アドレス帳同期システムの開発

05H101 藤木仙雅

目次

1 はじめに 1

2 卒業研究の目的 2

3 既存のシステム 3

4 開発したシステム 4
4.1 開発環境 4
4.2 処理の流れ 4

5 考察 8
5.1 利点 8
5.2 問題点 8

6 まとめ 9

7 今後の課題 10

付録 A フローチャート 12
A.1 アドレス帳入力のフローチャート 12
A.2 アドレス帳集約時のフローチャート 14
A.3 アドレス帳更新確認のフローチャート 23

付録 B ソースコード 29
B.1 main-cui.rb 29
B.2 collect.rb 31
B.3 watch.rb 76

1　はじめに

今日では、様々な場所にアドレス帳が分散している。例えばパーソナルコンピュータ (以下、コンピュータ) 内に年賀状作成ソフトウェア、メーラー[*1]、Web メールなどのアドレス帳が存在し、さらには携帯電話端末にもアドレス帳が存在する。このように複数のアドレス帳を扱っていると管理するのは容易ではなく、全てのアドレス帳に一つずつ情報を入力していくには非常に手間がかかる。さらには情報の入力洩れなども起こりうる。例えば、会社のコンピュータに仕事用のアドレス帳を更新した時、自宅のコンピュータにも同じ情報を入力しておないと、いざ自宅で必要な時に必要な情報が分からないという情況になってしまう。さらには、年賀状作成ソフトウェアは用途が限られており、頻繁に扱うことが少ない。よって、他のソフトウェアでアドレス帳を更新した際に年賀状作成ソフトウェアのアドレス帳の更新を忘れることがしばしば見受けられる。そこでアドレス帳同期、共有システム[*2]を用い、アドレス帳管理を容易にすることができる。

本研究ではユーザのアドレス帳管理や情報入力の効率を上げるため、アドレス帳同期システムを開発した。

[*1] 電子メールの作成、送受信や送受信したメールを保存、管理するソフトウェア。

[*2] アドレス帳同期システム: 分散したアドレス帳を集約し同期させるもの。また異なるコンピュータ間で自動同期させ常に最新の情報を維持させる。アドレス帳共有システム:アドレス帳同期システムに、他ユーザとも共有する機能を持たせたもの

2　卒業研究の目的

本研究では複数のソフトウェアに分散したアドレス帳を集約し、同期させることを目的とする。マルチプラットフォームに[*3]対応し様々な OS[*4] で動作させる。CUI[*5]と GUI[*6]で動作させ、幅広いユーザが扱えるようにする。この時、CUI をバックエンドで動作させ、GUI をフロントエンドで動作させる。異なるコンピュータ間の同期には LDAP [*7] [1] サーバを用いる。LDAP サーバを用いたアドレス帳の同期は Web メールやメーラで標準搭載されている機能でもあり、開発するアドレス帳同期システムで同期させたアドレス帳を LDAP サーバに登録し、Web メールやメーラで扱うこともできる。対応するアドレス帳形式には、様々なソフトウェアで互換性が高い vCard 形式[*8]を用いる。

[*3] アプリケーションソフトウェアが複数の OS に対応していること。
[*4] Operating System
[*5] Character-based User Interface
キーボードを用いて入力を行い、文字によって出力を行うユーザインターフェース
[*6] Graphical User Interface
コンピュータグラフィックスとポインティングデバイスを用いて、直感的な操作を提供するユーザインターフェース
[*7] Lightweight Directory Access Protocol
インターネットやイントラネットなどの TCP/IP ネットワークで、ディレクトリデータベースにアクセスするためのプロトコル
[*8] 電子名刺の標準フォーマット。RFC2426 で規定。

3 既存のシステム

アドレス帳共有、同期システムとして以下のシステムが存在している。

- Ripplex [5]
  Ripplex 社が開発した無料アドレス帳ソフト。web サービスではなく、クライアントアプリ[*9]である。複数のコンピュータ間でアドレス帳情報が自動的に同期。他ユーザのアドレス帳と共有し情報管理が容易。アドレス帳だけでなく、Skype[*10]、Twitter[*11]や mixi[*12] サービスの情報も管理可能である。各情報は Ripplex サーバではなく、ユーザのコンピュータ内にて管理される。

- Plaxo [6]
  Plaxo 社が提供するサービス。ネットワークを活用した今までにない、常に最新の情報を保つアドレス帳として開発。自分の情報を公開し、他ユーザと共有するのが第一趣旨であり、アドレス帳やスケジュール帳など共有することができる。現在は pulse という機能が追加され、ブログ、写真、ビデオや様々なレビューなども共有することができる。各情報は Plaxo サーバにて管理。

- Mobile Me [7]
  Apple 社が提供しているいサービス。メール、アドレス帳、カレンダーやギャラリーなどを同期。メールは iPodTouch[*13]や iPhone[*14]だけでなく、MacOS の Mail.app や MS-Windows の Outlook [*15], Outlook Express[*16], Windows Mail とも連係。アドレス帳は MacOS、MS-Windows のアドレス帳と同期させることができる。各情報はクラウド[*17]に保存。

- Soocial [8]
  コンタクト情報にフォーカスしているオランダ発の Web サービス。携帯電話端末、Mac、Gmail[*18]や Outlook のアドレス帳データを同期できる。

[*9] クライアント・アプリケーション
デスクトップパソコンやモバイルパソコン内で動作するアプリケーション。
[*10] ルクセンブルクに籍を置く Skype Technologies 社が提供するインターネット電話サービス。
[*11] Twitter 社が開発した、個々のユーザが「つぶやき」を投稿し合うことでつながるコミュニケーション、サービス。
[*12] 株式会社ミクシィが運営する SNS(Social Networking Service)
[*13] Apple 社が開発したミュージックプレイヤー。iPod にゲーム、インターネットの機能を追加したもの。
iPod とは Apple 社が開発したミュージックプレイヤーである。
[*14] Apple 社が開発した、携帯電話端末、iPod やインターネットデバイスを一つにしたもの。
[*15] Microsoft Outlook
Microsoft 社が Windows 向けに販売している Exchange Server の標準クライアント。電子メールの機能以外に、予定表、連絡先管理、仕事管理やメモなどの機能が実装されている。
[*16] Microsoft の Internet Explorer に付属する電子メール及びニュースグループ送受信用クライアントソフトウェア。
Microsoft Outlook とは別物であり、Internet/Intranet 専用の電子メールクライアント。
[*17] Apple のオンラインサーバ
[*18] Google が提供している web メール

4　開発したシステム

本研究で、アドレス帳同期システム (Address-Book Synchronous System) を開発した。
本システムの概略図を図 **1** に示す。

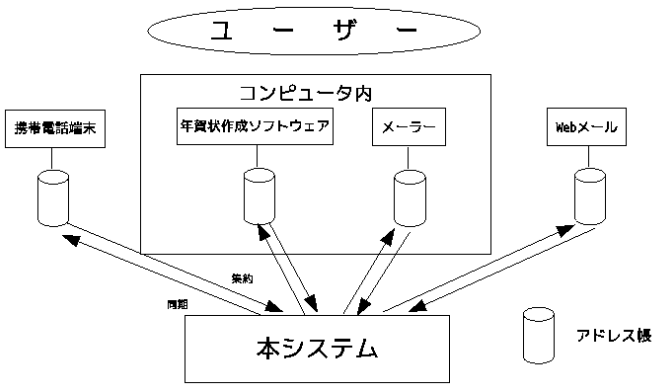

図 1　システムの概略図

4.1　開発環境

- ハードウェア (AT 互換機)
  - CPU:Intel(R) Celeron(R) CPU 2.80GHz
  - Cache Memory:256KB
- OS：Vine Linux 4.2 [2] / Ubuntu Linux 8.04 [3] [研究室内マシン/ 個人所有マシン]

- 開発言語：Ruby [4] 1.8.5 / 1.8.6 [研究室内マシン/ 個人所有マシン]

4.2　処理の流れ

以下に処理の流れと付加する機能について説明する。

4.2.1　同期させるアドレス帳の入力

同期させるアドレス帳 Path を一つずつ入力。その際に入力したアドレス帳が存在するか、また vCard 形式であるかを判定する。入力されたアドレス帳が存在し、vCard 形式ならば、アドレス帳の Path を path.txt ファイルに書き込む。
アドレス帳入力画面を図 **2** に示す。
アドレス帳入力のフローチャートを図 **8**、**9** に示す。

図 2　アドレス帳入力画面

4.2.2　入力したアドレス帳の集約

アドレス帳集約の際には、入力されたアドレス帳を一件ずつ解析。解析が終了すれば Backup データを用い、既に集約しているアドレス帳と比較し、同一名称で登録されているデータがないか確認。同一名称で登録されているデータを発見すれば、双方が同一ユーザなのかユーザ側が判断。同一ユーザであれば双方のデータを結合させる。同一ユーザ結合の際、同項目に別の情報が入っていた場合もユーザ側に判断させる。同一名称で登録されているデータが発見されなければ解析終了。

解析には一件のアドレス帳一項目ずつ解析し、名前、住所、電話番号、E-MAIL の個人情報を抽出。抽出した個人情報の内、名前は同一ユーザ発見の際に使用、その他の項目は各ユーザの情報として使用。

入力されたアドレス帳の解析が全て終了すれば、RFC2426 [9] で定められている vCard 形式のデータ型に置き換える。

アドレス帳集約画面を図 **3** に示す。

アドレス帳集約のフローチャートを図 **10**～**21** に示す。

4.2.3　集約したアドレス帳の同期

path.txt ファイルを参照し、同期させているアドレス帳全てに書き込み処理を行う。書き込む際には nkf[*19] [10] を用いて文字コード変換を行う。同時に Backup ファイルも作成する。Backup ファイル作成時には vCard の version [*20]別で作成する。

4.2.4　同期させたアドレス帳の更新

起動時もしくは renewal コマンドにて同期させたアドレス帳の更新を行う。

path.txt ファイルを参照し、全てのアドレス帳の最終更新時間を取得。最終更新時間の新しいものと、Backup ファイルの更新時間を比較。最終更新時間を取得したアドレス帳が新しものならば、他の同期させているアドレス帳へ書き込み処理を行い。新しく Backup ファイルを作成する。この場合も、nkf を用いて文字コードの変換を行う。

アドレス帳更新画面 (更新無し) を図 **4** に示す。

アドレス帳更新画面 (更新有り) を図 **5** に示す。

アドレス帳更新のフローチャートを図 **22**～**27** に示す。

[*19] 文字コードを変換するコマンド

[*20] 現在、version2.1 と version3 が存在する。

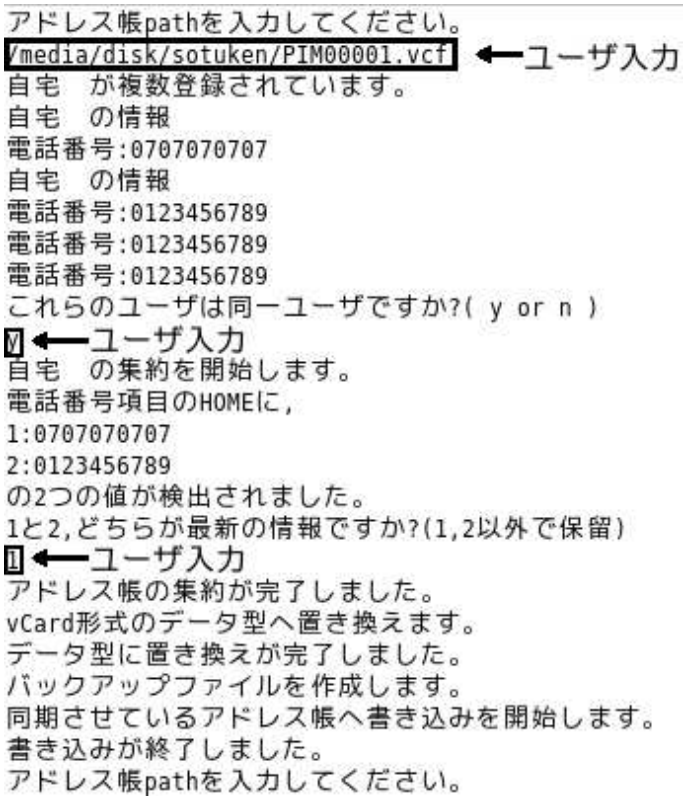

図 3　アドレス帳集約画面

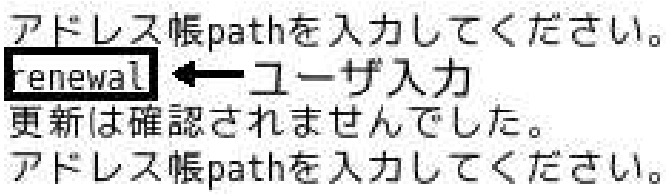

図 4　アドレス帳更新 (更新無し)

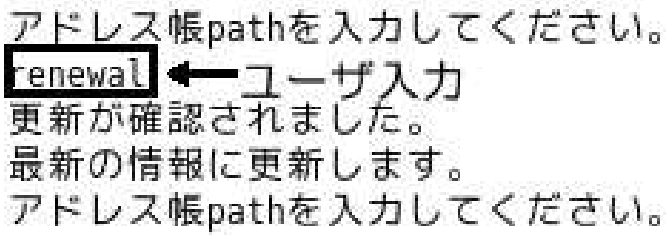

図 5　アドレス帳更新 (更新有り)

4.2.5　同期させているアドレス帳の一覧表示

pathfile コマンドにて同期させているアドレス帳の一覧を表示させる。

アドレス帳一覧表示画面を図 **6** に示す。

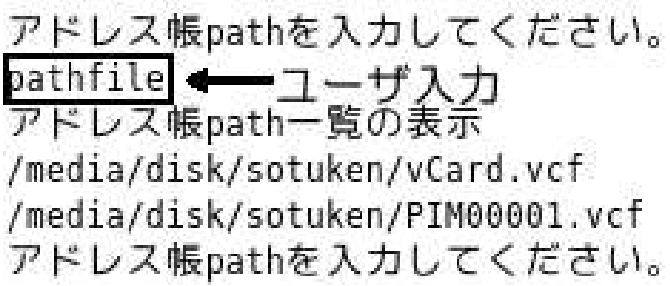

図 6　アドレス帳一覧表示

4.2.6　同期の解除

delete コマンドにて同期させているアドレス帳の解除を行う。

解除方法として、現在同期させているアドレス帳の一覧を表示。その中から同期を解除させたいアドレス帳の Path を入力し、入力したアドレス帳が正しければ一覧から削除させる。

アドレス帳同期解除画面を図 **7** に示す。

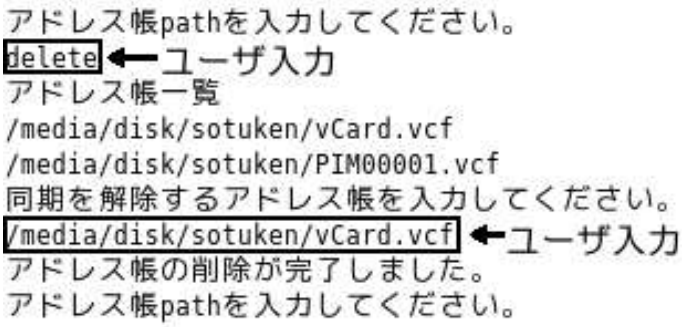

図 7　アドレス帳同期解除

5　考察

アドレス帳同期システムを開発し、実際に稼働させた。
本システムについて、以下の利点と問題点が挙げられる。

5.1　利点

マルチプラットフォームで動作する Ruby を用いて開発したので、様々な OS で動作する。コンピュータ内に分散されたアドレス帳を集約させることで、アドレス帳の管理、情報入力の手間を省くことができる。対応するアドレス帳に vCard 形式を用いた。vCard 形式は RFC2426 で規定されているので様々なソフトウェアでの互換性が高い。

