

情報基礎シリーズ4

アルゴリズムとデータ構造 別冊2

練習問題解答

電子開発学園出版局

*この【練習問題】は、書籍『情報基礎シリーズ4 アルゴリズムとデータ構造』（発行:電子開発学園出版局／発売:株式会社 S C C）のダウンロード用【別冊】として、当該書籍の読者に限定して提供しています。

第 1 章

練習 1.1

a × b ○ c ○ d × e ○ f ×

練習 1.2

a ク b オ c イ d キ e カ f ウ

練習 1.3

a シ b ア c ウ d エ e コ f カ g キ h ケ

練習 1.4

a オ b カ c エ d オ e イ f ア

練習 1.5

a ア b イ c エ d ク e オ f キ

練習 1.6

ウ

練習 1.7

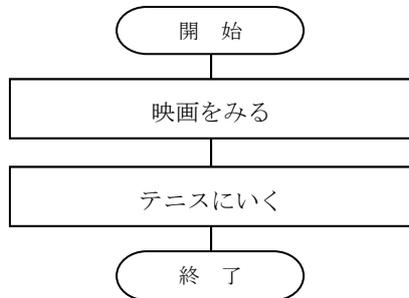
ア

練習 1.8

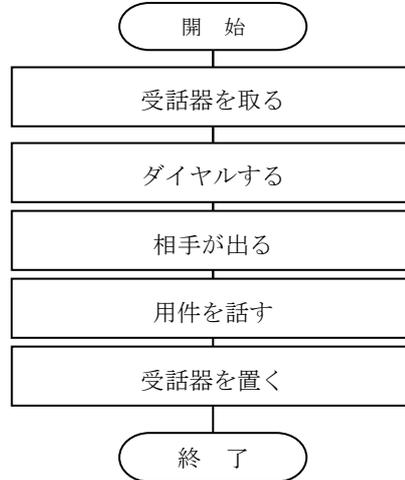
イ

第2章

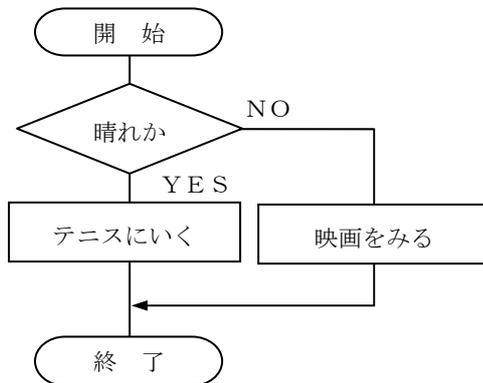
練習 2.1



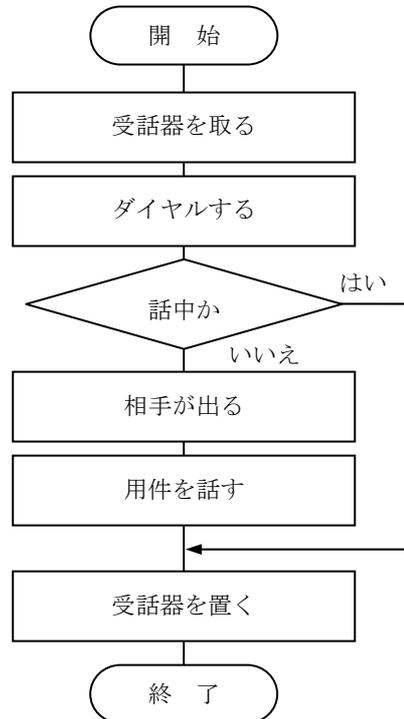
練習 2.2



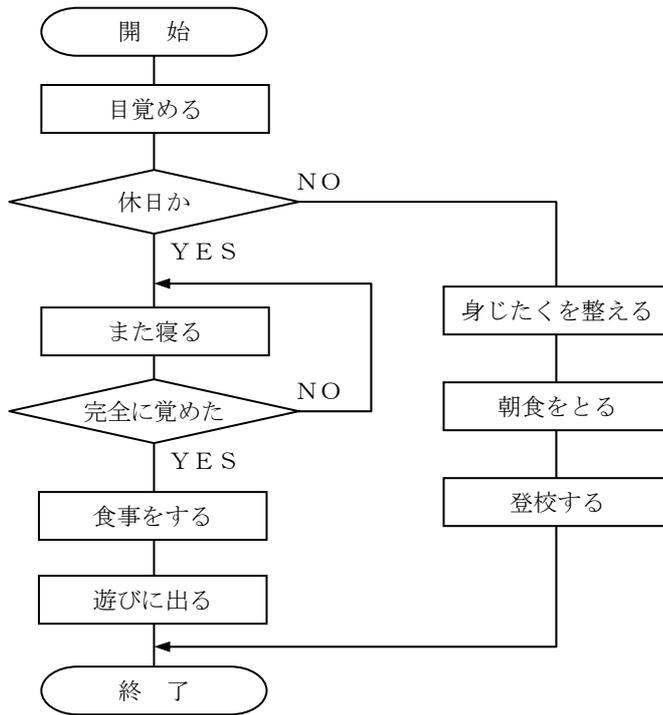
練習 2.3 (解答例)



練習 2.4



練習 2.5 (解答例)



練習 2.6

X = 400

Y = 20

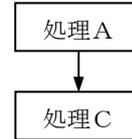
練習 2.7

X = 20

Y = 10

Z = 5

練習 2.8



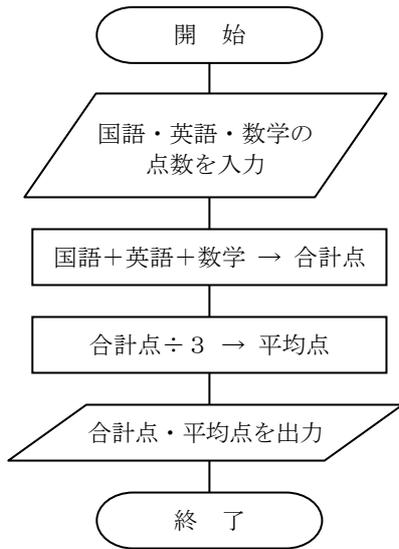
練習 2.9

- (1) 20 → 年齢
- (2) “トム” → 名前
- (3) 18 → NEN
- (4) “アイウ” → NAMAE

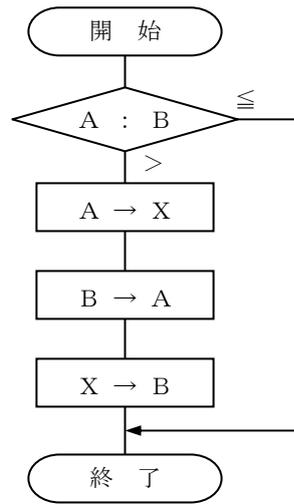
練習 2.10

処理No.	A	B	W
a	7	1	7
b	1	1	7
c	1	7	7

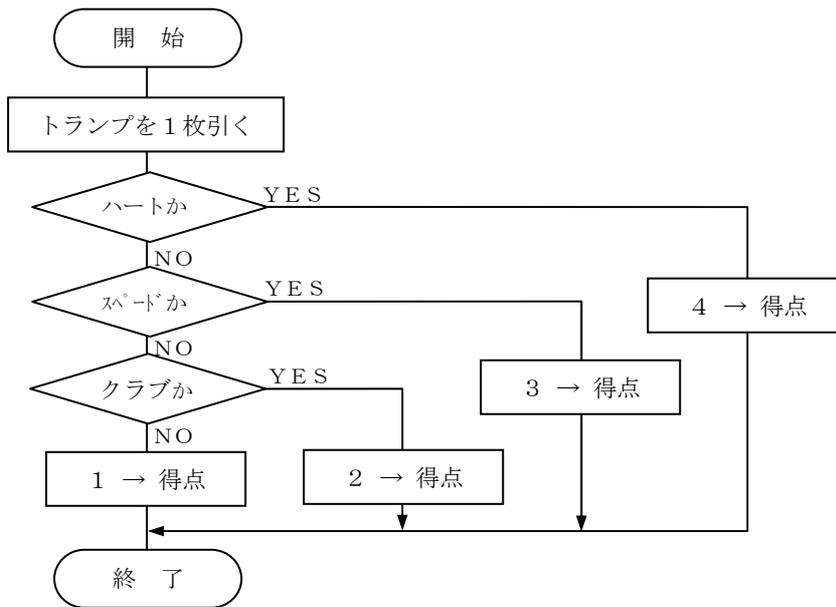
練習 2.11 (解答例)



