

プログラミング B レポート

課題 3

担当教員: 武政 淳二

大坪 祐清

学籍番号: 09B25014

2025 年 12 月 30 日

1 課題内容

以下に挙げる 2 種類の検索を実現するプログラムを作成し、レポートにまとめよ。

(検索 1) 郵便番号を入力すると、対応する住所を出力する。

(検索 2) 住所名の一部を表す文字列を入力すると、その文字列を含む一連の住所およびそれらに対応する郵便番号のリストを出力する。ただし、リストは郵便番号で昇順にソートすること。

ただし、プログラムは以下の 2 段階の処理を実行すること。

1. 前処理

検索を容易にするための前処理を担当する。郵便番号と住所を含むテキストファイルを入力として、構造体を利用してメモリに格納する。

2. 検索処理

(検索 1) もしくは (検索 2) を処理し、検索結果を画面に出力する。

なお、入力のテキストファイルは、以下の、日本郵政株式会社の Web ページ^{*1}からダウンロードした csv ファイルとする。

csv ファイルの各列の意味は、表 1 で示す。ただし、本課題で扱わない 10 列目以降は省略した。

2 アルゴリズムの説明

以下の手順で、処理を進める。

^{*1} http://www.post.japanpost.jp/zipcode/dl/kogaki/zip/ken_all.zip

表 1: csv ファイルの各列の意味 (9 列目まで)

1 列目	全国地方公共団体コード (JIS X0401、X0402)
2 列目	旧郵便番号 (5 桁)
3 列目	郵便番号 (7 桁)
4 列目	都道府県名 (半角カタカナ)
5 列目	市区町村名 (半角カタカナ)
6 列目	町域名 (半角カタカナ)
7 列目	都道府県名 (漢字)
8 列目	市区町村名 (漢字)
9 列目	町域名 (漢字)

1. csv ファイルの 1 行目からファイルの終わりまで以下の処理を繰り返す.
 - (a) csv ファイルを読み込み, 3 行目, 7 行目, 8 行目, 9 行目を変数に読み込む.
 - (b) 読み込んだ変数を, 構造体配列に読み込む.
2. (検索 1) と (検索 2) のどちらの検索をしたいのか, あるいはプログラムを終了したいのか, ユーザーからの入力を受け付ける.
3. (検索 1) あるいは (検索 2) をユーザーが望んだ場合, 検索したい郵便番号ないしは住所の入力を受けつけ, 変数として受け取る.
4. 入力に応じて条件分岐を行う
5. (検索 1) の場合の処理
 - (a) クイックソートを用いて, 郵便番号が昇順になるように構造体配列をソートする.
 - (b) ユーザーからの郵便番号の入力と, ソートされた構造体配列をもとに, 二分探索法で目的の住所を探す.
 - (c) 条件にあったデータがあったら, 郵便番号と住所を出力する.
6. (検索 2) の場合の処理
 - (a) $n=1$ から $n=「1$ で読み込んだ行数」まで, 以下の処理を繰り返す
 - i. n 行目の都道府県名, 市区町村名, 町域名のいずれかに入力の文字列が含まれていたら, 配列に n を追加する.
 - ii. 6 (a) i の操作でいずれにも, 入力が含まれていなかったかつ, n 行目の住所に入力の文字列が含まれていたら, 配列に n を追加する.
 - (b) 配列をクイックソートで, 郵便番号が昇順になるようにソートする.
 - (c) 配列に入っている情報をもとに, 郵便番号と住所を出力する.
 - (d) ユーザーに, 絞り込み検索をしたいか尋ねる.
 - (e) 絞り込み検索をする場合, 絞り込みたい住所の入力を受けつけ, 変数として受け取る.
 - (f) $n=1$ から $n=「6 (a) i の配列で追加した数」$ まで, 以下の処理を繰り返す.

- i. 6 (a) i の配列の n 番目の要素の行の都道府県名, 市区町村名, 町域名のいずれかに
入力の文字列が含まれていたら, 6 (a) i の配列とは別の配列に n を追加する.
 - ii. 前述のいずれにも, 入力が含まれていなかったかつ, 6 (a) i の配列の n 番目の要素
の行の住所に入力の文字列が含まれていたら, 6 (f) i の配列に n を追加する.
 - (g) ユーザーに絞り込み検索をさらに, したいか尋ねる.
 - (h) 絞り込み検索をしたい場合, 6e ~ 6g の操作を同様に繰り返す.
7. いずれかの操作が終わったら, 再び, 2 ~ 6 の操作を繰り返す.

3 プログラムの説明

本プログラムは, C 言語にて記述した.

3.1 入力形式

プログラムへの入力は, モードによって, 異なる.

プログラムに引数を与えずに, 実行する場合, インタラクティブモードとなる. この場合, プログラムは実行中に, 検索モードと検索クエリを標準入力から受け取り, 検索を行う.

プログラム実行時に, 検索モードと検索クエリの列を記入した入力ファイル名を引数として指定し, 実行する場合, ファイルモードとなる. この場合, プログラムは検索モードと検索クエリをファイルから読み取り, 検索する.

3.2 出力形式

プログラムの出力は, 検索の結果, 得られた住所である. 出力は, [郵便番号]:[住所] の形式で出力し, 結果が複数ある場合は, 郵便番号順が昇順になるように出力する.

3.3 データ構造

本プログラムのデータ構造を, 表 2 に示す.

表 2: プログラムのデータ構造

データ型	型名	概要
構造型	ADDRESS	csv の 1 行分のデータを記憶する. 要素は表 3 に示す

表 3: ADDRESS 型の要素

要素の型	変数名	概要
int	code	郵便番号を記憶する
char 型配列	pref[20]	住所の都道府県名を記憶する.
char 型配列	city[32]	住所の市区町村名を記憶する.
char 型配列	town[116]	住所の町域名を記憶する.

3.4 大域変数

本プログラムの大域変数を, 表 4 に示す.

表 4: プログラムの大域変数

型名	変数名	概要
ADDRESS	address_data[200000]	CSV の全行のデータを ADDRESS 型配列として記憶する.
int	mode	検索モードを記憶する.
int	refine_flag	絞り込み検索をするかどうか記憶する.
char	query[200]	検索クエリ (郵便番号または住所) を記憶する
long	total_count	何行データを読み込んだかを記憶する.
int	hit_index_list[200000]	住所検索で何行目がヒットしたかを記憶する.
int	hit_list_count	住所検索でヒットした回数を記憶する.

3.5 関数の設計

関数の呼び出し関係を図 1 に示す.

main 関数から処理が始まり, まず init 関数で構造体の初期化が行われる. 次に, 関数に引数にあるかどうかで処理が分かれれる. 引数がある場合は run_from_file 関数が実施される. run_from_file 関数での呼び出し関係を図??に示す. 引数がない場合, respond 関数が呼び出される.

run_from_file 関数では, ファイルの検索モードに応じて, code_search 関数, address_search 関数が呼ばれる.

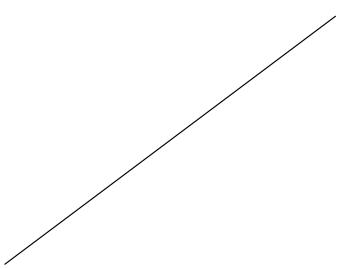


図 1: 関数の呼び出し関係 (main 関数始まり)