5.2　問題点

現在は CUI のみの動作なので幅広いユーザが扱うことは難しい。Ruby でプログラミングするにあたり、RFC2426 で規定されているデータ型全て (非標準の型は除く) に対応させて開発した。それにより、一つずつ型の解析を行っている為、読みにくいソースコードとなっている。vCard 形式は RFC2426 で規定されているが、“X-” で始まる名前を持つものは非標準の私的な型であり、各ソフトウェアによって様々な情報が当てはめられている。よって、全てに対応させることは出来ない。

6 まとめ

本研究では、分散したアドレス帳を集約し、同期させるシステムを開発した。これにより複数のアドレス帳の管理を容易にし、情報入力の手間を省いた。また、対応するアドレス帳に vCard 形式を用いることにより、様々なソフトウェアに対応することができる。

テスト運用として、携帯電話端末 (docomo P905i)、Windows マシンに標準搭載されている “アドレス帳” ソフトウェアと Gmail アドレス帳を用いてアドレス帳の同期を行い、アドレス帳同期システムとして稼働することを確認。しかし、携帯電話端末のアドレス帳は JIS X 0201 右側部分が使われている。現状では JIS X 0201 右側部分が認識されておらず携帯電話端末で集約したアドレス帳を読み込んだ際、振り仮名が空白となってしまう。

7　今後の課題

- LDAP サーバの導入
  異なるコンピュータ間で同期させるため LDAP サーバを導入する。しかし、アドレス帳をサーバに置くことを嫌うユーザが存在する可能性があるので、異なるコンピュータと同期させるか否かをユーザ側で判定をさせる。

- GUI でも動作
  幅広いユーザが扱えるように、CUI だけでなく、GUI を用いて動作させるようにする。

- アドレス帳の自動更新
  更新を確認するのに、開発したシステムの再起動、“renewal” コマンドや更新ボタンを押す手間を省くため、アドレス帳更新の確認を分単位で行う。更新確認時間の間隔はユーザ側で指定する。

- Ruby 環境でないマシンでの稼働
  現在は Ruby 環境のマシンでしか稼働しない。よって Exerb[*21] や RubyScript2Exe[*22] を用いて exe ファイルに変換し、MS-Windows でも稼働させる。

- Backup データの作成
  現在、Backup データは過去一回分しか作成していない。よってユーザの誤操作により意図しない結果により更新してしまった場合のことを考慮し、過去数回分の Backup を作成する。過去何回分作成するかをユーザ側で指定させ、複数の Backup データを作成する。

[*21] Ruby のスクリプト、拡張ライブラリを単一で動作可能な Windows 実行形式ファイルに変換するソフトウェア。
[*22] Ruby スクリプトから Windows 用と Linux 用の実行形式ファイルを作成するソフトウェア。

謝辞

本研究を進めていく上で、ご教授頂いた大垣 斉講師、藤井 信夫教授、情報安全工学研究室の卒業研究生並びに情報システム工学科卒業生の方々に深く感謝の意を表します。

参考文献

[1] LDAP, [http://e-words.jp/w/LDAP.html]
[2] Vine Linux, [http://www.vinelinux.org/]
[3] Ubuntu Linux, [http://www.ubuntulinux.jp/]
[4] オブジェクト指向スクリプト言語 Ruby, [http://www.ruby-lang.org/ja/]
[5] Ripplex, [http://www.ripplex.com/jpn/index.html]
[6] Plaxo, [http://www.plaxo.com/?lang=ja]
[7] Mobile Me, [http://www.apple.com/ja/mobileme]
[8] Soocial, [http://www.soocial.com/]
[9] RFC2426, [http://www.y-adagio.com/public/standards/tr_vcd_2426/toc.htm]
[10] nkf, [http://itpro.nikkeibp.co.jp/article/COLUMN/20060227/230849/]