練習 2.12 (解答例)



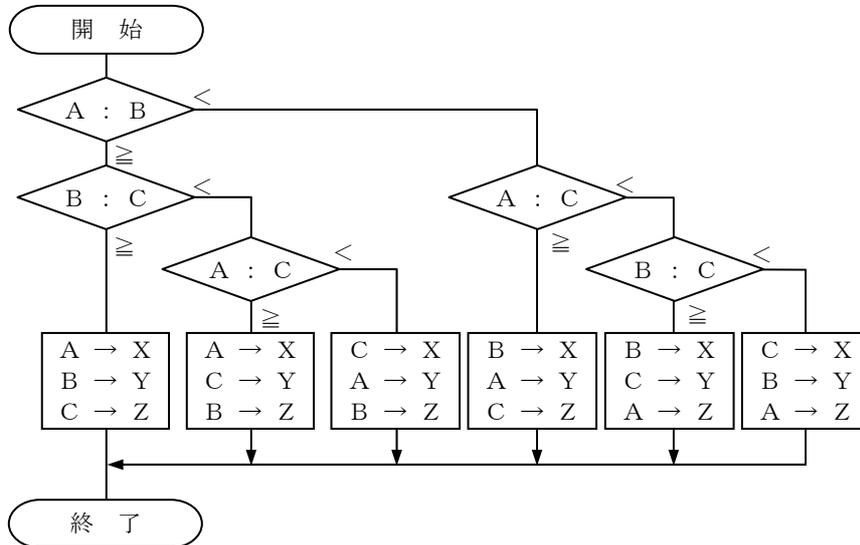
練習 2.13



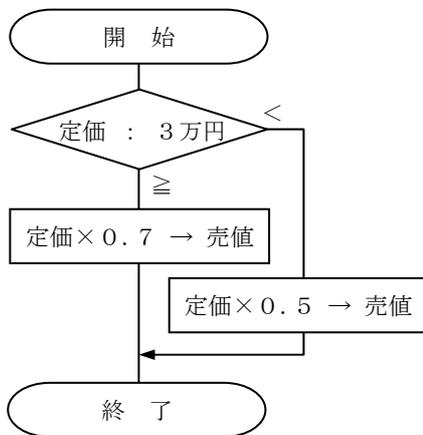
練習 2.14

b

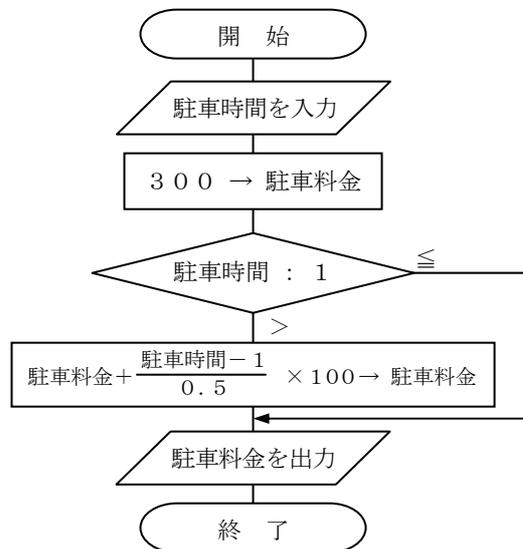
練習 2.15 (解答例)



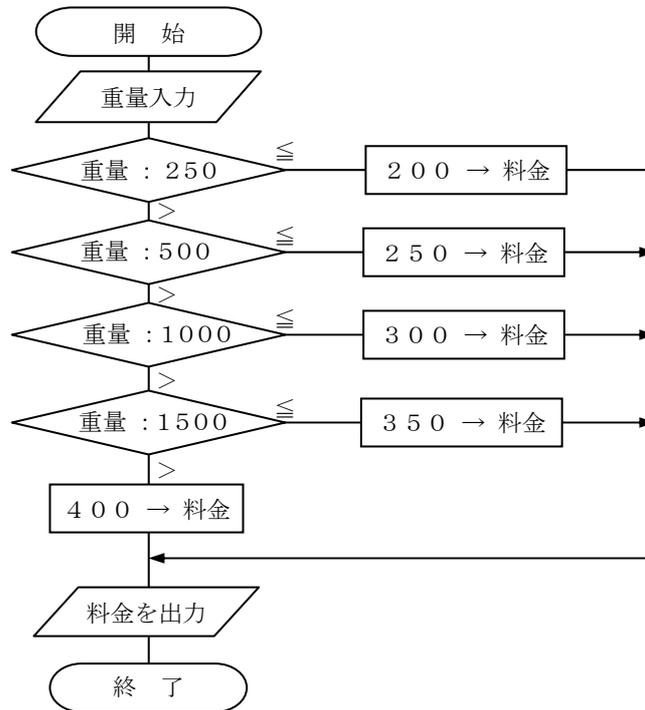
練習 2.16 (解答例)



練習 2.17 (解答例)



練習 2.18 (解答例)



