傘，系統就會免費為使用者保留 30

分鐘，如果逾時未取消系統則會從第 31 分鐘開始計算收費。

流程圖：



B. Goal: 提高共享雨傘系統的使用者滿意度和營運效率

Subgoal 1: 提升使用者滿意度

Q1:使用者對租借流程的滿意度如何？

Metric 1.1：點下「我要還傘」到完成還傘平均完成時間（秒）

Metric 1.2：使用者對借還流程的滿意度評分(1-5 分)

Metric 1.3：每月收到的關於借還流程的投訴數量

Q2:使用者對 App 介面的使用體驗如何?

Metric 2.1：App 介面的使用者滿意度評分(1-5 分)

Metric 2.2：完成借還操作的平均點擊次數

Metric 2.3：App 使用過程中的錯誤發生率

Metric 2.4：App 平均使用時間

Q3:使用者對服務價格的滿意程度為何？

Metric 3.1：使用者對計時租借價格合理性的滿意度評分 (1-5 分)

Metric 3.2：使用者對月租租借價格合理性的滿意度評分 (1-5 分)

Metric 3.3：使用者對租後買斷價格合理性的評分(1-5 分)

註：買斷指逾期未還超過一定天數之費

Q4:使用者對於此服務的滿意度

Metric 4.1：每月有多少新客戶

Metric 4.2：有多少客戶會推薦此服務給親友

Metric 4.3：用戶回頭率(完成兩次以上交易的用戶/總用戶)

Q5:使用 App 服務之體驗

Metric 5.1：對於線上客服的滿意度(1-5 分)

Metric 5.2：地圖反應時間(s)

Metric 5.3：每人的平均註冊時間、付款方式設定時間

Metric 5.4：付款成功比(付款成功次數/總付款次數)

Subgoal 2: 提升營運效率

Q1:系統的維護成本如何?

Metric 1.1：系統維護人員的工作效率 (每小時處理的任務數)

Metric 1.2：系統故障的平均修復時間(小時)

Metric 1.3：單日租借量

Q2:租借站點設置在哪邊？

Metric 1.1：站點每月借還次數

Metric 1.2：一年內午後雷陣雨天數

Metric 1.3：一年內梅雨季天數

Metric 1.4：使用者對於站點分布的滿意度(1-5 分)

Q3:雨傘的使用狀況

Metric 3.1：雨傘平均使用天數

Metric 3.2：定期雨傘品質檢查中發現的問題雨傘比例

Metric 3.3：每個月有多少雨傘逾期未還

Metric 3.4：每個月有多少雨傘失竊

Q4:站點使用頻率？

Metric 4.1：每月無法租還的次數

Metric 4.2：租還總次數

Metric 4.3：不同氣候的站點使用率

Q5:系統的收益狀況如何?

Metric 5.1：月平均營收

Metric 5.2：每把雨傘的日均收益

Metric 5.3: 每個站點的平均營收

C. GQM 7 步驟的自動化程度

1. 設定目標

設定目標需要根據業務上的需求或組織的策略，與團隊合作進行規劃與設計，所以無法完全自動化。不過，可以使用開源工具或專案管理系統來記錄與追蹤目標設定的過程。

可自動化占比：10%

原因：工具主要用於協助目標的文件化和共享資料。

2. 設定問題

設計與目標相關的問題，同樣依賴團隊的分析能力和對業務的深入理解，所以無法完全自動化。不過，這一階段相比前一個步驟，需要更多專業的軟體專案管理技能參與。同時，過去的經驗和案例也可以作為寶貴的參考資料。

可自動化占比：20%

原因：此步驟需人工定制具體問題，需要根據目標進行深入思考和設計，自動化工具在此僅輔助設計，像是提供範例或問題模板。

3. 設定衡量標準

大部分需依靠 PM 的經驗以及專業能力設定，此部分相較前兩個步驟所需討論程度較低，可以利用 AI 或自動化工具範本協助訂定衡量標準。

可自動化占比：30%
原因：或許 AI 工具可以根據問題生成具體的衡量項目，但人為經驗是無法複製取代的，且每種產品案例在現實中存在不同的變數。

4. 設計資料蒐集與分析的方法

設計具體的資料收集方式，確保度量標準和資料的正確性和一致性，並選擇適合的統整工具和分析方法。

可自動化占比：50%
原因：在此可以使用 AI 工具，依制定的標準去設計資料蒐集方法，交叉比較不同種類的工具及其可行性，但還是需要人為審視評估最後結果，較為完善。

5. 蒐集驗證分析資料並採取行動

透過已經設計好的方法與軟體可以自動生成數據報表，後續再透過經驗者根據報表思考所應採取的行動。

可自動化程度：70%
原因：僅需依照所設計方法實作蒐集和驗證，自動化工具能大力發揮所長，因此更能實現高度自動化。

6. 事後分析

同樣使用第四步所構建的方法進行程式自動化分析產生事後分析報表，並與之前第五步所產生的數據報表進行交叉分析產生交叉分析報表，即可得知此次行動的成效。

可自動化程度：50%
原因：資料工具可以自動化生成報告，但具體深度解釋仍需人工探討。

7. 提供報告反饋給關係人

根據分析資料解釋結果，提出具體的改進建議，幫助系統或產品達成目標。這些建議可以涉及技術改進、流程優化或資源分配等變更。

可自動化占比：50%
原因：取決分析報表彙整給關係人的形式，若以圖表形式表示，可以使用 AI 工具生成相關資料作為書面報告，進行口頭報告或是實際與會報告時，便需要人為進行。

V. 請同學用 Group Part I.設計好的範本記錄本次會議(如：群組題 I、II…、V 等小組內的意見與討論)，須至少包含會議日期、時間、地點、參與成員、討論事項、作業分工列表、組員討論合照。會議紀錄最多 5 頁。

如附件

Reference

A. Microsoft projects v.s. Azure DevOps: https://stackshare.io/stackups/azure-devops-vs-microsoft-project

B. 圈複雜度計算：https://kaelzhang81.github.io/2017/06/18/%E8%AF%A6%E8%A7%A3%E5%9C%88%E5%A4%8D%E6%9D%82%E5%BA%A6/

C. cccc 圈複雜度計算邏輯：https://sarnold.github.io/cccc/CCCC_User_Guide.html

D. Sonar 圈複雜度計算邏輯：https://docs.sonarsource.com/sonarqube/latest/user-guide/code-metrics/metrics-definition/