(注)URI は全て 2008 年 11 月 25 日時点のもの

付録 A　フローチャート

A.1　アドレス帳入力のフローチャート

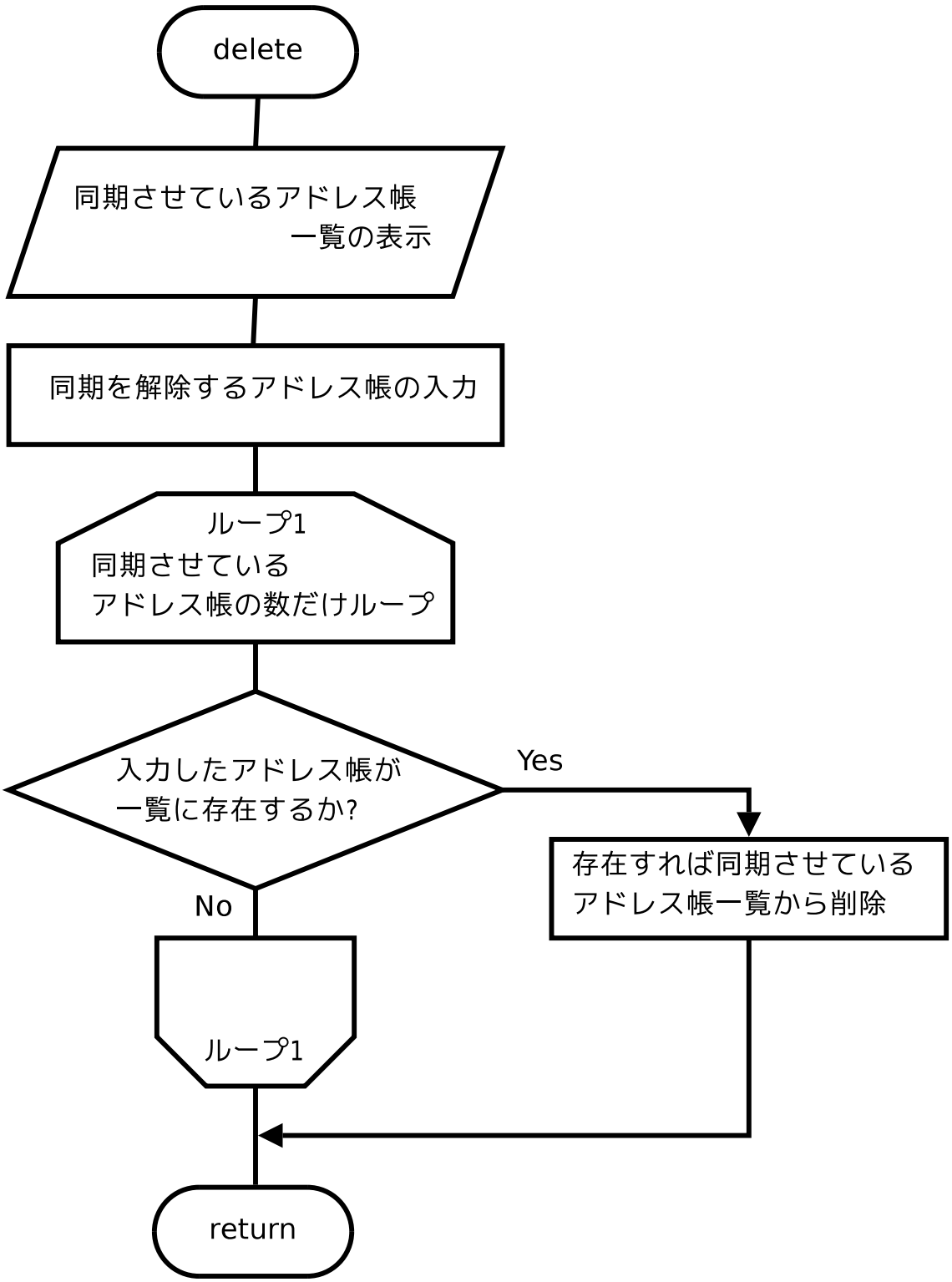

図 8　delete メソッド

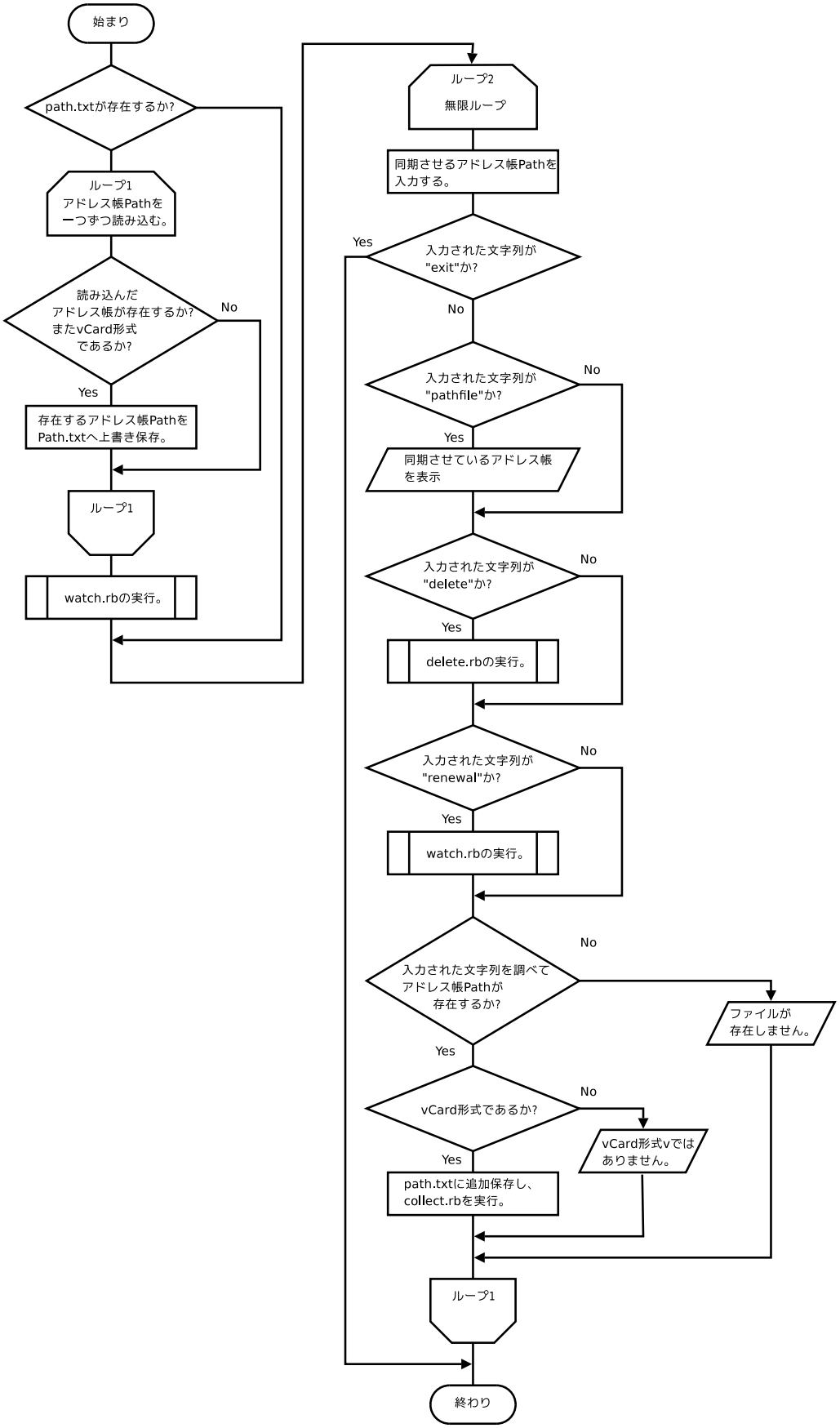

図 9　main-cui.rb

A.2　アドレス帳集約時のフローチャート

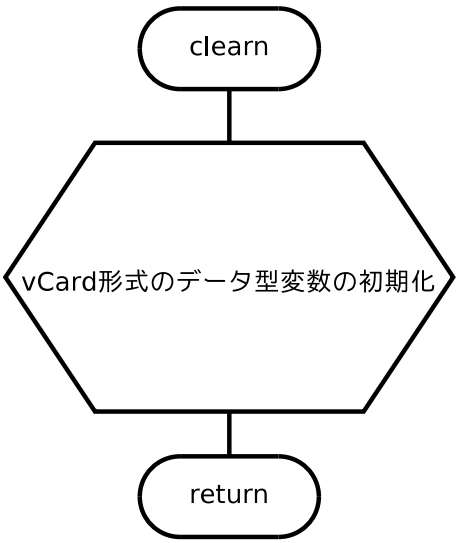

図 10　clearn メソッド

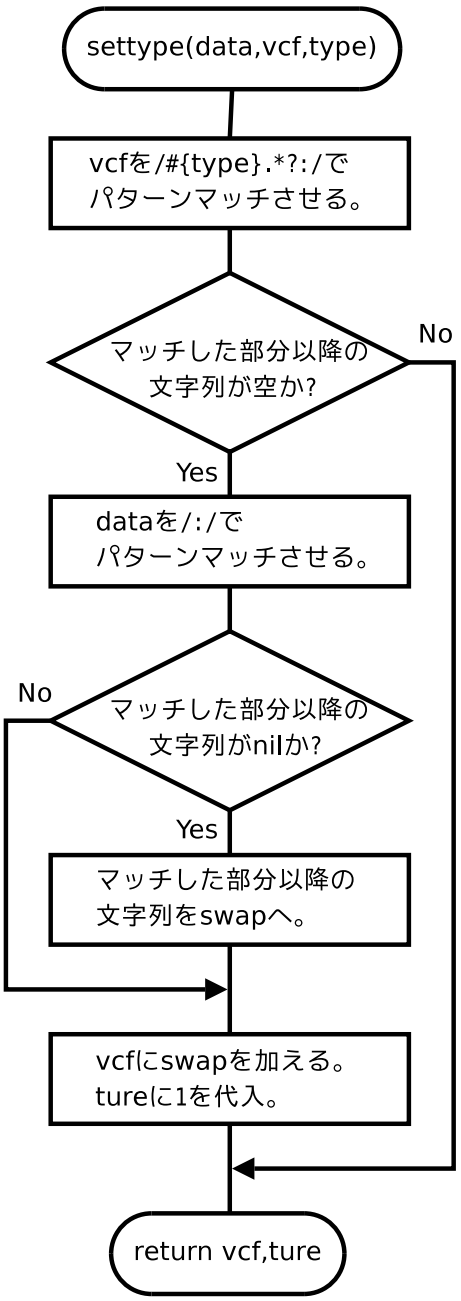

図 11　settype メソッド

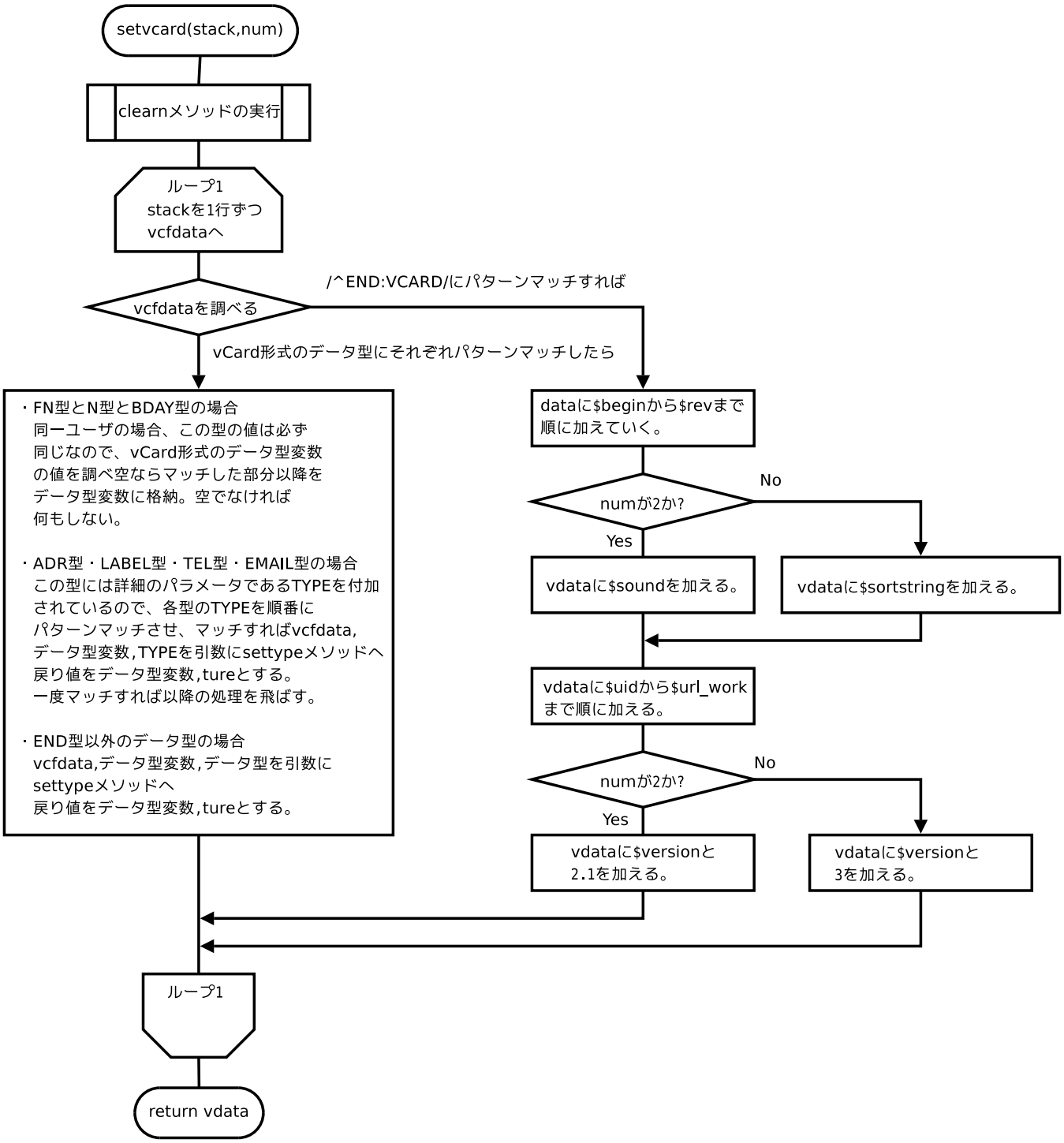

図 12　setvcard メソッド

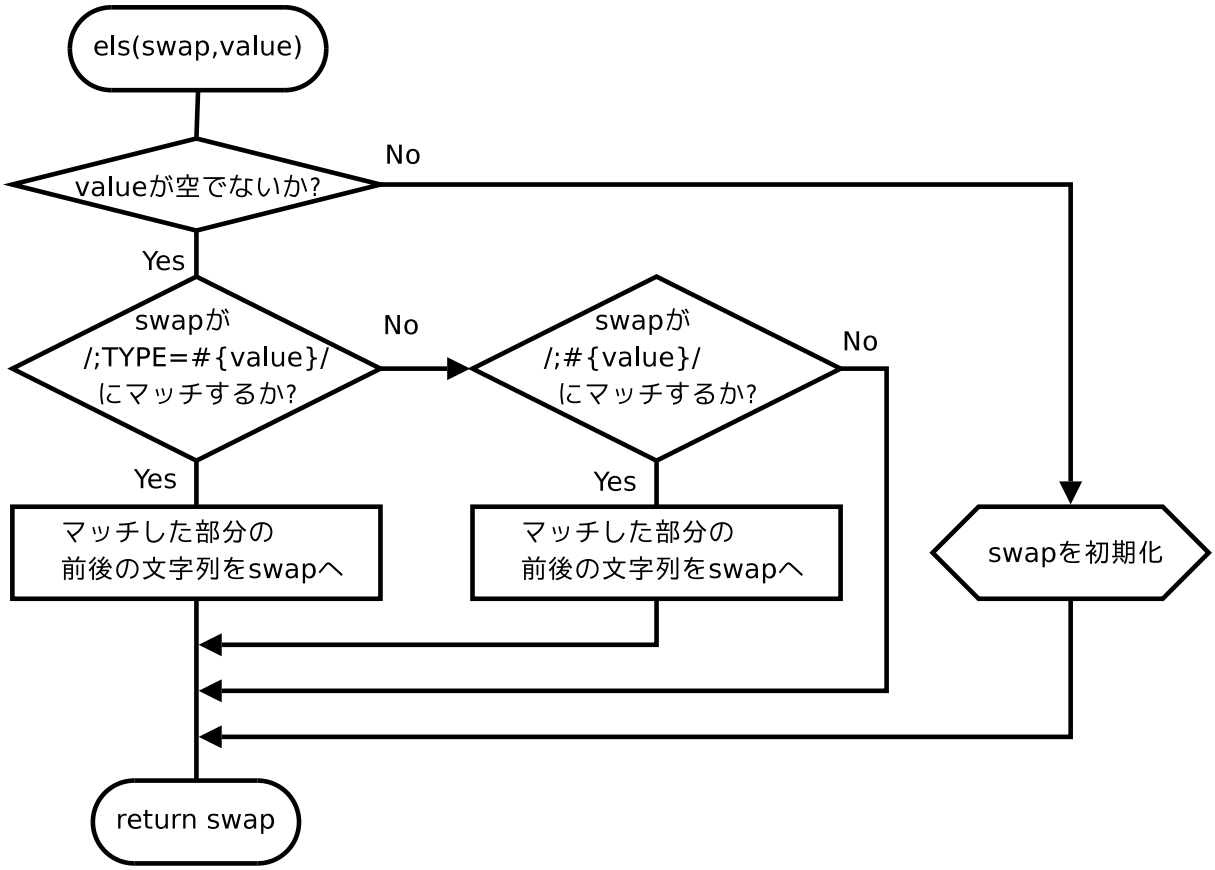
els(swap,value)
valueが空でないか?
No
Yes
swapが
/;TYPE=#{value}/
にマッチするか?
No
swapが
/;#{value}/
にマッチするか?
No
Yes
Yes
マッチした部分の
前後の文字列をswapへ
マッチした部分の
前後の文字列をswapへ
swapを初期化
return swap