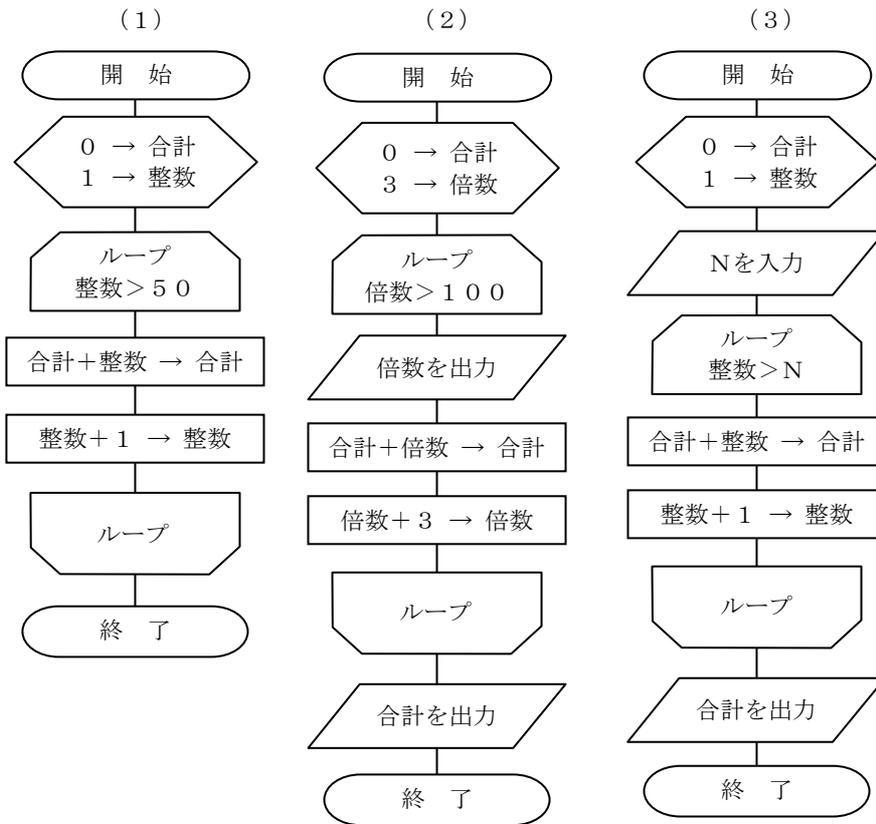
練習 2.19

b

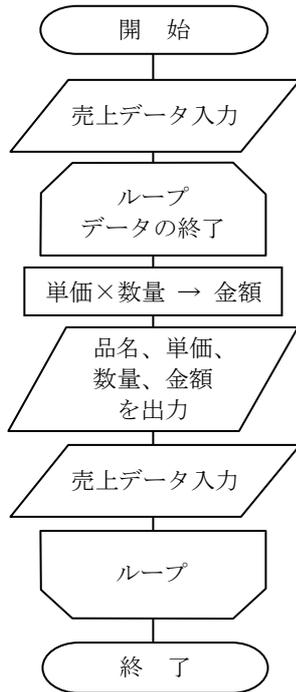
練習 2.20

- (ア) $I = 10$ (または $I > 9$ または $I \geq 10$)
- (イ) $I = 11$ (または $I > 10$ または $I \geq 11$)
- (ウ) $I = 10$ (または $I > 9$ または $I \geq 10$)
- (エ) $I = 0$ (または $I < 1$ または $I \leq 0$)
- (オ) $I = 0$ (または $I < 1$ または $I \leq 0$)

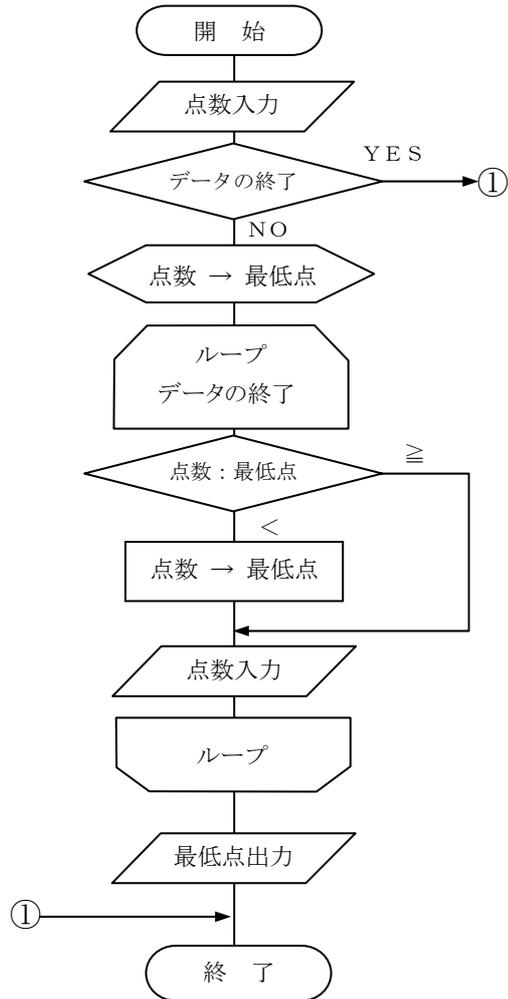
練習 2.21 (解答例)



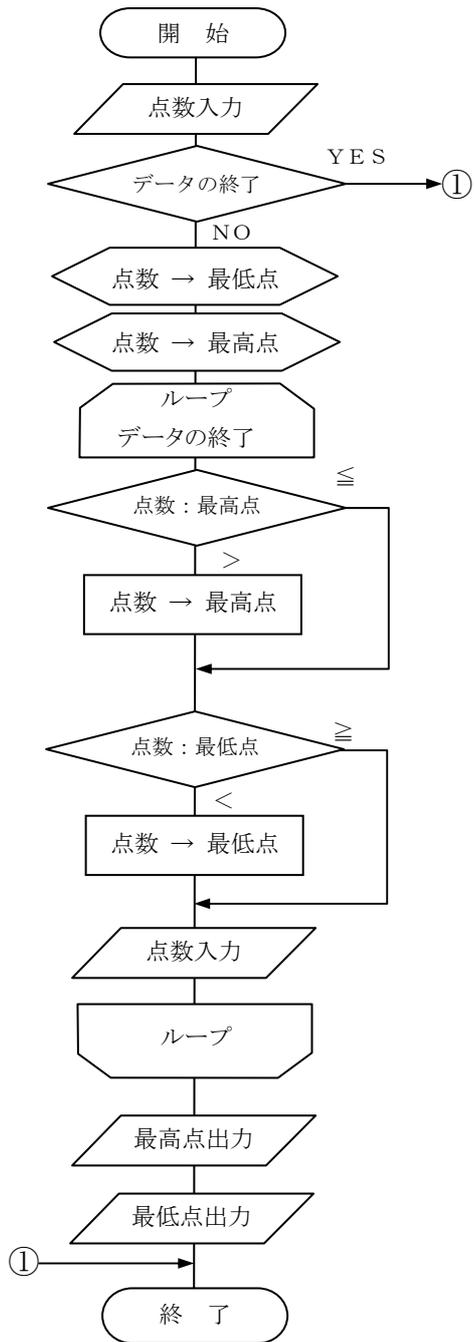
練習 2.22 (解答例)



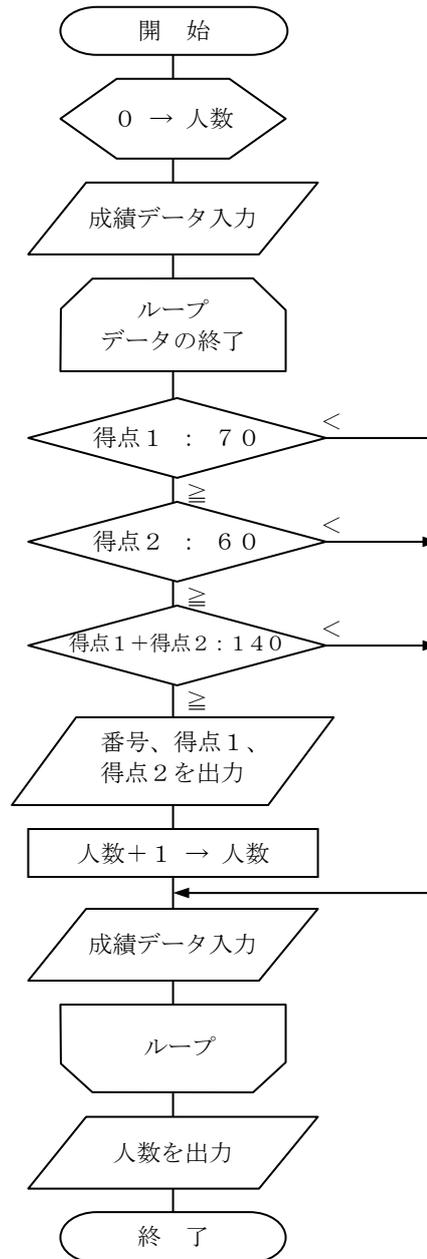
練習 2.23 (解答例)



