# Data Structure Assignment 1

|  |  |  |
| --- | --- | --- |
| **ID:** E14066282 | **Name:** 溫梓傑 | **Department:** ME 110 |

## Result Screenshots



Figure Screenshot of command line

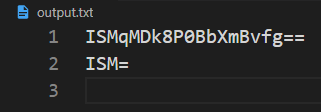
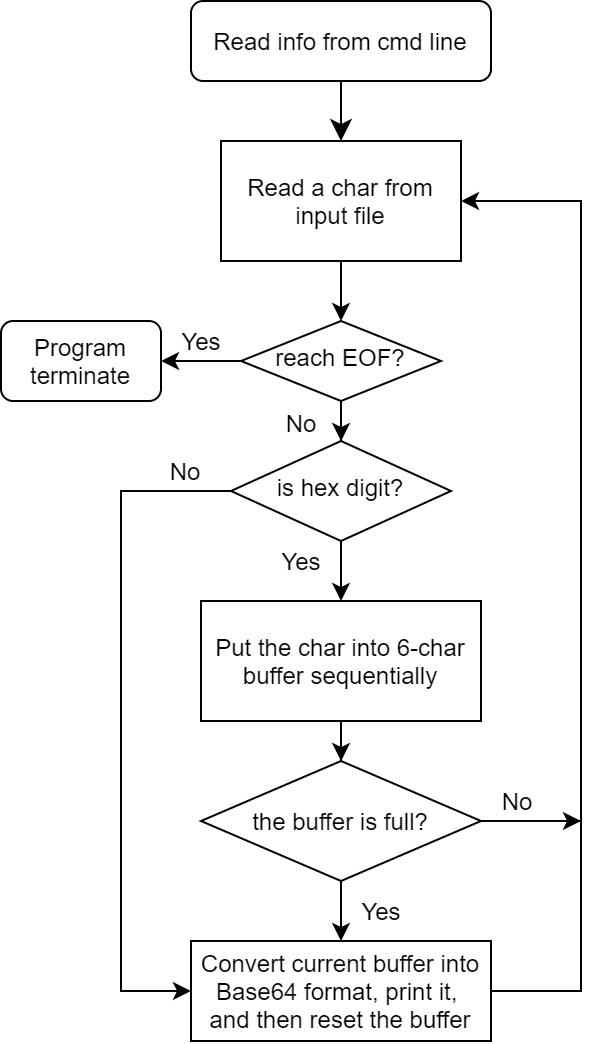


Figure Screenshot of “output.txt”

## Program Architecture



## Program Functions



Parameters

|  |  |
| --- | --- |
| * argc | 命令列參數個數 |
| * argv | 命令列輸入文字 |
| * optstring | 解析字元陣列 |

Return Value

* If the option takes a value, that value is pointer to the external variable optarg**.**
* ‘-1’ if there are no more options to process.
* ‘?’ when there is an unrecognized option and it stores into external variable optopt**.**
* If an option requires a value (such as -f in our example) and no value is given, getopt normally returns ‘?’. By placing a colon as the first character of the options string, getopt returns: instead of ‘?’ when no value is given.



Parameters

|  |  |
| --- | --- |
| * argc | 命令列參數個數 |
| * argv | 命令列輸入文字 |
| * in | 讀取檔案名稱 |
| * out | 輸出檔案名稱 |



Parameters

|  |  |
| --- | --- |
| * ch | 轉換字元 |

Return Value

當ch為hex digit (0-9, a-f, A-F)，則轉換為整數輸出；

若否，則輸出-1。



Parameters

|  |  |
| --- | --- |
| * ch | 當前讀取檔案之字元 |
| * str | buffer字元陣列指標 |

## Program Design

### Base64 Conversion

將六個hex digits存入一個int中，並透過bitwise operation、shift operation來完成轉換，詳見以下解說：

There are six conditions when converting as following:

1. **6** valid hex digits ⇒ output **4** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | | D | | | | 6 | | | | 1 | | | | 6 | | | | E | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| **Base64 encoded** | **Sextets** | 19 | | | | | | 22 | | | | | | 5 | | | | | | 46 | | | | | |
| **Character** | T | | | | | | W | | | | | | F | | | | | | u | | | | | |

b. **5** valid hex digits ⇒ output **4** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | | D | | | | 6 | | | | 1 | | | | 6 | | | |  | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Base64 encoded** | **Sextets** | 19 | | | | | | 22 | | | | | | 5 | | | | | | 32 | | | | | |
| **Character** | T | | | | | | W | | | | | | F | | | | | | g | | | | | |

c. **4** valid hex digits ⇒ output **3** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | | D | | | | 6 | | | | 1 | | | |  | | | |  | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |  |  |  |  |  |  |
| **Base64 encoded** | **Sextets** | 19 | | | | | | 22 | | | | | | 4 | | | | | |  | | | | | |
| **Character** | T | | | | | | W | | | | | | E | | | | | | = | | | | | |

d. **3** valid hex digits ⇒ output **2** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | | D | | | | 6 | | | |  | | | |  | | | |  | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |
| **Base64 encoded** | **Sextets** | 19 | | | | | | 22 | | | | | |  | | | | | |  | | | | | |
| **Character** | T | | | | | | W | | | | | | = | | | | | | = | | | | | |

e. **2** valid hex digits ⇒ output **2** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | | D | | | |  | | | |  | | | |  | | | |  | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |
| **Base64 encoded** | **Sextets** | 19 | | | | | | 16 | | | | | |  | | | | | |  | | | | | |
| **Character** | T | | | | | | Q | | | | | | = | | | | | | = | | | | | |

f. **1** valid hex digits ⇒ output **1** characters

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hex digits** | | 4 | | | |  | | | |  | | | |  | | | |  | | | |  | | | |
| **Bits** | | 0 | 1 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Base64 encoded** | **Sextets** | 16 | | | | | |  | | | | | |  | | | | | |  | | | | | |
| **Character** | Q | | | | | | = | | | | | | = | | | | | | = | | | | | |

Hence, a table created as following:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **# valid hex digits** | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| **# output characters** | 0 | 1 | 2 | 2 | 3 | 4 | 4 |

which is corresponding to



The source code about Base64 conversion is shown below.



## Operating System

Windows 10

## Compiler

(MinGW.org GCC Build-20200227-1) 9.2.0

## Compile

gcc -std=c11 -o compressor compressor.c

## Run

./compressor -i input.txt -o output.txt