図 13　els メソッド

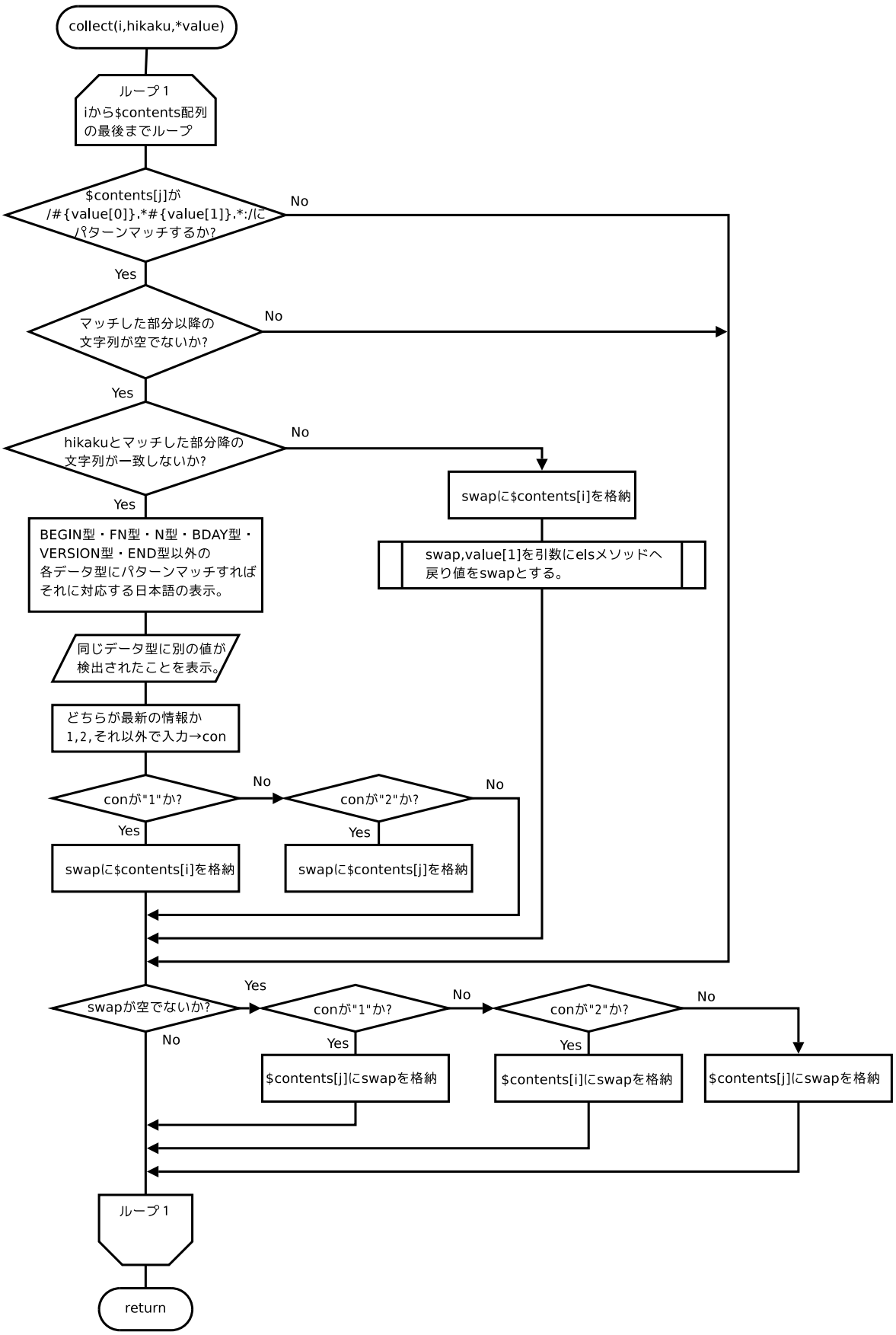

図 14　collect メソッド

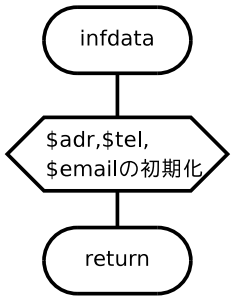

図 15　infdata メソッド

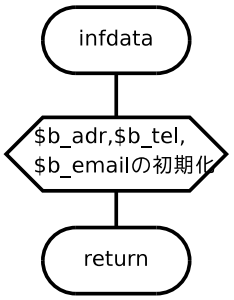

図 16　infbdata メソッド

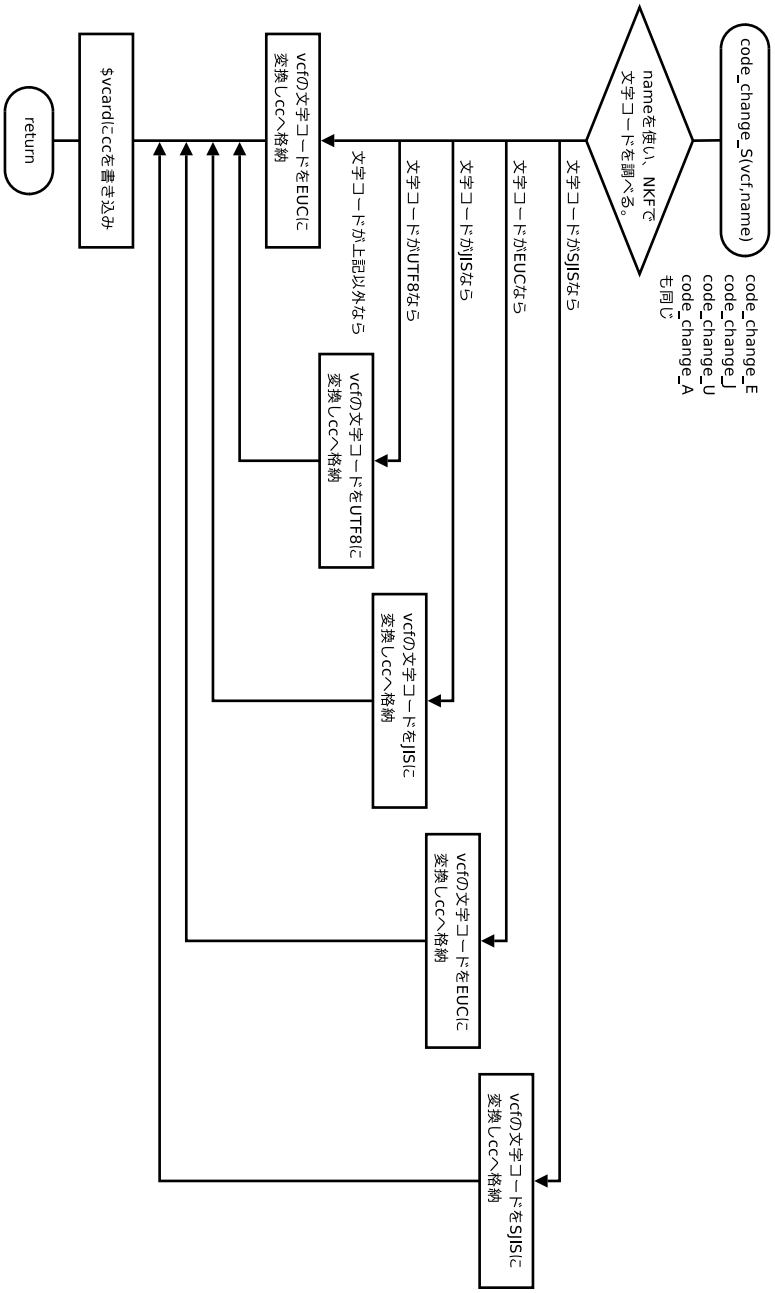

図 17　code-change メソッド

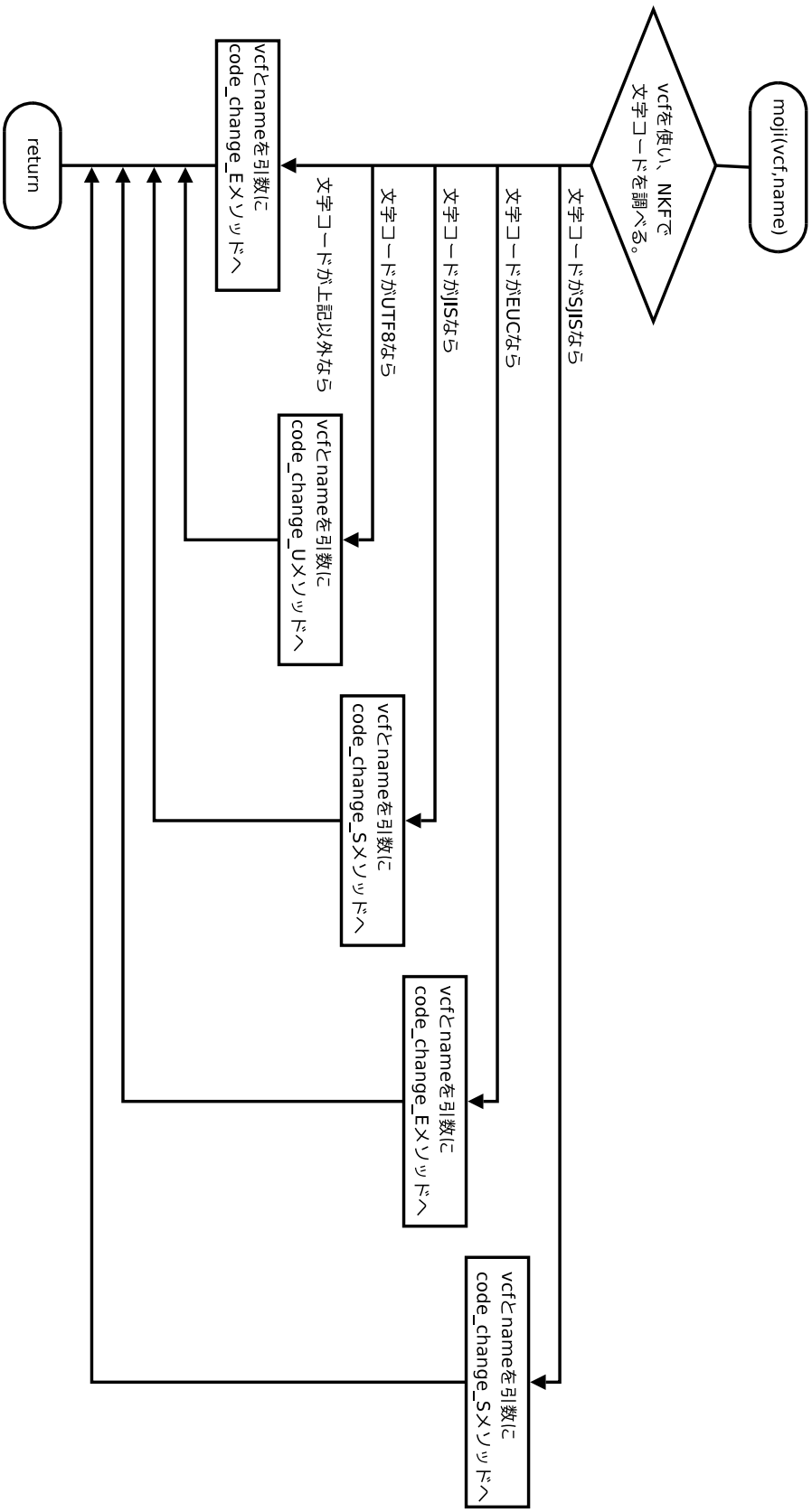

図 18　moji メソッド

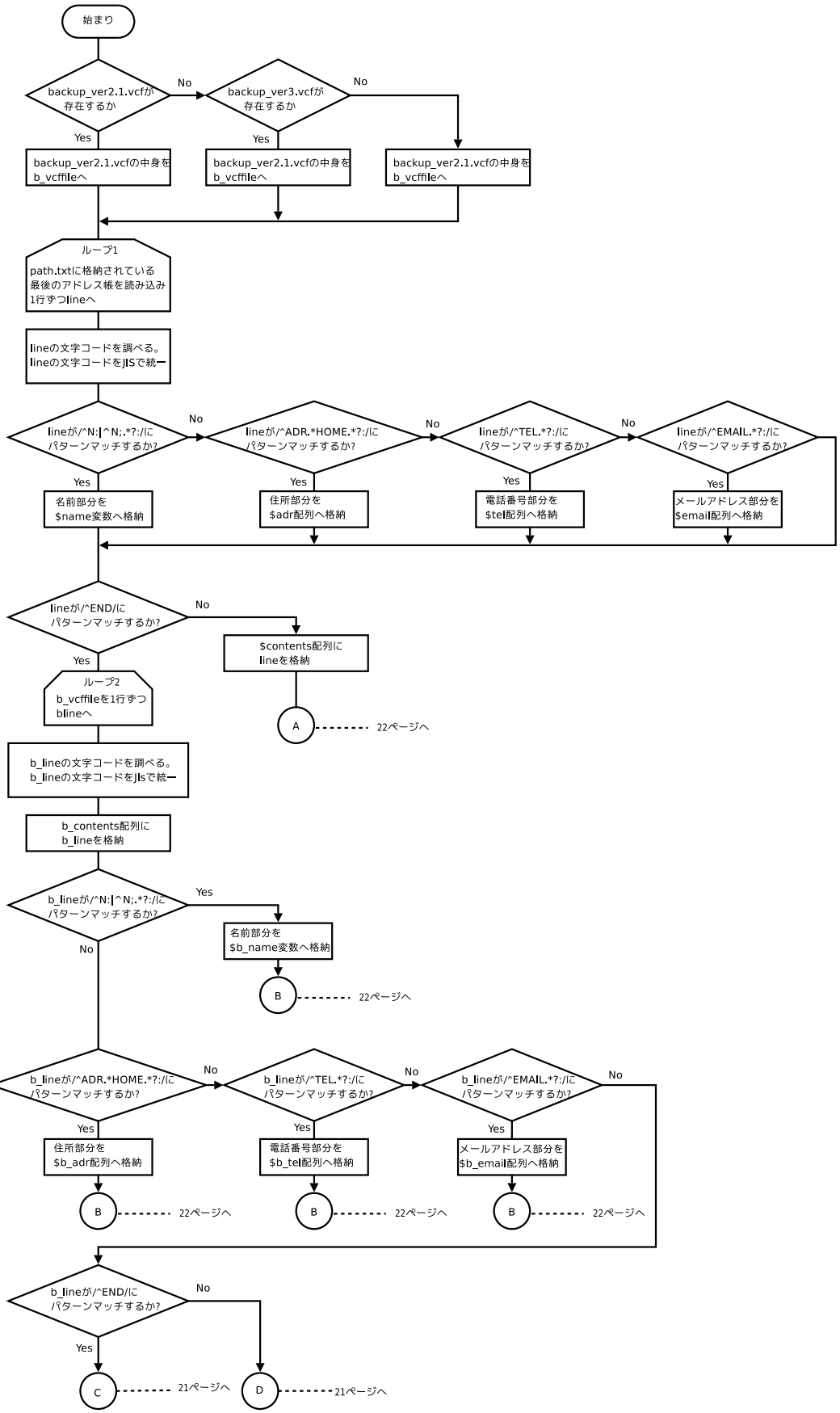

図 19　collect.rb(1)

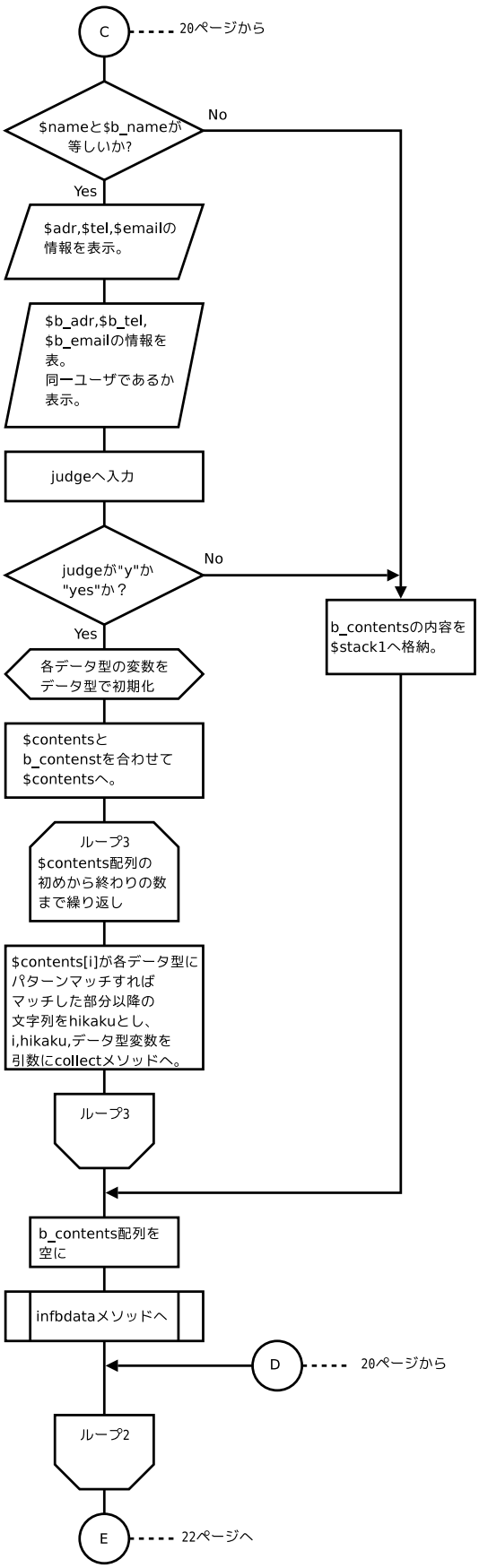

図 20　collect.rb(2)

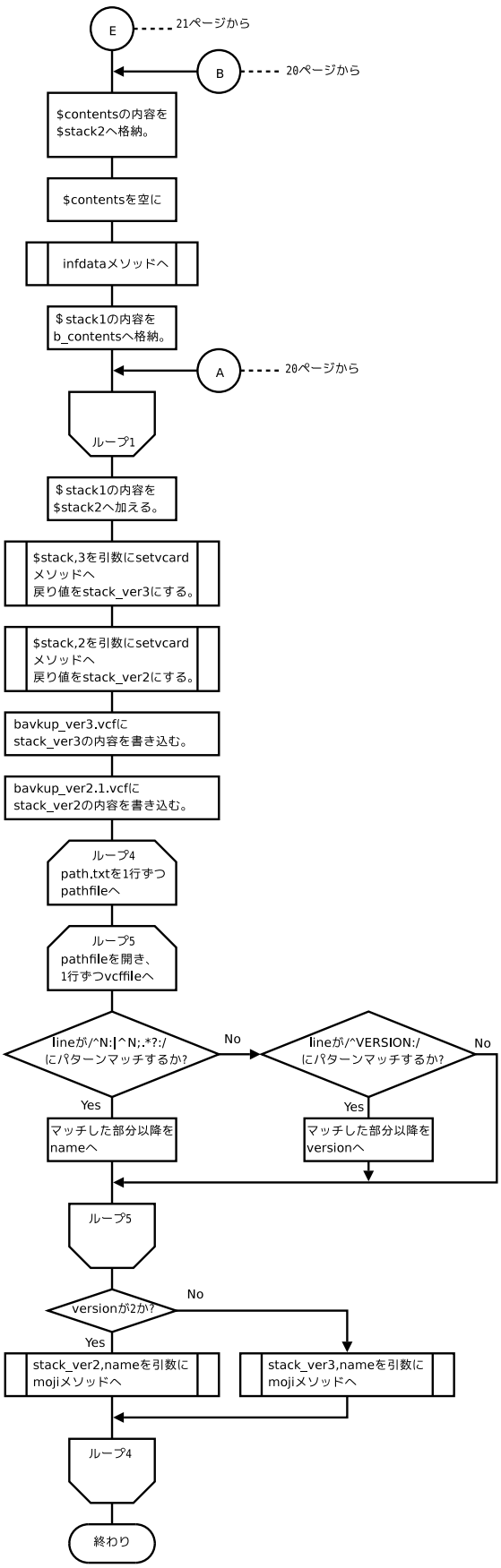

図 21　collect.rb(3)