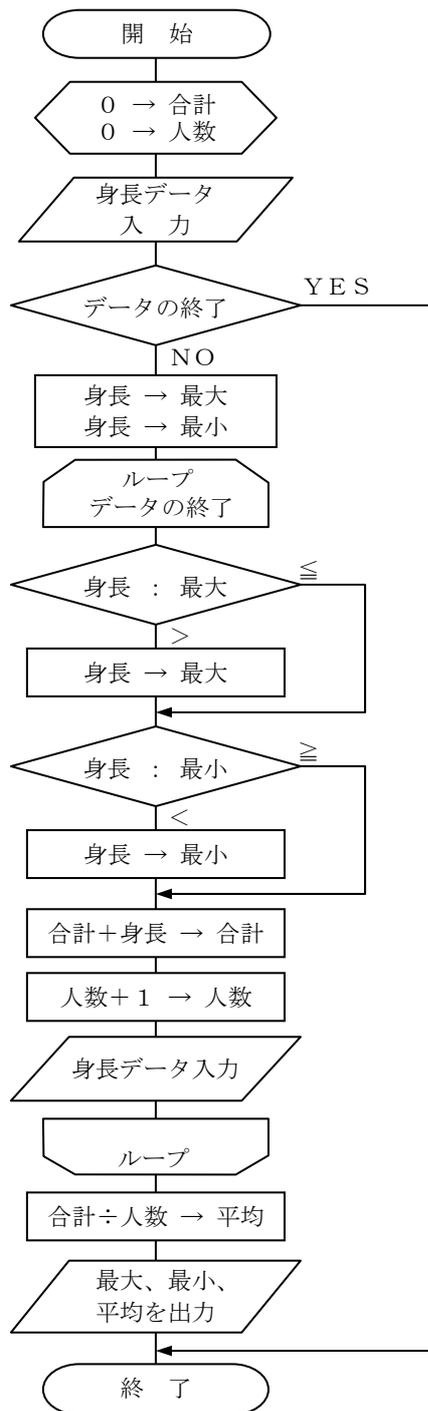
練習 2.24 (解答例)



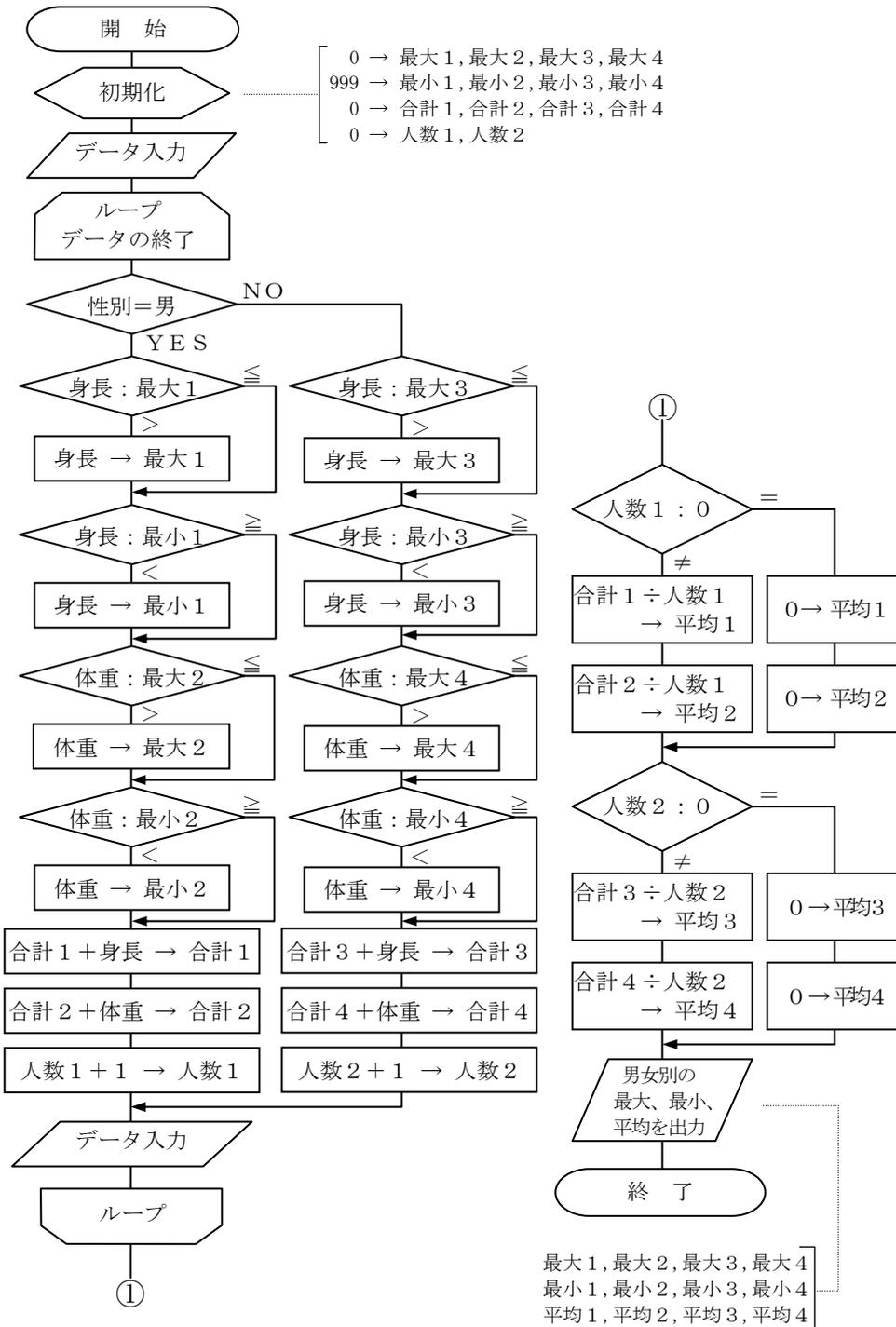
練習 2.25 (解答例)



