解説など

1. ディレクトリ構成

|- in　：入力データ
|- out　：プログラム(exam5)を実行した結果
|-src　：ソースコード
|-Class.png　ちょっとしたクラス図

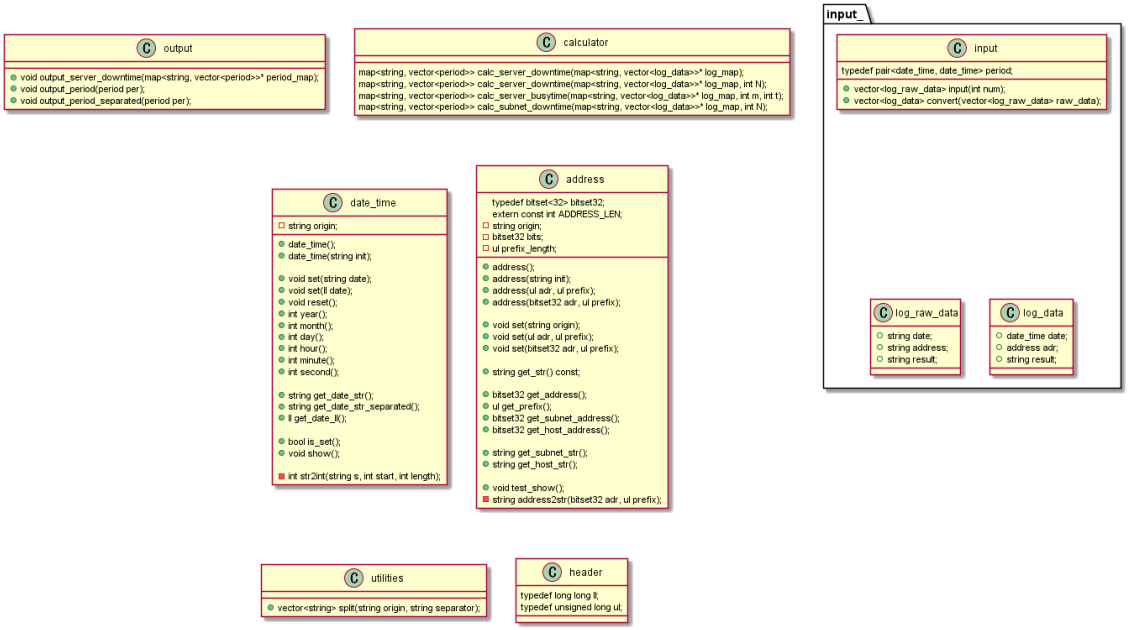

図 1. Class.png

2. 実行方法

src ディレクトリ下で実行する。makefile を書いたので、”make exam1”などでコンパイルできる。また、それぞれの設問に対応するプログラムは exam3 のようになっている。実行方法の例を以下に記載する。

設問 1: ./exam1 < ../in/test02.txt
　　exam1
設問 2: ./exam2 3 < ../in/test02.txt
　　exam2 [N]
設問 3: ./exam3 2 100 < ../in/test02.txt
　　exam3 [m] [t]

設問 4: ./exam4 2 < ../in/test02.txt

exam4 [N]

(設問 5): ./exam5 3 2 100< ../in/test02.txt

exam5 [N] [m] [t]

ただし、exam5 は exam1~exam4 の内容すべてを含むプログラムである。実行時のパラメータは正の整数を仮定しており、文字列は一応実行時エラーになる。

3. 構成説明

大まかに表 1 のようになっている。

表 1 各ファイルの説明と役割

ファイル名	説明・役割
address.cpp	アドレスを扱うためのクラス
calculator.cpp	ログデータから実際に故障期間を求める
date_time.cpp	日付を扱うためのクラス
exam1.cpp ~ exam5.cpp	各設問に対応する実行部分
input.cpp	入力・クラスへの変換
output.cpp	出力
utilities.cpp	文字列を切り出す機能

各 exam.cpp の中では以下のような流れで処理が行われている。

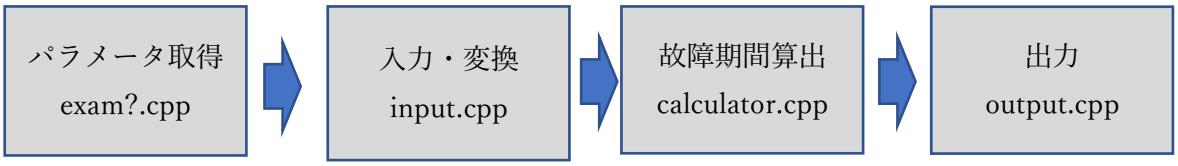

図 2. フロー図

変換では day_time, address クラスを用いた表現にログデータを変換している。その後アドレスごとにログデータを振り分け、各サーバについて故障期間を算出した。設問 1,2,3 は流れが似通っているが、設問 4 のみ故障期間算出の中身が異なっている。

具体的には再度サーバをサブネットごとに振り分けている。設問 4 はサーバ全てが故障している期間を求めるため、サブネットごとの処理とすると具合がいいためである。サブネットごとにさえしてしまえば共通区間を求める問題に帰着でき時間をキーにした連想配列と累積和で解くことができる。

4. 使用したテストデータと実行結果

使用したテストデータは in ディレクトリに、実行結果は out ディレクトリに格納している。ただし実行結果の各データの先頭は実行時のパラメータが記入されている。

各テストケースの

ファイル名	狙い・特徴
test01.txt	複数行の読み込みができるか
test02.txt	複数行の読み込み 単一のタイムアウトを検出できるか
test03.txt	複数・連続のタイムアウト 末尾がタイムアウトで終わる場合を 処理できるか
test04-1.txt	単一のサブネットに属する二つのサーバ
test04-2.txt	test04-1.txt の二倍の長さ
test05-1.txt	単一のサーバ ping の結果が長い部分がある
test05-2.txt	test05-1.txt の二倍の長さ
test06.txt	二つのサブネット サブネットの故障が検出できるか
test07.txt	日付が遠く離れていても処理できるか
test08.txt	結果がタイムアウトのみの場合

感想になってしまうが、もう少し時間があれば自動でテストケースを生成したり故障期間をエクセルなどで分かりやすく可視化をしたかったと思った。