A.3　アドレス帳更新確認のフローチャート

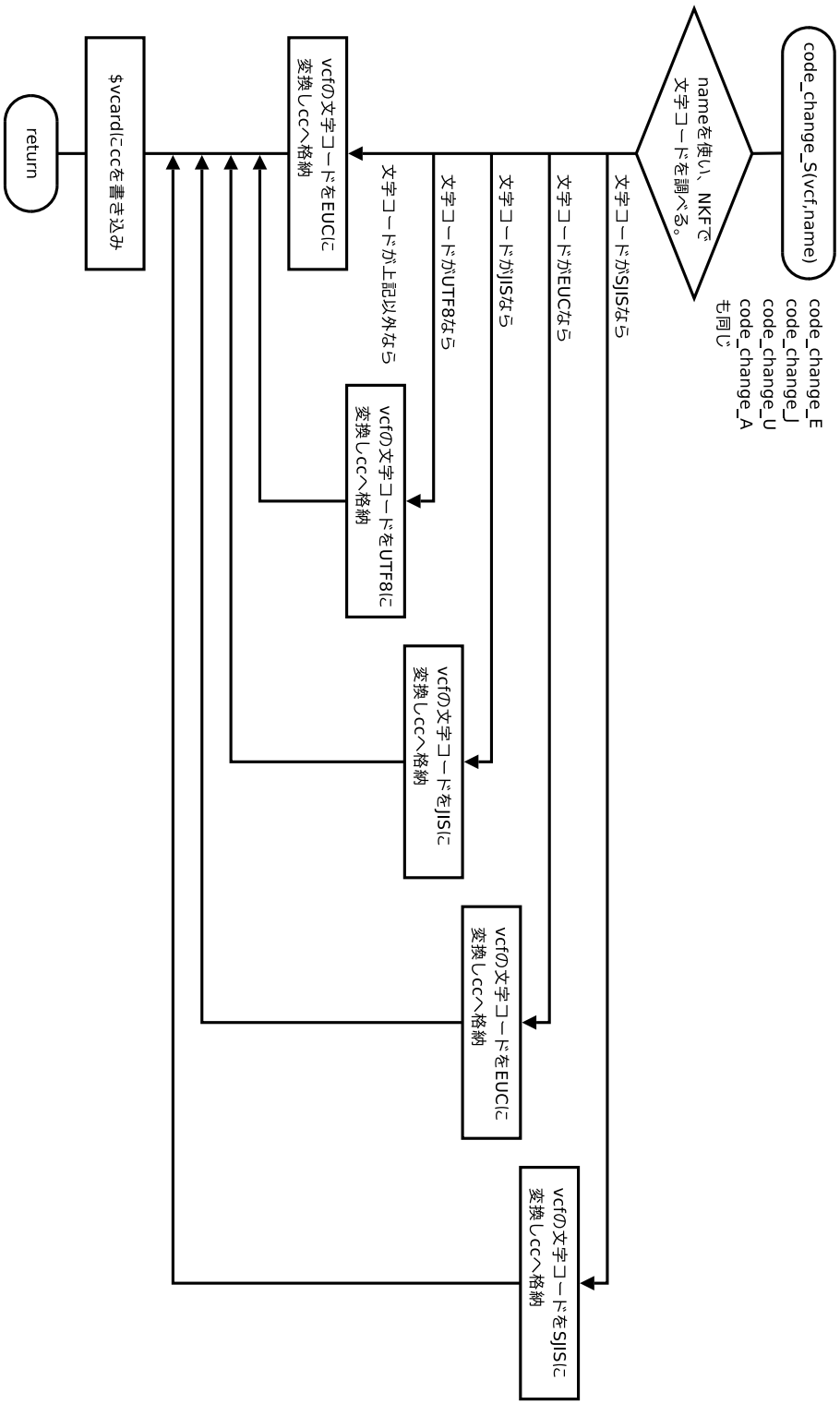

図 22　code-change メソッド

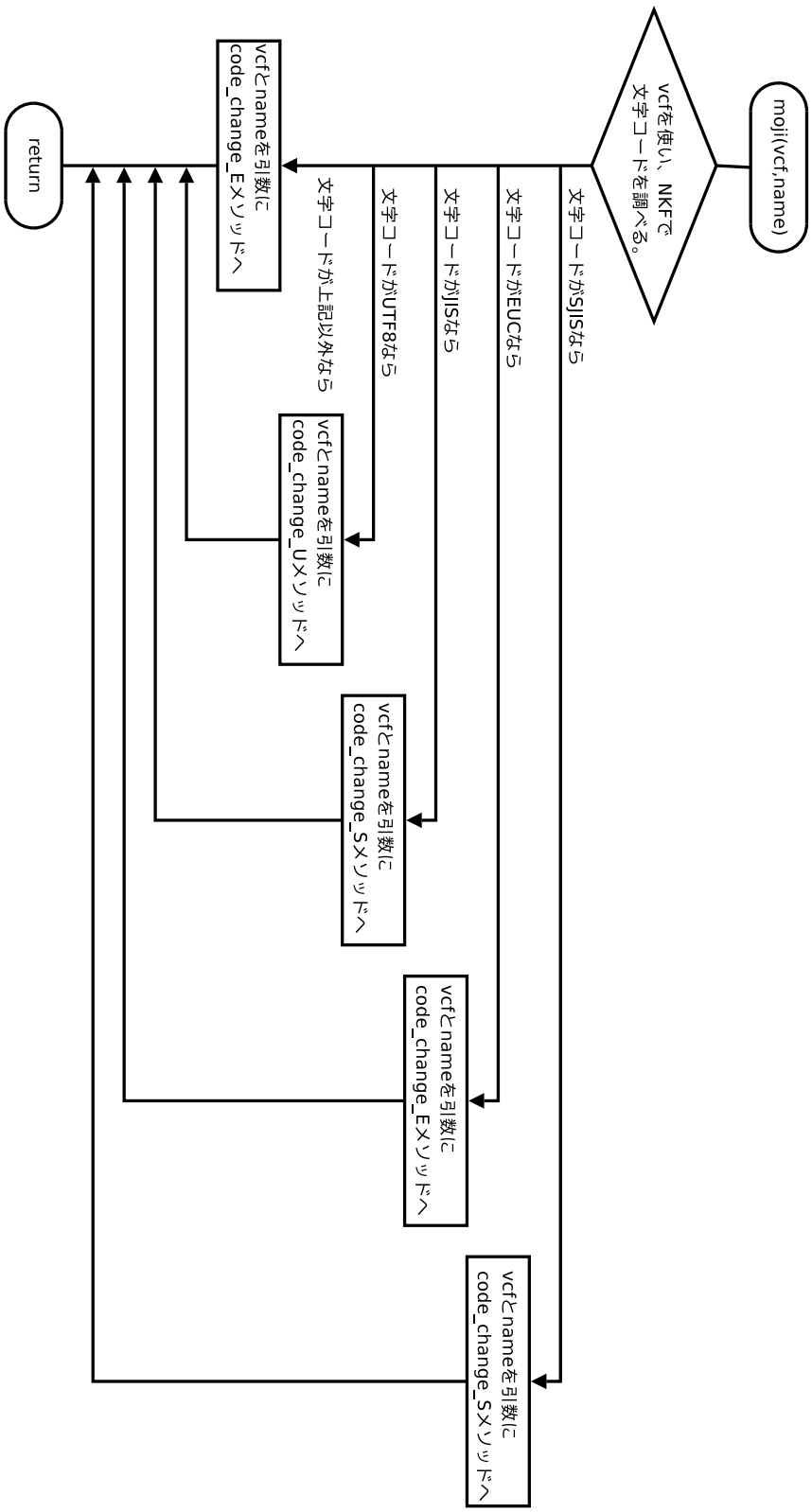

図 23　moji メソッド

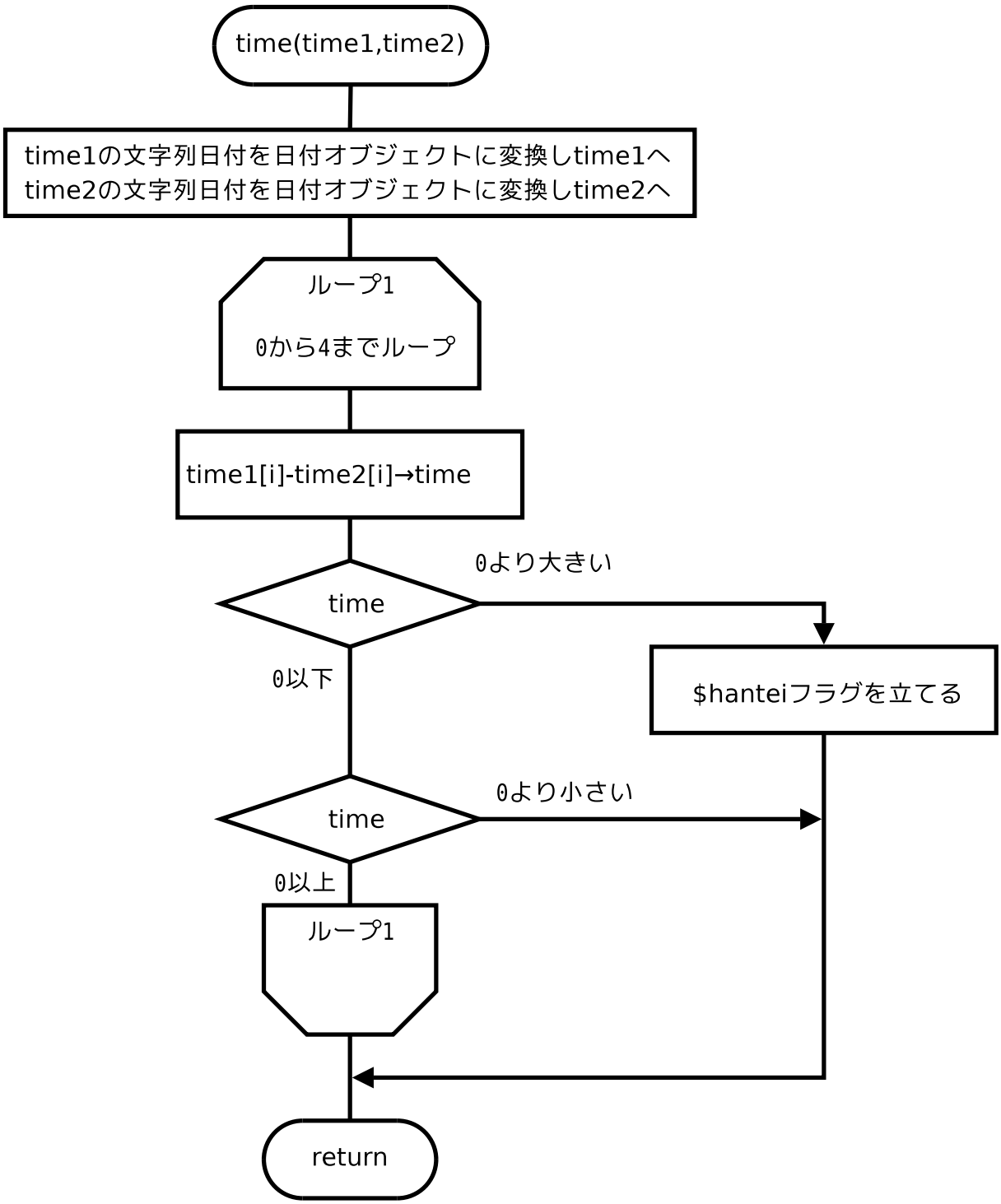

図 24　time メソッド

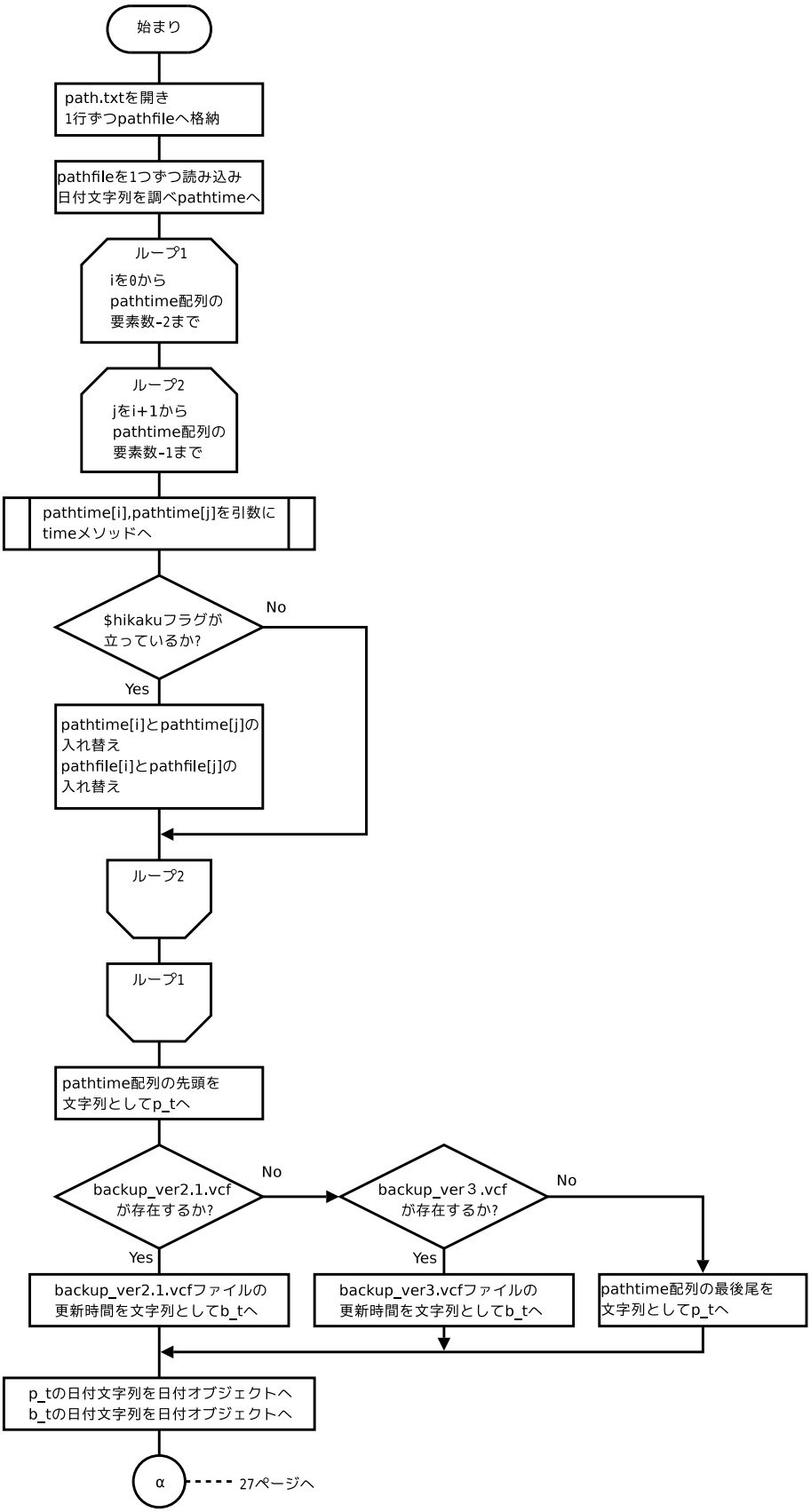

図 25　watch.rb (1)

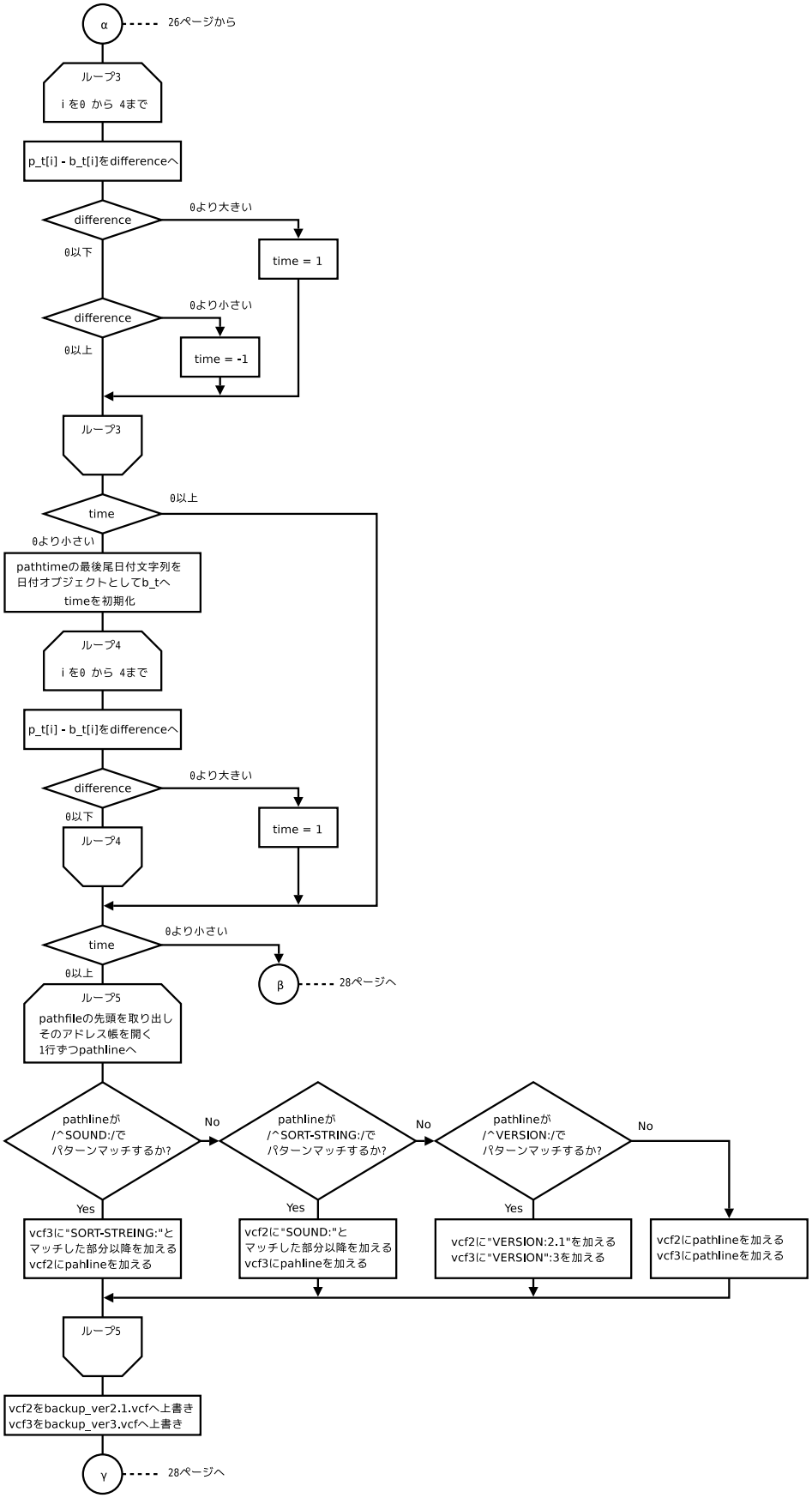

図 26　watch.rb (2)

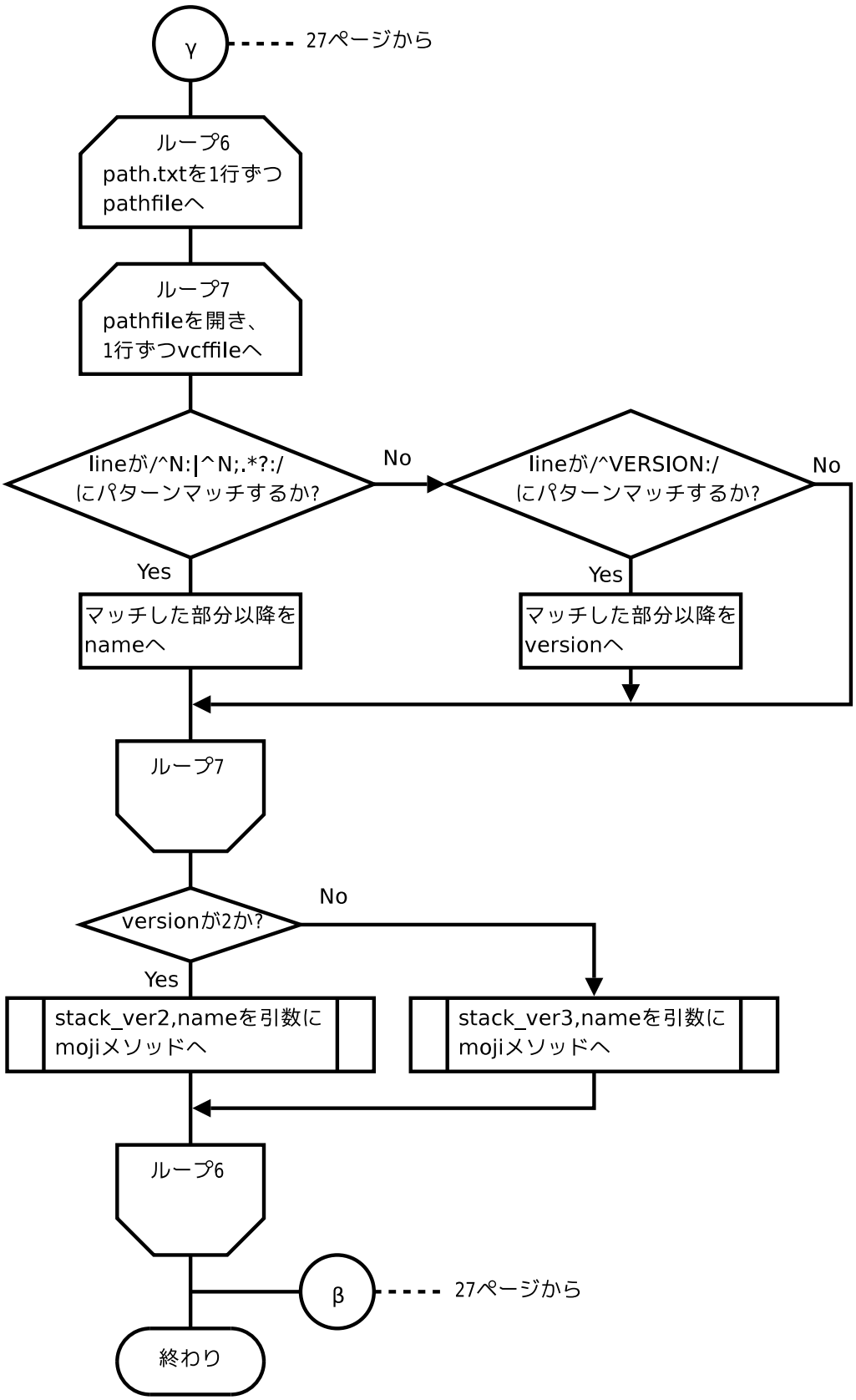

図 27　watch.rb (3)