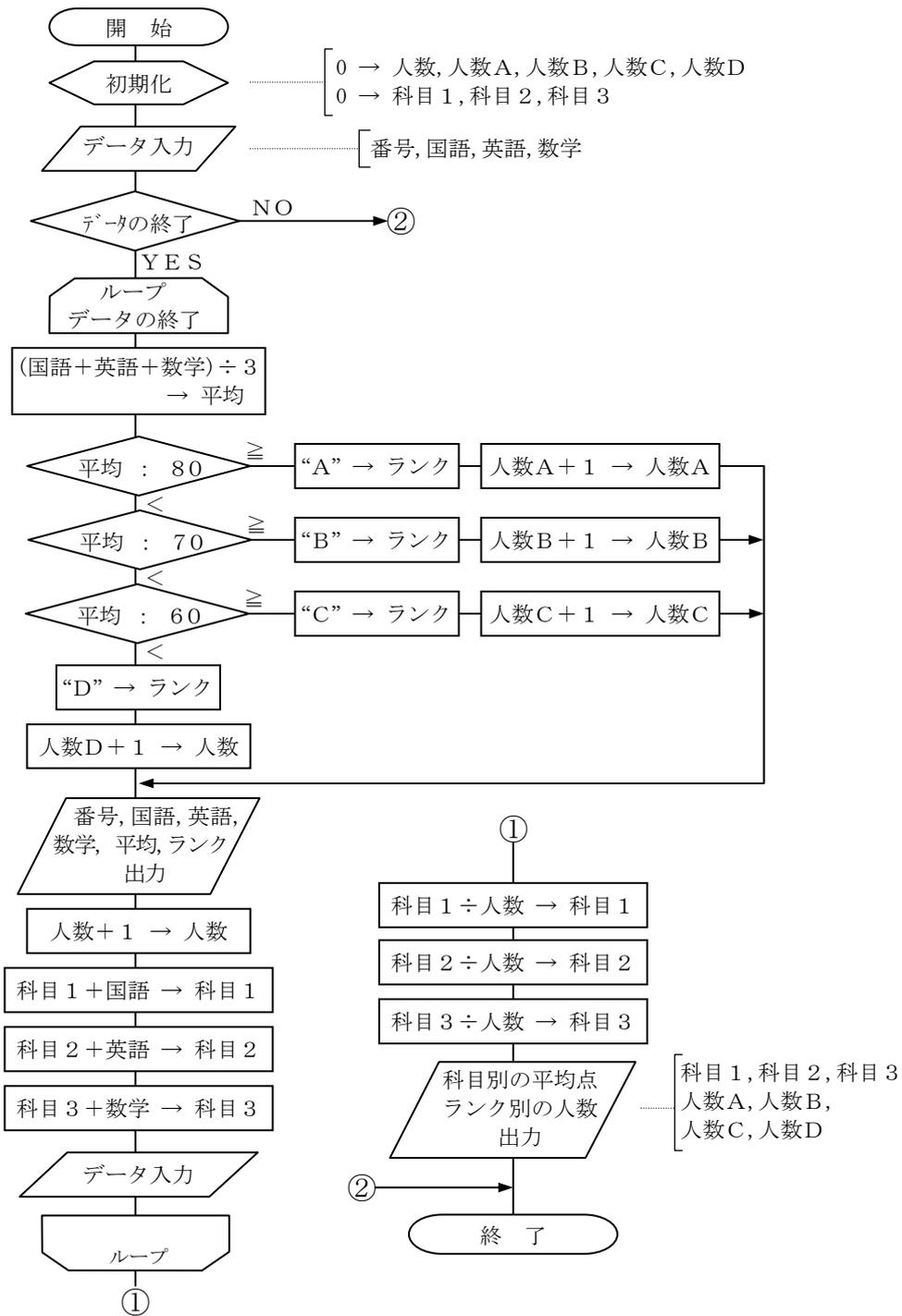
練習 2.26 (解答例)



練習 2.27 (解答例)



練習 2.28 (解答例)



練習 2.29

- (ア) $0 \rightarrow S$
- (イ) $100 \rightarrow M$
- (ウ) $N > M$
- (エ) $M - N \rightarrow M$
- (オ) $S + 1 \rightarrow S$

練習 2.30

I	C	S
1	1	1
2	1	2
3	2	4
4	3	7
5	5	12
6	8	20
7	13	33
8	21	54
9	34	88

【補足】

出力している各変数は、I が項番、C が数列の第 I 項、S が初項からの合計です。この解答から、フィボナッチ数列は、初項と第 2 項が 1 であり、その後の第 I 項 ($I \geq 3$) は、第 ($I - 2$) 項 + 第 ($I - 1$) 項 \rightarrow 第 I 項であることがわかります。流れ図を見直すと、C が計算途中では第 ($I - 1$) 項、W1 が保存時の第 ($I - 1$) 項で次の計算の第 ($I - 2$) 項であることがわかります。この数列では初期値とみなせる第 2 項を、ループの中で計算できるよう工夫されていたこともわかります。

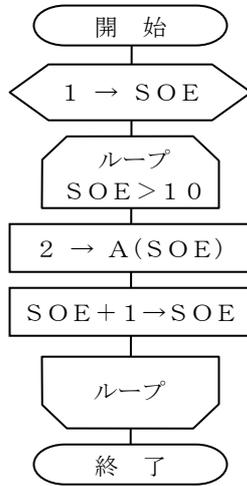
なお、初項を 0 とし、出題とは項番を +1 ずつずらした数列、値が異なっても出題と比例して変化する数列もフィボナッチ数列といえます。また、出題の S : 初項からの合計は、トレース練習用であり、自然界における意味はあまりありません。

練習 2.31

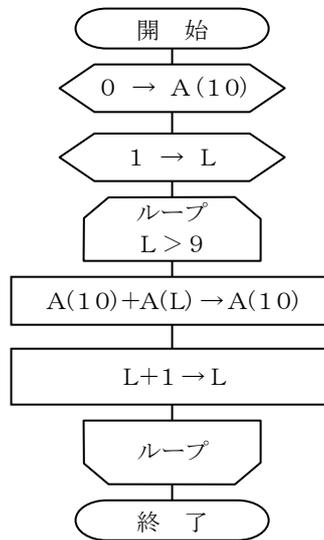
問1 3 問2 8 問3 3 問4 9 問5 6

練習 2.32 (解答例)