付録 B　ソースコード

B.1　main-cui.rb

```
#!/usr/bin/env ruby

$KCODE = 'e'

def delete
  del = "nil"
  print("アドレス帳一覧\n")
  print $path
  print("同期を解除するアドレス帳を入力してください。\n")
  enter = gets.chomp
  for i in 0..$path.length-1
    if enter == $path[i].chomp
      del = "ture"
      break
    end
  end
  if del == "ture"
    $path.delete_at(i)
    pathfile = File.open("path.txt","w")
    pathfile.puts $path
    pathfile.close
    print("アドレス帳の削除が完了しました。\n")
  else
    print("アドレス帳が存在しません。\n")
  end
end

#cui によるメイン処理

# path.txt が存在すれば、中の path が存在するか判定
$path = Array.new
if File.exist?("path.txt")
  pathfile = File.open("path.txt")
  while pathline = pathfile.gets
    if File.exist?(pathline.chomp)
      if pathline =~ /.vcf$/ or pathline =~ /.VCF$/
        $path << pathline
      end
```

```
    end
  end
  pathfile.close
end

pathfile = File.open("path.txt","w")
if $path != []
  pathfile.puts($path)
end
pathfile.close

if $path != []
  print(‘ruby watch.rb‘)
end

#アドレス帳入力処理
while (1)

  print("アドレス帳 path を入力してください。\n")
  input = gets.chomp
  if input == "exit"
    break
  elsif input == "pathfile"
    print"アドレス帳 path 一覧の表示\n"
    pathfile = File.open("path.txt")
    print pathfile.read
    pathfile.close

  elsif input == "delete"
    delete

  elsif input == "synchronous"
    print(‘ruby watch.rb‘)

  elsif File.exist?(input)
    if input =~ /.vcf$/ or input =~ /.VCF$/
      pathfile = File.open("path.txt","a")
      pathfile.puts(input)
      pathfile.close

      #collect.rb の呼び出し
      system("ruby collect.rb")
```

```
    else
      print("vCard形式ではありません。\n")
    end
  else
    print("ファイルが存在しません。\n")
  end
end
```