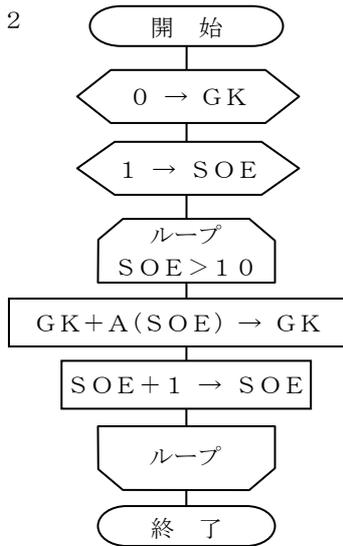
問1



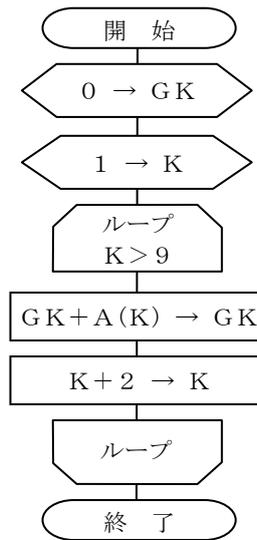
問3



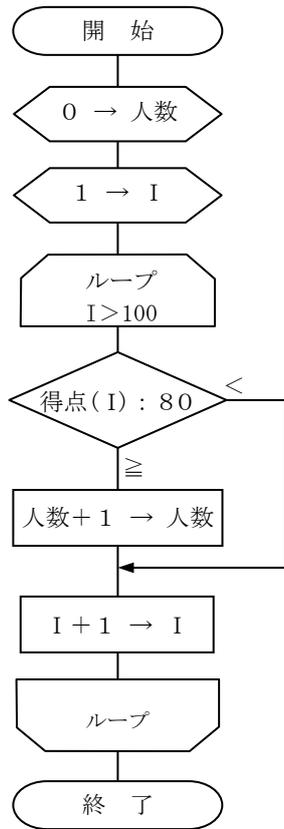
問2



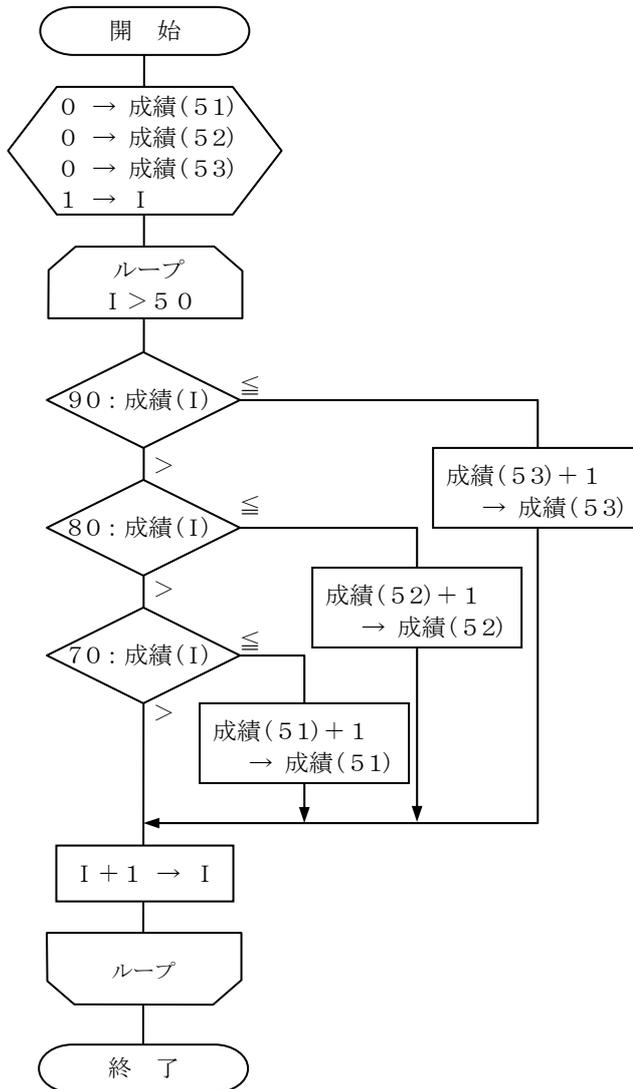
問4



練習 2.33 (解答例)

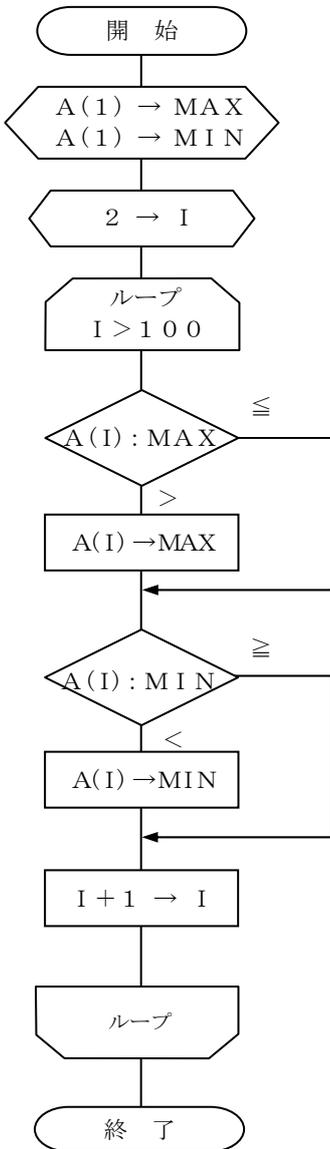


練習 2.34 (解答例)

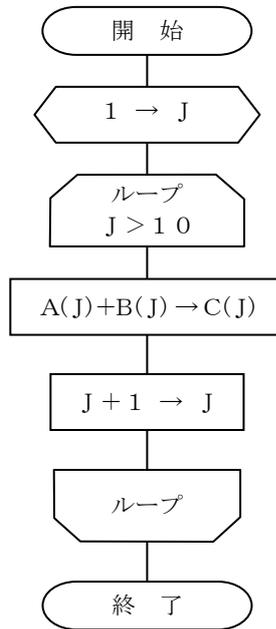


練習 2.35 (解答例)

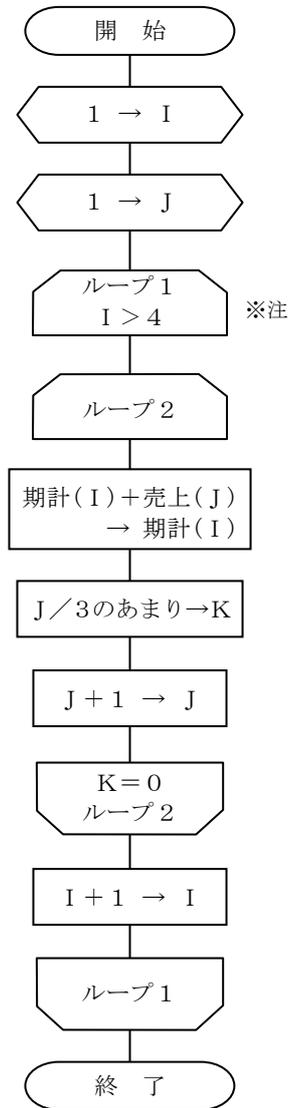
問 1



問 2



問 3



※注

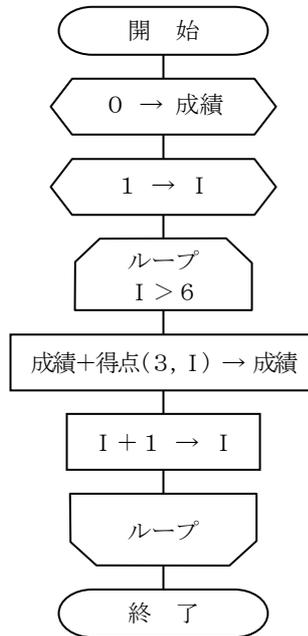
※注 : J > 12でも正解

練習 2.36

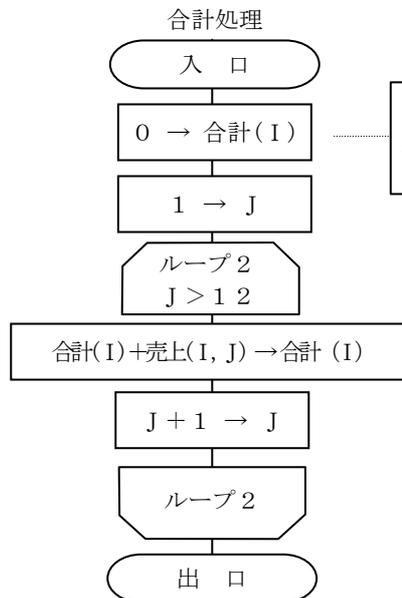
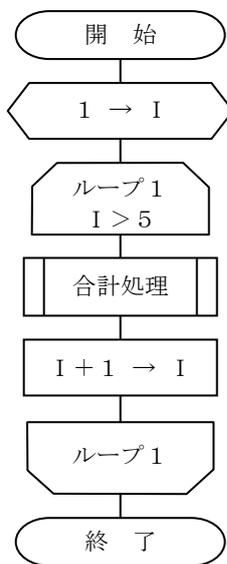
問 1

- (1) 5行 6列
- (2) 60
- (3) 112
- (4) 170

問 2 (解答例)

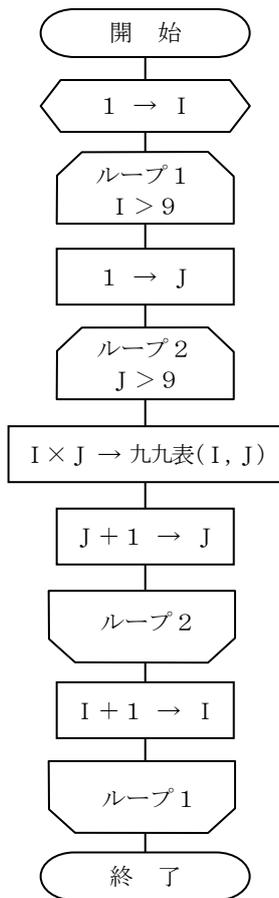


練習 2.37 (解答例)

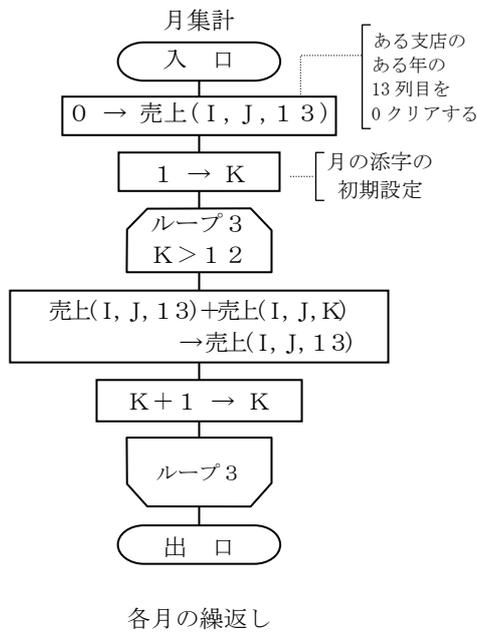
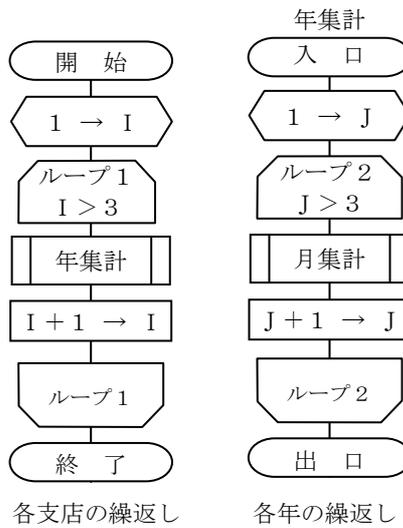


合計表の I 番目に、
I 行目の合計を格納
するため、0クリア
しておく

練習 2.38 (解答例)



練習 2.39 (解答例)



第3章

練習 3.1

X = 400

Y = 20

練習 3.2

X = 20

Y = 10

Z = 5

練習 3.3

(1)

・年齢 ← 20

(2)

・名前 ← “トム”

(3)

・NEN ← 18

(4)

・NAMAЕ ← ”アイウ”

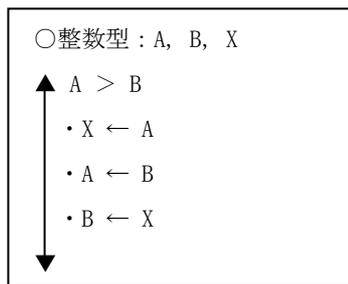
練習 3.4

処理No.	A	B	W
a	7	1	7
b	1	1	7
c	1	7	7

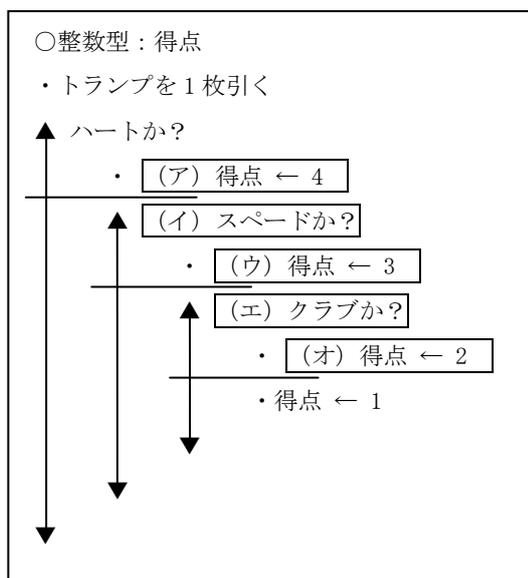
練習 3.5 (解答例)

- | |
|--|
| <ul style="list-style-type: none">○整数型：国語，英語，数学○手続：入力(国語，英語，数学) /* 国語，英語，数学の点数を入力する */○手続：出力(合計点，平均点) /* 合計点，平均点を出力する */・入力(国語，英語，数学)・合計点 ← 国語 + 英語 + 数学・平均点 ← 合計点 ÷ 3・出力(合計点，平均点) |
|--|

練習 3.6 (解答例)



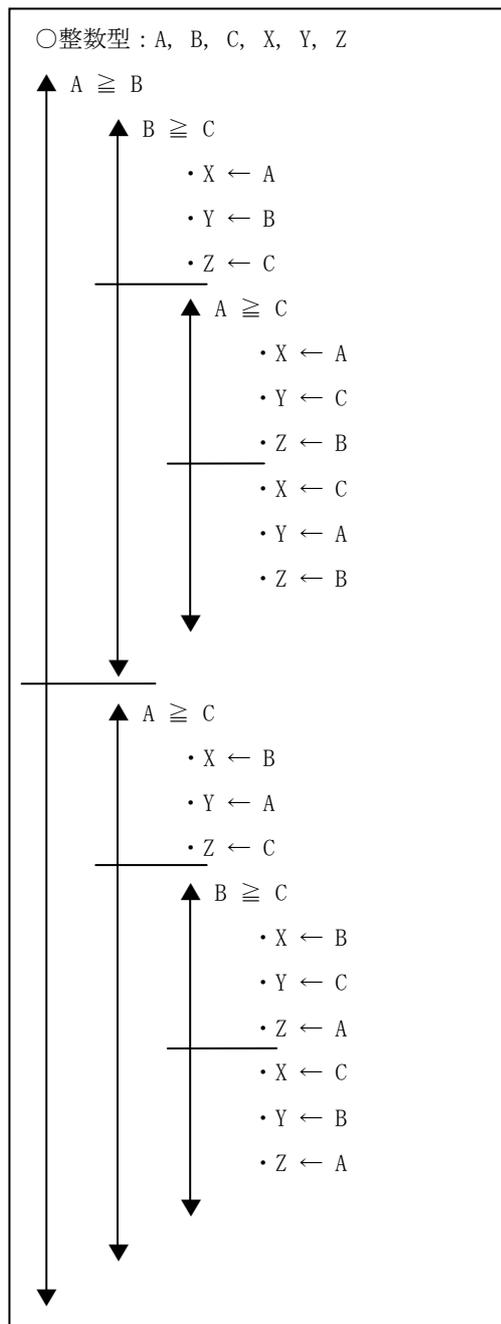
練習 3.7



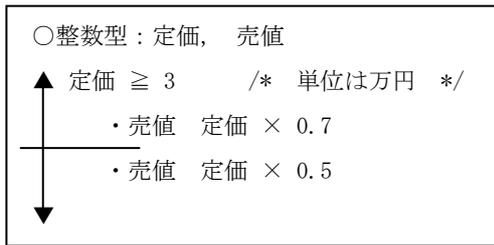
練習 3.8

b

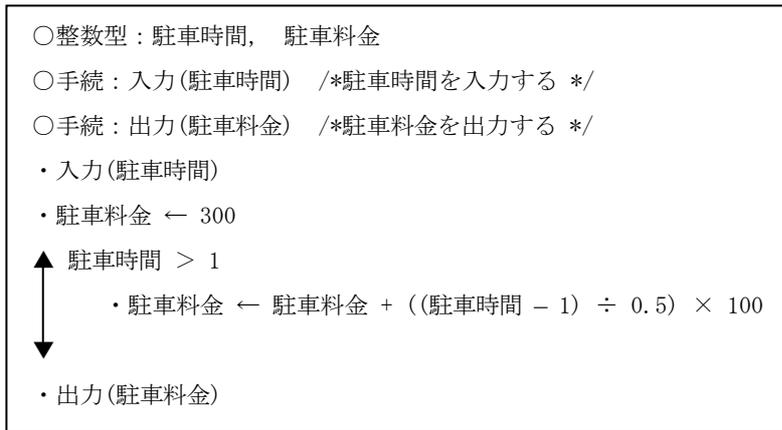
練習 3.9 (解答例)