B.2 collect.rb

```
#!/usr/bin/env ruby

$KCODE = 'e'
require 'nkf'

#vCardのデータ型初期化メソッド
def clearn
  $begin = "BEGIN:VCARD"
  $fn = "FN:"
  $n = "N:"
  $nickname = "NICKNAME:"
  $photo = "PHOTO:"
  $bday = "BDAY:"
  $adr_pref = "ADR;TYPE=PREF:"
  $adr_dom = "ADR;TYPE=DOM:"
  $adr_intl = "ADR;TYPE=INTL:"
  $adr_postal = "ADR;TYPE=POSTAL:"
  $adr_parsel = "ADR;TYPE=PARCEL:"
  $adr_home = "ADR;TYPE=HOME:"
  $adr_work = "ADR;TYPE=WORK:"
  $label_pref = "LABEL;TYPE=PREF:"
  $label_dom = "LABEL;TYPE=DOM:"
  $label_intl = "LABEL;TYPE=INTL:"
  $label_postal = "LABEL;TYPE=POSTAL:"
  $label_parsel = "LABEL;TYPE=PARSEL:"
  $label_home = "LABEL;TYPE=HOME:"
  $label_work = "LABEL;TYPE=WORK:"
  $tel_pref = "TEL;TYPE=PREF:"
  $tel_home = "TEL;TYPE=HOME:"
  $tel_msg = "TEL;TYPE=MSG:"
  $tel_work = "TEL;TYPE=WORK:"
  $tel_voice = "TEL;TYPE=VOICE:"
  $tel_fax = "TEL;TYPE=FAX:"
```

```
  $tel_cell = "TEL;TYPE=CELL:"
  $tel_video = "TEL;TYPE=VIDEO:"
  $tel_pager = "TEL;TYPE=PAGER:"
  $tel_bbs = "TEL;TYPE=BBS:"
  $tel_modem = "TEL;TYPE=MODEM:"
  $tel_car = "TEL;TYPE=CAR:"
  $tel_isdn = "TEL;TYPE=ISDN:"
  $tel_pcs = "TEL;TYPE=PCS:"
  $email_pref = "EMAIL;TYPE=PREF:"
  $email_internet = "EMAIL;TYPE=INTERNET:"
  $email_home = "EMAIL;TYPE=HOME:"
  $email_work = "EMAIL;TYPE=WORK:"
  $email_x400 = "EMAIL;TYPE=X400:"
  $mailer = "MAILER:"
  $tz = "TZ:"
  $geo = "GEO:"
  $title = "TITLE:"
  $role = "ROLE:"
  $logo = "LOGO:"
  $agent = "AGENT:"
  $org = "ORG:"
  $categories = "CATEGORIES:"
  $note = "NOTE:"
  $prodid = "PRODID:"
  $rev = "REV:"
  $sound = "SOUND:"
  $sortstring = "SORT-STRING:"
  $uid = "UID:"
  $url_home = "URL;HOME:"
  $url_work = "URL;WORK:"
  $version = "VERSION:"
  $class = "CLASS:"
  $key = "KEY:"
  $end = "END:VCARD"
end

#各データ形式に当てはめる際の重複を調べるメソッド
def settype(data,vcf,type)
  swap = ""
  ture = 0
  vcf =~ /#{type}.*:/
  if $' == ""
    data =~ /:/
```

```
    if $' != nil
      swap = $'
    end
    vcf += swap
    ture = 1
  end
  return vcf,ture
end

#vCard のデータ形式に当てはめるメソッド
def setvcard(stack,num)
  clearn
  vdata = ""
  stack.each do |vcfdata|
    vcfdata = vcfdata.chomp
    ture = 0
    case vcfdata
    when /^FN.*?:/
      fullname = $'
      $fn =~ /^FN.*?:/
      if $' == ""
        $fn += fullname
      end
    when /^N:|^N;.*?:/
      name = $'
      $n =~ /^N:|^N.*?:/
      if $' == ""
        $n += name
      end
    when /^NICKNAME.*?:/
      $nickname,ture = settype(vcfdata,$nickname,"NICKNAME")
    when /^PHOTO.*?:/
      $photo,ture = settype(vcfdata,$photo,"PHOTO")
    when /^BDAY.*?:/
      barthday = $'
      $bday =~ /^BDAY.*?/
      if $' == ""
        $bday += barthday
      end
    when /^ADR/
      if vcfdata !~ /DOM.*?:|INTL.*?:|POSTAL.*?|PARCEL.*:|HOME.*?:|WORK.*?:/
        $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
        if ture != 1
```

```
    $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  end
  if ture != 1
    $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
  end
  if ture != 1
    $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  end
  if ture != 1
    $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  end
  if ture != 1
    $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /DOM.*?:/
  $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
  end
  if ture != 1
    $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
  end
  if ture != 1
    $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  end
  if ture != 1
    $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  end
  if ture != 1
    $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /INTL.*?:/
  $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
  if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
```

```
  end
  if ture != 1
    $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  end
  if ture != 1
    $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  end
  if ture != 1
    $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  end
  if ture != 1
    $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /POSTAL.*?:/
  $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
  end
  if ture != 1
    $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  end
  if ture != 1
    $adr_intl,ture = settype(vcfdata,$adr_intal,"INTL")
  end
  if ture != 1
    $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  end
  if ture != 1
    $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /PARCEL.*?:/
  $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
  end
```

```
  if ture != 1
    $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  end
  if ture != 1
    $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
  end
  if ture != 1
    $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  end
  if ture != 1
    $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /HOME.*?:/
  $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
          if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
  end
  if ture != 1
    $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
  end
  if ture != 1
    $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
  end
  if ture != 1
    $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
  end
  if ture != 1
    $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
  end
  if ture != 1
    $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  end
end
if vcfdata =~ /WORK.*?:/
  $adr_work,ture = settype(vcfdata,$adr_work,"WORK")
  if ture != 1
    $adr_pref,ture = settype(vcfdata,$adr_pref,"PREF")
  end
  if ture != 1
```

```
      $adr_dom,ture = settype(vcfdata,$adr_dom,"DOM")
    end
    if ture != 1
      $adr_intl,ture = settype(vcfdata,$adr_intl,"INTL")
    end
    if ture != 1
      $adr_postal,ture = settype(vcfdata,$adr_postal,"POSTAL")
    end
    if ture != 1
      $adr_parcel,ture = settype(vcfdata,$adr_parcel,"PARCEL")
    end
    if ture != 1
      $adr_home,ture = settype(vcfdata,$adr_home,"HOME")
    end
  end
when /^LABEL.*?:/
  if vcfdata !~ /DOM.*?:|INTL.*?:|POSTAL.*?|PARCEL.*:|HOME.*?:|WORK.*?:/
    $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
    if ture != 1
      $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
    end
    if ture != 1
      $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
    end
    if ture != 1
      $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
    end
    if ture != 1
      $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
    end
    if ture != 1
      $label_home,ture = settype(vcfdata,$label_home,"HOME")
    end
    if ture != 1
      $label_work,ture = settype(vcfdata,$label_work,"WORK")
    end
  end
  if vcfdata =~ /DOM.*?/
    $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
    if ture != 1
      $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
    end
    if ture != 1
```

```
    $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
  end
  if ture != 1
    $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
  end
  if ture != 1
    $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
  end
  if ture != 1
    $label_home,ture = settype(vcfdata,$label_home,"HOME")
  end
  if ture != 1
    $label_work,ture = settype(vcfdata,$label_work,"WORK")
  end
end
if vcfdata =~ /INTL.*?/
  $label_dom,ture = settype(vcfdata,$label_intl,"INTL")
  if ture != 1
    $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
  end
  if ture != 1
    $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
  end
  if ture != 1
    $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
  end
  if ture != 1
    $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
  end
  if ture != 1
    $label_home,ture = settype(vcfdata,$label_home,"HOME")
  end
  if ture != 1
    $label_work,ture = settype(vcfdata,$label_work,"WORK")
  end
end
if vcfdata =~ /POSTAL.*?/
  $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
  if ture != 1
    $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
  end
  if ture != 1
    $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
```

```
  end
  if ture != 1
    $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
  end
  if ture != 1
    $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
  end
  if ture != 1
    $label_home,ture = settype(vcfdata,$label_home,"HOME")
  end
  if ture != 1
    $label_work,ture = settype(vcfdata,$label_work,"WORK")
  end
end
if vcfdata =~ /PARCEL.*?/
  $label_parcel,ture = settype(vcfdata,$label_parCel,"PARCEL")
  if ture != 1
    $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
  end
  if ture != 1
    $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
  end
  if ture != 1
    $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
  end
  if ture != 1
    $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
  end
  if ture != 1
    $label_home,ture = settype(vcfdata,$label_home,"HOME")
  end
  if ture != 1
    $label_work,ture = settype(vcfdata,$label_work,"WORK")
  end
end
if vcfdata =~ /HOME.*?/
  $label_home,ture = settype(vcfdata,$label_home,"HOME")
  if ture != 1
    $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
  end
  if ture != 1
    $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
  end
```

```
    if ture != 1
      $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
    end
    if ture != 1
      $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
    end
    if ture != 1
      $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
    end
    if ture != 1
      $label_work,ture = settype(vcfdata,$label_work,"WORK")
    end
  end
  if vcfdata =~ /WORK.*?/
    $label_work,ture = settype(vcfdata,$label_work,"WORK")
    if ture != 1
      $label_pref,ture = settype(vcfdata,$label_pref,"PREF")
    end
    if ture != 1
      $label_dom,ture = settype(vcfdata,$label_dom,"DOM")
    end
    if ture != 1
      $label_intl,ture = settype(vcfdata,$label_intl,"INTL")
    end
    if ture != 1
      $label_postal,ture = settype(vcfdata,$label_postal,"POSTAL")
    end
    if ture != 1
      $label_parcel,ture = settype(vcfdata,$label_parcel,"PARCEL")
    end
    if ture != 1
      $label_home,ture = settype(vcfdata,$label_home,"HOME")
    end
  end
when /^TEL/
  if vcfdata != /PREF.*?:|HOME.*?:|MSG.*?|WORK.*?|
                VOICE.*?:|FAX.*?:|CELL.*?:|VIDEO.*?:|
                PAGER.*?:|BBS.*?:|MODEM.*?:|CAR.*?:|
                ISDN.*?:|PCS.*?:/
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
    if ture != 1
      $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
    end
```

```
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /HOME.*?:/
  $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
```

```
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /MSG.*?:/
  $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
```

```
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /WORK.*?:/
  $tel_work,ture = settype(vcfdata,$tel_work,"WORK")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
```

```
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /VOICE.*?:/
  $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
```

```
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /FAX.*?:/
  $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
```

```
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
 if vcfdata =~ /CELL.*?:/
  $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
```

```
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /VIDEO.*?:/
  $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
```

```
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /PAGER.*?:/
  $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
```

```
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /BBS.*?:/
  $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
```

```
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /MODEM.*?:/
  $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
```

```
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /CAR.*?:/
  $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
```

```
  end
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /ISDN.*?:/
  $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
```

```
  if ture != 1
    $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
  end
  if ture != 1
    $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
  end
  if ture != 1
    $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
  end
  if ture != 1
    $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
  end
  if ture != 1
    $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
  end
  if ture != 1
    $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
  end
  if ture != 1
    $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
  end
  if ture != 1
    $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  end
end
if vcfdata =~ /PCS.*?:/
  $tel_pcs,ture = settype(vcfdata,$tel_pcs,"PCS")
  if ture != 1
    $tel_pref,ture = settype(vcfdata,$tel_pref,"PREF")
  end
  if ture != 1
    $tel_home,ture = settype(vcfdata,$tel_home,"HOME")
  end
  if ture != 1
    $tel_msg,ture = settype(vcfdata,$tel_msg,"MSG")
  end
  if ture != 1
    $tel_work,turre = settype(vcfdata,$tel_work,"WORK")
  end
  if ture != 1
    $tel_voice,ture = settype(vcfdata,$tel_voice,"VOICE")
  end
  if ture != 1
```

```
      $tel_fax,ture = settype(vcfdata,$tel_fax,"FAX")
    end
    if ture != 1
      $tel_cell,ture = settype(vcfdata,$tel_cell,"CELL")
    end
    if ture != 1
      $tel_video,ture = settype(vcfdata,$tel_video,"VIDEO")
    end
    if ture != 1
      $tel_pager,ture = settype(vcfdata,$tel_pager,"PAGER")
    end
    if ture != 1
      $tel_bbs,ture = settype(vcfdata,$tel_bbs,"BBS")
    end
    if ture != 1
      $tel_modem,ture = settype(vcfdata,$tel_modem,"MODEM")
    end
    if ture != 1
      $tel_car,ture = settype(vcfdata,$tel_car,"CAR")
    end
    if ture != 1
      $tel_isdn,ture = settype(vcfdata,$tel_isdn,"ISDN")
    end
  end
when /^EMAIL/
  if vcfdata !~ /INTERNET.*?:|HOME.*?:|WORK.*?:|X400.*?:/
    vcfdata =~ /.*?:/
    $email_pref,ture = settype(vcfdata,$email_pref,"PREF")
    if ture != 1
      $email_internet,ture = settype(vcfdata,$email_internet,"INTERNET")
    end
    if ture != 1
      $email_home,ture = settype(vcfdata,$email_home,"HOME")
    end
    if ture != 1
      $email_work,ture = settype(vcfdata,$email_work,"WORK")
    end
    if ture != 1
      $email_x400,ture = settype(vcfdata,$email_x400,"X400")
    end
  end
  if vcfdata =~ /INTERNET.*?:/
    $email_internet,ture = settype(vcfdata,$email_internet,"INTERNET")
```

```
  if ture != 1
    $email_pref,ture = settype(vcfdata,$email_pref,"PREF")
  end
  if ture != 1
    $email_home,ture = settype(vcfdata,$email_home,"HOME")
  end
  if ture != 1
    $email_work,ture = settype(vcfdata,$email_work,"WORK")
  end
  if ture != 1
    $email_x400,ture = settype(vcfdata,$email_x400,"X400")
  end
end
if vcfdata =~ /HOME.*?:/
  $email_home,ture = settype(vcfdata,$email_home,"HOME")
  if ture != 1
    $email_pref,ture = settype(vcfdata,$email_pref,"PREF")
  end
  if ture != 1
    $email_internet,ture = settype(vcfdata,$email_internet,"INTERNET")
  end
  if ture != 1
    $email_work,ture = settype(vcfdata,$email_work,"WORK")
  end
  if ture != 1
    $email_x400,ture = settype(vcfdata,$email_x400,"X400")
  end
end
if vcfdata =~ /WORK.*?:/
  $email_work,ture = settype(vcfdata,$email_work,"WORK")
  if ture != 1
    $email_pref,ture = settype(vcfdata,$email_pref,"PREF")
  end
  if ture != 1
    $email_internet,ture = settype(vcfdata,$email_internet,"INTERNET")
  end
  if ture != 1
    $email_home,ture = settype(vcfdata,$email_home,"HOME")
  end
  if ture != 1
    $email_x400,ture = settype(vcfdata,$email_x400,"X400")
  end
end
```

```
  if vcfdata =~ /X400.*?:/
    $email_x400,ture = settype(vcfdata,$email_x400,"X400")
    if ture != 1
      $email_pref,ture = settype(vcfdata,$email_pref,"PREF")
    end
    if ture != 1
      $email_internet,ture = settype(vcfdata,$email_internet,"INTERNET")
    end
    if ture != 1
      $email_home,ture = settype(vcfdata,$email_home,"HOME")
    end
    if ture != 1
      $email_work,ture = settype(vcfdata,$email_work,"WORK")
    end
  end
when /^MAILER.*?:/
  $mailer,ture = settype(vcfdata,$mailer,"MAILER")
when /^TZ.*?:/
  $tz,ture = settype(vcfdata,$tz,"TZ")
when /^GEO.*?:/
  $geo,ture = settype(vcfdata,$geo,"GEO")
when /^TITLE.*?:/
  $title,ture = settype(vcfdata,$title,"TITLE")
when /^ROLE.*?:/
  $role,ture = settype(vcfdata,$role,"ROLE")
when /^LOGO.*?:/
  $logo,ture = settype(vcfdata,$logo,"LOGO")
when /^AGENT.*?:/
  $agent,ture = settype(vcfdata,$agent,"AGENT")
when /^ORG.*?:/
  $org,ture = settype(vcfdata,$org,"ORG")
when /^CATEGORIES.*?:/
  $categories,ture = settype(vcfdata,$categories,"CATEGORIES")
when /^NOTE.*?:/
  $note,ture = settype(vcfdata,$note,"NOTE")
when /^PRODID.*?:/
  $prodid,ture = settype(vcfdata,$prodid,"PRODID")
when /^REV.*?:/
  $rev,ture = settype(vcfdata,$rev,"REV")

# vCard のバージョンが 3 の場合の振り仮名型
when /^SORT-STRING.*?:/
  $sortstring,ture = settype(vcfdata,$sortstring,"SORT-STRING")
```

```
# vCard のバージョンが 2.1 の場合の振り仮名型
when /^SOUND.*?:/
  $sound,ture = settype(vcfdata,$sound,"SOUND")

when /^UID.*?:/
  $uid,ture = settype(vcfdata,$uid,"UID")
when /^URL/
  if vcfdata =~ /HOME.*?:/
    $url_home,ture = settype(vcfdata,$url_home,"HOME")
    if ture != 1
      $url_work,ture = settype(vcfdata,$url_work,"WORK")
    end
  end
  if vcfdata =~ /WORK.*?:/
    $url_work,ture = settype(vcfdata,$url_work,"WORK")
    if ture != 1
     $url_home,ture = settype(vcfdata,$url_home,"HOME")
    end
  end
when /^CLASS.*?:/
  $class,ture = settype(vcfdata,$class,"CLASS")
when /^KEY.*?:/
  $key,ture = settype(vcfdata,$key,"KEY")
when /^END:VCARD/
  vdata += $begin + "\n"
  vdata += $fn + "\n"
  vdata += $n + "\n"
  vdata += $nickname + "\n"
  vdata += $photo + "\n"
  vdata += $bday + "\n"
  vdata += $adr_pref + "\n"
  vdata += $adr_dom + "\n"
  vdata += $adr_intl + "\n"
  vdata += $adr_postal + "\n"
  vdata += $adr_parsel + "\n"
  vdata += $adr_home + "\n"
  vdata += $adr_work + "\n"
  vdata += $label_pref + "\n"
  vdata += $label_dom + "\n"
  vdata += $label_intl + "\n"
  vdata += $label_postal + "\n"
  vdata += $label_parsel + "\n"
```

```
vdata += $label_home + "\n"
vdata += $label_work + "\n"
vdata += $tel_pref + "\n"
vdata += $tel_home + "\n"
vdata += $tel_msg + "\n"
vdata += $tel_work + "\n"
vdata += $tel_voice + "\n"
vdata += $tel_fax + "\n"
vdata += $tel_cell + "\n"
vdata += $tel_video + "\n"
vdata += $tel_pager + "\n"
vdata += $tel_bbs + "\n"
vdata += $tel_modem + "\n"
vdata += $tel_car + "\n"
vdata += $tel_isdn + "\n"
vdata += $tel_pcs + "\n"
vdata += $email_pref + "\n"
vdata += $email_internet + "\n"
vdata += $email_home + "\n"
vdata += $email_work + "\n"
vdata += $email_x400 + "\n"
vdata += $mailer + "\n"
vdata += $tz + "\n"
vdata += $geo + "\n"
vdata += $title + "\n"
vdata += $role + "\n"
vdata += $logo + "\n"
vdata += $agent + "\n"
vdata += $org + "\n"
vdata += $categories + "\n"
vdata += $note + "\n"
vdata += $prodid + "\n"
vdata += $rev + "\n"
if num == 2
  vdata += $sound + "\n"
else
  vdata += $sortstring + "\n"
end
vdata += $uid + "\n"
vdata += $url_home + "\n"
vdata += $url_work + "\n"
if num == 2
  vdata += $version + "2.1\n"
```

```
      else
        vdata += $version + "3\n"
      end
      vdata += $class + "\n"
      vdata += $key + "\n"
      vdata += $end + "\n"
      clearn
    end
  end
  return vdata
end

def els(swap,value)
  if value != ""
    if swap =~ /;TYPE=#{value}/
      swap = $‘ + $’
    elsif swap =~ /;#{value}/
      swap = $‘ + $’
    end
  else
    swap = ""
  end
  return swap
end

#各データが重複していた場合の処理メソッド
def collect(i,hikaku,*value)
  swap = ""
  for j in i+1..$contents.length
    if $contents[j] =~ /#{value[0]}.*#{value[1]}.*:/
      if $’.chomp != ""
        if hikaku.chomp != $’.chomp
          if value[0] == "NICKNAME"
            print("ニックネーム")
          elsif value[0] == "ADR"
            print("住所")
          elsif value[0] == "LABEL"
            print("配達用住所")
          elsif value[0] == "TEL"
            print("電話番号")
          elsif value[0] == "EMAIL"
            print("E-MAIL")
          elsif value[0] == "MAILER"
```

```
  print("メーラー")
elsif value[0] == "TZ"
  print("地理的な位置")
elsif value[0] == "GEO"
  print("地球上の位置")
elsif value[0] == "TITLE"
  print("役職名")
elsif value[0] == "ROLE"
  print("職業")
elsif value[0] == "LOGO"
  print("画像")
elsif value[0] == "AGENT"
  print("エージェント")
elsif value[0] == "ORG"
  print("会社名")
elsif value[0] == "CATEGORIES"
  print("カテゴリ")
elsif value[0] == "NOTE"
  print("メモ")
elsif value[0] == "PRODID"
  print("製品情報")
elsif value[0] == "REV"
  print("時間")
elsif value[0] == "URL"
  print("URL")
elsif value[0] == "CLASS"
  print("アクセス分類")
elsif value[0] == "KEY"
  print("公開鍵")
end
if value[1] == ""
  print("項目に,\n")
else
  print("項目の#{value[1]}に,\n")
end
print("1:#{hikaku}2:#{$'}")
print("の 2 つの値が検出されました。\n")
print("1 と 2, どちらが最新の情報ですか?(1,2 以外で保留)\n")
con = gets.downcase.chomp
if con == "1"
  swap = $contents[j]
elsif con == "2"
  swap = $contents[i]
```

```
          else
            print("保留します。\n")
          end
          swap = els(swap,value[1])
        else
          swap = $contents[j]
          swap = els(swap,value[1])
        end
      end
    end
    if swap != ""
      if con == "1"
        $contents[j] = swap
      elsif con == "2"
        $contents[i] = swap
      else
        $contents[j] = swap
      end
      break
    end
  end
end

# ユーザの情報を格納する配列の初期化
def infdata
  $adr.clear     # 住所
  $tel.clear     # 電話番号
  $email.clear   # E-MAIL
end

def infbdata
  $b_adr.clear
  $b_tel.clear
  $b_email.clear
end

#文字コードを変換し、backup に格納（元の文字コードが SJIS の場合）
def code_change_S(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sSx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eSx",vcf)
```

```
  when NKF::JIS
    cc = NKF.nkf("-jSx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wSx",vcf)
  else
    cc = NKF.nkf("-uSx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが EUC-JP の場合
def code_change_E(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sEx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eEx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jEx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wEx",vcf)
  else
    cc = NKF.nkf("-uEx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが JIS の場合
def code_change_J(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sJx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eJx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jJx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wJx",vcf)
  else
    cc = NKF.nkf("-uJx",vcf)
  end
  $vcard.puts cc
end
```

```
#元の文字コードが UTF-8 の場合
def code_change_U(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sWx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eWx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jWx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wWx",vcf)
  else
    cc = NKF.nkf("-uWx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが不明の場合
def code_change_A(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-ex",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wx",vcf)
  else
    cc = NKF.nkf("-ux",vcf)
  end
  $vcard.puts cc
end

#集約したアドレス帳の文字コードを調べるメソッド
def moji(vcf,name)
  case NKF.guess(vcf)
  when NKF::SJIS
    code_change_S(vcf,name)
  when NKF::EUC
    code_change_E(vcf,name)
  when NKF::JIS
```

```
    code_change_J(vcf,name)
  when NKF::UTF8
    code_change_U(vcf,name)
  else
    code_change_A(vcf,name)
  end
end

# アドレス帳集約開始

# アドレス帳を一件ずつ格納する配列の初期化
# 比較対象としてのアドレス帳を格納する配列の初期化
$contents = Array.new
b_contents = Array.new

# アドレス帳の一時避難させる変数の初期化
$stack1 = ""

# 比較が終了したアドレス帳を格納する変数の初期化
$stack2 = ""

# 各情報を格納する配列の初期化
n = 0
$adr = Array.new
$tel = Array.new
$email = Array.new
$b_adr = Array.new
$b_tel = Array.new
$b_email = Array.new

if File.exist?("backup_ver2.1.VCF")
  # バックアップファイルを開き、その内容を b_vcffile へ
  backup = File.open("backup_ver2.1.VCF")
  b_vcffile = backup.read
  backup.close
elsif File.exist?("backup_ver3.VCF")
  backup = File.open("backup_ver3.VCF")
  b_vcffile = backup.read
  backup.close
else
  b_vcffile = ""
end
```

```
# path.txt の最後のアドレス帳パスを読み込む
file = File.open("path.txt")
file.each do |$path|
end
vcffile = File.open($path.chomp)
vcfall = vcffile.read
vcffile.close

# アドレス帳を一行ずつ line へ格納
vcfall.each do |line|
  # 文字コードを JIS で統一
  case NKF.guess(line)
  when NKF::SJIS
    line = NKF.nkf("-jSx",line)
  when NKF::EUC
    line = NKF.nkf("-jEx",line)
  when NKF::JIS
    line = NKF.nkf("-jJx",line)
  when NKF::UTF8
    line = NKF.nkf("-jWx",line)
  else
    line = NKF.nkf("-jx",line)
  end

  # 名前の情報を取得
  if line =~ /^N:|^N;.*?:/
    $' =~ /.*?;.*?;/
    if $& != nil
      # セミコロンと空白の削除
      $name = $&.gsub(/;/,"").gsub(/\s+/,"")
    else
      $name = ""
    end
    # 各情報の取得
  elsif line =~ /^ADR.*HOME.*?:/ # 住所
    $adr << $'.chomp
  elsif line =~ /^TEL.*?:/ # 電話番号
    $tel << $'.chomp
  elsif line =~ /^EMAIL.*?:/ # E-MAIL
    $email << $'.chomp
  end

  # /^END/にマッチすれば比較対象を解析
```

```
if line =~ /^END/

  #比較が終了したアドレス帳を格納する変数の初期化
  $stack1 = ""

  # b_vcffile を読み込み、bline へ格納
  b_vcffile.each do |bline|
    # 文字コードを JIS で統一
    case NKF.guess(bline)
    when NKF::SJIS
      bline = NKF.nkf("-jSx",bline)
    when NKF::EUC
      bline = NKF.nkf("-jEx",bline)
    when NKF::JIS
      bline = NKF.nkf("-jJx",bline)
    when NKF::UTF8
      bline = NKF.nkf("-jWx",bline)
    else
      bline = NKF.nkf("-jx",bline)
    end

    b_contents << bline

    if bline =~ /^N:|^N;.*?:/
      $' =~ /.*?;.*?;/
      if $& != nil
        $b_name = $&.gsub(/;/,"").gsub(/\s+/,"")
      else
        $b_name = ""
      end
    elsif bline =~ /^ADR.*HOME.*?:/
      $b_adr << $'.chomp
    elsif bline =~ /^TEL.*?:/
      $b_tel << $'.chomp
    elsif bline =~ /^EMAIL.*?:/
      $b_email << $'.chomp

    # /^END/
    elsif bline =~ /^END/
      if $name == $b_name
        print($name,"が複数登録されています。\n")

        # 各情報の表示
```

```
print($name,"の情報\n")
$adr.each {|adr|
  if adr =~ /\S/
    print("住所:",adr,"\n")
  end
}
$tel.each {|tel|
  if tel =~ /\S/
    print("電話番号:",tel,"\n")
  end
}
$email.each {|email|
  if email =~ /\S/
    print("E-mail:",email,"\n")
  end
}
print($b_name,"の情報\n")
$b_adr.each {|b_adr|
  if b_adr =~ /\S/
    print("住所:",b_adr,"\n")
  end
}
$b_tel.each {|b_tel|
  if b_tel =~ /\S/
    print("電話番号:",b_tel,"\n")
  end
}
$b_email.each {|b_email|
  if b_email =~ /\S/
    print("E-mail:",b_email,"\n")
  end
}
print("これらのユーザは同一ユーザですか?( y or n )\n")

judge = gets.chomp.downcase

if judge == "y" or judge == "yes"
  print($name,"の集約を開始します。\n")

  #集約開始

  cang = ""
  con = ""
```

```
nickname = "NICKNAME"
adr = "ADR"
dom = "DOM"
intl = "INTL"
postal = "POSTSL"
parsel = "PARSEL"
home = "HOME"
work = "WORK"
label = "LABEL"
tel = "TEL"
pref = "PREF"
msg ="MSG"
voice = "VOICE"
fax = "FAX"
cell = "CELL"
video = "VIDEO"
pager = "PAGER"
bbs = "BBS"
modem = "MODEM"
car = "CAR"
isdn = "ISDN"
pcs = "PCS"
email = "EMAIL"
internet = "INTERNET"
x400 = "X400"
mailer = "MAILER"
tz = "TZ"
geo = "GEO"
title = "TITLE"
role = "ROLE"
logo = "LOGO"
agent = "AGENT"
org = "ORG"
categories = "CATEGORIES"
note = "NOTE"
prodid = "PRODID"
rev = "REV"
url = "URL"
clas = "CLASS"
key = "KEY"

sample = Array.new
sample = Marshal.load(Marshal.dump($contents))
```

```
$contents = sample | b_contents
$contents.shift
$contents.pop

for i in 0..$contents.length-1
  if $contents[i] =~ /#{nickname}:/
    #メソッド呼出
    hikaku = $'
    collect(i,hikaku,nickname)
  end
  if $contents[i] =~ /#{adr}.*#{pref}.*:/
    hikaku = $'
    collect(i,hikaku,adr,pref)
  end
  if $contents[i] =~ /#{adr}.*#{dom}.*:/
    hikaku = $'
    collect(i,hikaku,adr,dom)
  end
  if $contents[i] =~ /#{adr}.*#{intl}.*:/
    hikaku = $'
    collect(i,hikaku,adr,intl)
  end
  if $contents[i] =~ /#{adr}.*#{postal}.*:/
    hikaku = $'
    collect(i,hikaku,adr,postal)
  end
  if $contents[i] =~ /#{adr}.*#{parsel}.*:/
    hikaku = $'
    collect(i,hikaku,adr,parsel)
  end
  if $contents[i] =~ /#{adr}.*#{home}.*:/
    hikaku = $'
    collect(i,hikaku,adr,home)
  end
  if $contents[i] =~ /#{adr}.*#{work}.*:/
    hikaku = $'
    collect(i,hikaku,adr,work)
  end
  if $contents[i] =~ /#{label}.*#{pref}.*:/
    hikaku = $'
    collect(i,hikaku,label,pref)
  end
  if $contents[i] =~ /#{label}.*#{dom}.*:/
```

```
  hikaku = $'
  collect(i,hikaku,label,dom)
end
if $contents[i] =~ /#{label}.*#{intl}.*:/
  hikaku = $'
  collect(i,hikaku,label,intl)
end
if $contents[i] =~ /#{label}.*#{postal}.*:/
  hikaku = $'
  collect(i,hikaku,label,postal)
end
if $contents[i] =~ /#{label}.*#{parsel}.*:/
  hikaku = $'
  collect(i,hikaku,label,parsel)
end
if $contents[i] =~ /#{label}.*#{home}.*:/
  hikaku = $'
  collect(i,hikaku,label,home)
end
if $contents[i] =~ /#{label}.*#{work}.*:/
  hikaku = $'
  collect(i,hikaku,label,work)
end
if $contents[i] =~ /#{tel}.*#{pref}.*:/
  hikaku = $'
  collect(i,hikaku,tel,pref)
end
if $contents[i] =~ /#{tel}.*#{home}.*:/
  hikaku = $'
  collect(i,hikaku,tel,home)
end
if $contents[i] =~ /#{tel}.*#{msg}.*:/
  hikaku = $'
  collect(i,hikaku,tel,msg)
end
if $contents[i] =~ /#{tel}.*#{work}.*:/
  hikaku = $'
  collect(i,hikaku,tel,work)
end
if $contents[i] =~ /#{tel}.*#{voice}.*:/
  hikaku = $'
  collect(i,hikaku,tel,voice)
end
```

```
if $contents[i] =~ /#{tel}.*#{fax}.*:/
  hikaku = $'
  collect(i,hikaku,tel,fax)
end
if $contents[i] =~ /#{tel}.*#{cell}.*:/
  hikaku = $'
  collect(i,hikaku,tel,cell)
end
if $contents[i] =~ /#{tel}.*#{video}.*:/
  hikaku = $'
  collect(i,hikaku,tel,video)
end
if $contents[i] =~ /#{tel}.*#{pager}.*:/
  hikaku = $'
  collect(i,hikaku,tel,pager)
end
if $contents[i] =~ /#{tel}.*#{bbs}.*:/
  hikaku = $'
  collect(i,hikaku,tel,bbs)
end
if $contents[i] =~ /#{tel}.*#{modem}.*:/
  hikaku = $'
  collect(i,hikaku,tel,modem)
end
if $contents[i] =~ /#{tel}.*#{car}.*:/
  hikaku = $'
  collect(i,hikaku,tel,car)
end
if $contents[i] =~ /#{tel}.*#{isdn}.*:/
  hikaku = $'
  collect(i,hikaku,tel,isdn)
end
if $contents[i] =~ /#{tel}.*#{pcs}.*:/
  hikaku = $'
  collect(i,hikaku,tel,pcs)
end
if $contents[i] =~ /#{email}.*#{pref}.*:/
  hikaku = $'
  collect(i,hikaku,email,pref)
end
if $contents[i] =~ /#{email}.*#{internet}.*:/
  hikaku = $'
  collect(i,hikaku,email,internet)
```

```
end
if $contents[i] =~ /#{email}.*#{home}.*:/
  hikaku = $'
  collect(i,hikaku,email,home)
end
if $contents[i] =~ /#{email}.*#{work}.*:/
  hikaku = $'
  collect(i,hikaku,email,work)
end
if $contents[i] =~ /#{email}.*#{x400}.*:/
  hikaku = $'
  collect(i,hikaku,email,x400)
end
if $contents[i] =~ /#{mailer}:/
  hikaku = $'
  collect(i,hikaku,mailer)
end
if $contents[i] =~ /#{tz}:/
  hikaku = $'
  collect(i,hikaku,tz)
end
if $contents[i] =~ /#{geo}:/
  hikaku = $'
  collect(i,hikaku,geo)
end
if $contents[i] =~ /#{title}:/
  hikaku = $'
  collect(i,hikaku,title)
end
if $contents[i] =~ /#{role}:/
  hikaku = $'
  collect(i,hikaku,role)
end
if $contents[i] =~ /#{logo}:/
  hikaku = $'
  collect(i,hikaku,logo)
end
if $contents[i] =~ /#{agent}:/
  hikaku = $'
  collect(i,hikaku,agent)
end
if $contents[i] =~ /#{org}:/
  hikaku = $'
```

```
      collect(i,hikaku,org)
    end
    if $contents[i] =~ /#{categories}:/
      hikaku = $'
      collect(i,hikaku,categories)
    end
    if $contents[i] =~ /#{note}:/
      hikaku = $'
      collect(i,hikaku,note)
    end
    if $contents[i] =~ /#{prodid}:/
      hikaku = $'
      collect(i,hikaku,prodid)
    end
    if $contents[i] =~ /#{rev}:/
      hikaku = $'
      collect(i,hikaku,rev)
    end
    if $contents[i] =~ /#{url}.*#{home}.*:/
      hikaku = $'
      collect(i,hikaku,url,home)
    end
    if $contents[i] =~ /#{url}.*#{work}.*:/
      hikaku = $'
      collect(i,hikaku,url,work)
    end
    if $contents[i] =~ /#{clas}:/
      hikaku = $'
      collect(i,hikaku,clas)
    end
    if $contents[i] =~ /#{key}:/
      hikaku = $'
      collect(i,hikaku,key)
    end
    hikaku = ""
  end
  $contents.compact!
  $contents.unshift "BEGIN:VCARD\n"
else
  # 比較対象のアドレス帳を
  # $stack1 へ格納
  b_contents.each {|temp2|
    $stack1 += temp2
```

```
            }
          end # if judg == "y" or judg == "yes"

        else
          # 比較対象のアドレス帳を
          # $stack1 へ格納
          b_contents.each {|temp2|
            $stack1 += temp2
          }
        end # if $name == $b_name

        # 比較対象のアドレス帳を格納した配列の初期化
        b_contents.clear
        # ユーザ情報の初期化
        infbdata
      end # if bline =~ /^N.*?:/

    end # b_vcffile.each do |bline|

    # $contentes の内容を$stack2 へ
    $contents.each{|temp3|
      $stack2 += temp3
    }
    $stack2 += "END:VCARD\n"

    $contents.clear

    infdata

    b_vcffile = $stack1

  else
    $contents << line
  end # line =~ /^END/

end # vcfall.each do |line|

$stack2 += $stack1

print("アドレス帳の集約が完了しました。\n")

print("vCard 形式のデータ型へ置き換えます。\n")
```

```
# バージョン 3 と 2.1 用のアドレス帳を格納する変数の初期化
stack_ver3 = ""
stack_ver2 = ""

#setvcard メソッドの呼出

stack_ver3 = setvcard($stack2,3)
stack_ver2 = setvcard($stack2,2)

print("データ型に置き換えが完了しました。\n")

print("バックアップファイルを作成します。\n")

# backup ファイルの作成
# version3 のバックアップファイル生成
backup = File.open("backup_ver3.VCF","w")
backup.puts  stack_ver3
backup.close
# version2.1 のバックアップファイル生成
backup = File.open("backup_ver2.1.VCF","w")
backup.puts stack_ver2
backup.close

print("同期させているアドレス帳へ書き込みを開始します。\n")

name = ""

# 元のアドレス帳の文字コードに合わせる
# その際に N 型のデータを使う

pathfile = File.open("path.txt")
while pathline = pathfile.gets
  vcffile = File.open(pathline.chomp)
  while line = vcffile.gets
    if line =~ /^N:|^N;.*?:/
      name = $'.gsub(/;/,"").gsub(/\s/,"")
    elsif line =~ /^VERSION:/
      version = $'.to_i
    end
  end
  if name == ""
    name ="a"
  end
```

```
  $vcard = File.open(pathline.chomp,"w")
  if version == 2
    # 文字コード変換メソッドの呼出
    moji(stack_ver2,name)
  else
    moji(stack_ver3,name)
  end
  $vcard.close
  vcffile.close

end
pathfile.close
print("書き込みが終了しました。\n")
```

B.3 watch.rb

```
#!/usr/bin/env ruby

$KCODE = 'e'
require 'nkf'
require 'parsedate'

#元の文字コードが SJIS の場合
def code_change_S(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sSx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eSx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jSx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wSx",vcf)
  else
    cc = NKF.nkf("-uSx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが EUC-JP の場合
def code_change_E(vcf,name)
  case NKF.guess(name)
```

```
  when NKF::SJIS
    cc = NKF.nkf("-sEx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eEx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jEx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wEx",vcf)
  else
    cc = NKF.nkf("-uEx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが JIS の場合
def code_change_J(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sJx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eJx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jJx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wJx",vcf)
  else
    cc = NKF.nkf("-uJx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが UTF-8 の場合
def code_change_U(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sWx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-eWx",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jWx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wWx",vcf)
  else
```

```
    cc = NKF.nkf("-uWx",vcf)
  end
  $vcard.puts cc
end

#元の文字コードが不明の場合
def code_change_A(vcf,name)
  case NKF.guess(name)
  when NKF::SJIS
    cc = NKF.nkf("-sx",vcf)
  when NKF::EUC
    cc = NKF.nkf("-ex",vcf)
  when NKF::JIS
    cc = NKF.nkf("-jx",vcf)
  when NKF::UTF8
    cc = NKF.nkf("-wx",vcf)
  else
    cc = NKF.nkf("-ux",vcf)
  end
  $vcard.puts cc
end

#更新されたアドレス帳の文字コードを調べる
def moji(vcf,name)
  case NKF.guess(vcf)
  when NKF::SJIS
    code_change_S(vcf,name)
  when NKF::EUC
    code_change_E(vcf,name)
  when NKF::JIS
    code_change_J(vcf,name)
  when NKF::UTF8
    code_change_U(vcf,name)
  else
    code_change_A(vcf,name)
  end
end

#更新時間でソート
#文字列更新時間を標準時へ変更して比較
def time(time1,time2)
  time1 = ParseDate::parsedate(time1.to_s)
  time2 = ParseDate::parsedate(time2.to_s)
```

```
  for i in 0..5
    time = time1[i] - time2[i]
    if time > 0
      $hantei = 1
      break
    elsif time < 0
      break
    end
  end
end

pathfile = Array.new
pathtime = Array.new
difference = Array.new
time = 0

#path ファイルを開き、アドレス帳 path を pathfile へ格納
path = File.open("path.txt")
while p = path.gets
  pathfile << p.chomp
end
path.close

#格納したアドレス帳 path の最終更新時間を pathtime へ格納
for i in 0..pathfile.length - 1
  pathtime << File.mtime(pathfile[i])
end

#更新時間純にソート（この時、pathfile もソート）
for i in 0..pathtime.length - 2
  for j in i + 1..pathtime.length - 1
    $hantei = 0
    time(pathtime[i],pathtime[j])
    if $hantei == 1
      swap = pathtime[i]
      pathtime[i] = pathtime[j]
      pathtime[j] = swap

      swap = pathfile[i]
      pathfile[i] = pathfile[j]
      pathfile[j] = swap
    end
  end
```

```
end

#最終更新時間の新しいアドレス帳の更新時間を文字列型で p_t へ格納
p_t = pathtime.pop.to_s

#backup ファイルの更新時間を文字列型で b_t へ格納
if File.exist?("backup_ver2.1.vcf")
  b_t = File.mtime("backup_ver2.1.vcf").to_s
#version2.1 がなければ version3 を使用
elsif File.exist?("backup_ver3.vcf")
  b_t = File.mtime("backup_ver3.vcf").to_s
#version3 もなければ最終更新時間の古いアドレス帳の更新時間を使用
else
  b_t = pathtime.shift.to_s
end

#p_t と b_t の文字列型更新時間を標準時型で格納
p_t = ParseDate::parsedate(p_t)
b_t = ParseDate::parsedate(b_t)

#年・月・日・時・分の差分を取り、time へ加算
for i in 0..4
  difference = p_t[i] - b_t[i]
  if difference > 0
    time = 1
    break
  elsif difference < 0
    time = -1
    break
  end
end

#time > 0  ならば更新が確認されたとみなす
#つまり分単位で更新されているか確認

#time < 0 ならば、backup ファイルに変更点がみられた可能性があるので
#更新時間の古いアドレス帳の更新時間を使用
if time < 0
  b_t = ParseDate::parsedate(pathtime.shift.to_s)
  time = 0
  for i in 0..4
  difference = p_t[i] - b_t[i]
  if difference > 0
```

```
    time = 1
    break
  end
end

end

if time > 0
  print("更新が確認されました。\n")
  print("最新の情報に更新します。\n")

  vcf2 = ""
  vcf3 = ""
  path = pathfile.pop
  vcffile = File.open(path)
  while vcfline = vcffile.gets
    if vcfline =~ /^SOUND:/
      vcf3 += "SORT-STRING" + $'
      vcf2 += vcfline
    elsif vcfline =~ /^SORT-STRING:/
      vcf2 += "SOUND" + $'
      vcf3 += vcfline
    elsif vcfline =~ /^VERSION:/
      vcf2 += "VERSION:2.1\n"
      vcf3 += "VERSION:3\n"
    else
      vcf2 += vcfline
      vcf3 += vcfline
    end
  end
  vcffile.close

  ver2 = File.open("backup_ver2.1.vcf","w")
  ver2.puts vcf2
  ver2.close

  ver3 = File.open("backup_ver3.vcf","w")
  ver3.puts vcf3
  ver3.close


  name = ""
```

```
  pathfile = File.open("path.txt")
  while pathline = pathfile.gets
    vcffile = File.open(pathline.chomp)
    while line = vcffile.gets
      if line =~ /^N:|^N;.*?:/
        name = $'.gsub(/;/,"").gsub(/\s/,"")
      elsif line =~ /^VERSION:/
        version = $'.to_i
      end
    end
    if name == ""
      name ="a"
    end
    $vcard = File.open(pathline.chomp,"w")
    if version == 2
      moji(vcf2,name)
    else
      moji(vcf3,name)
    end
    $vcard.close
    vcffile.close
  end
  pathfile.close

else
  print("更新は確認されませんでした。\n")
end
```