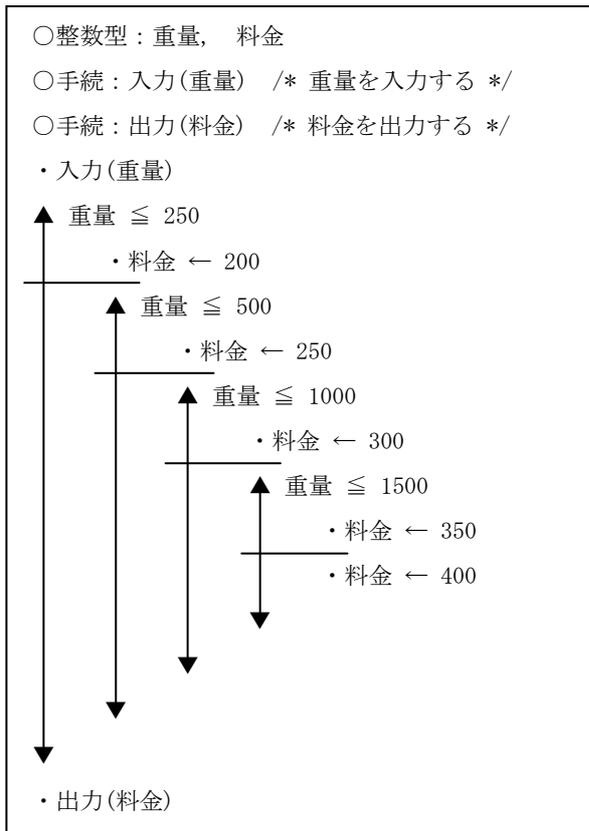
練習 3.10 (解答例)



練習 3.11 (解答例)



練習 3.12 (解答例)



練習 3.13

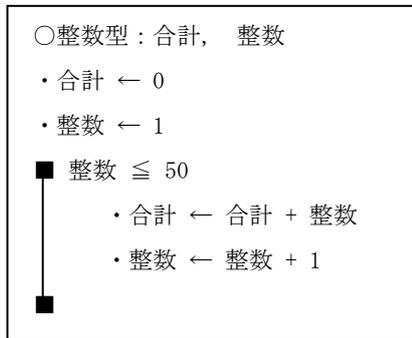
b

練習 3.14

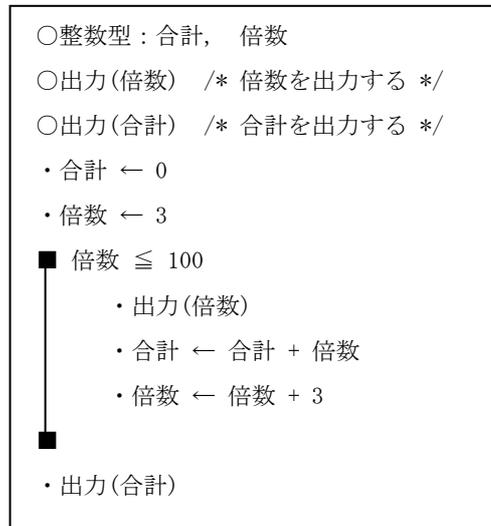
- (ア) $I \neq 10$ (または $I \leq 9$ または $I < 10$)
- (イ) $I \neq 11$ (または $I \leq 10$ または $I < 11$)
- (ウ) $I \neq 10$ (または $I \leq 9$ または $I < 10$)
- (エ) $I \neq 0$ (または $I \geq 1$ または $I > 0$)
- (オ) $I \neq 0$ (または $I \geq 1$ または $I > 0$)

練習 3.15 (解答例)

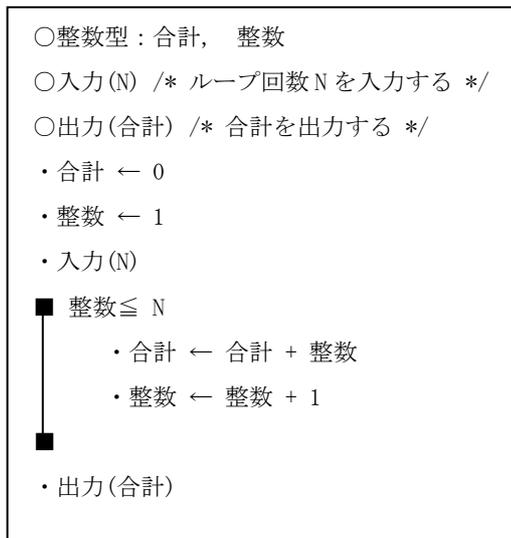
(1)



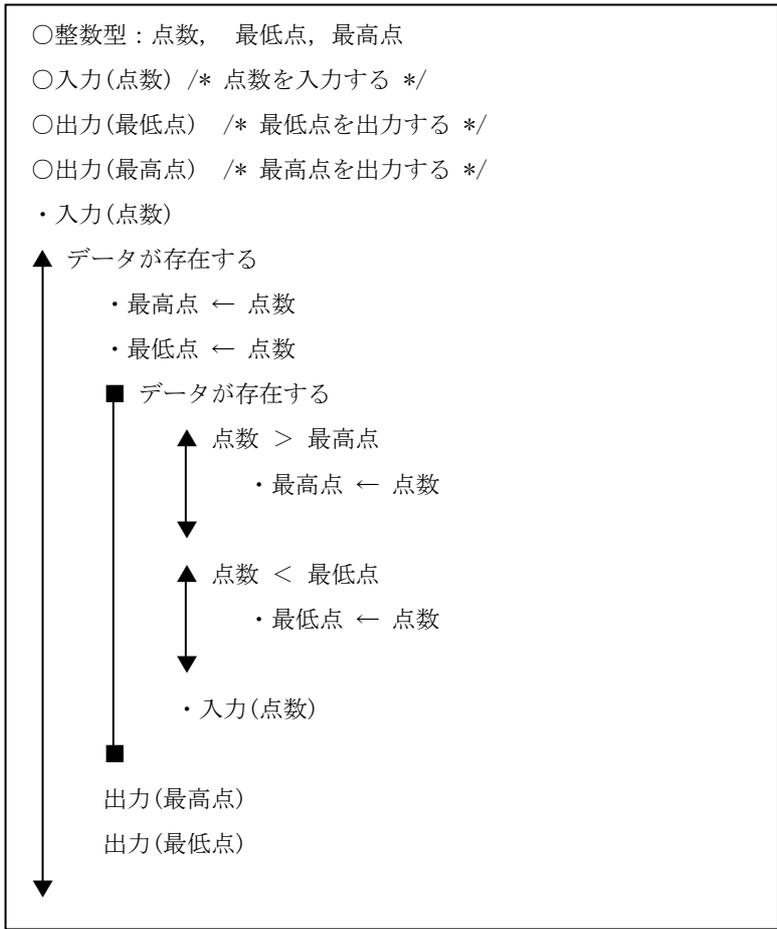
(2)



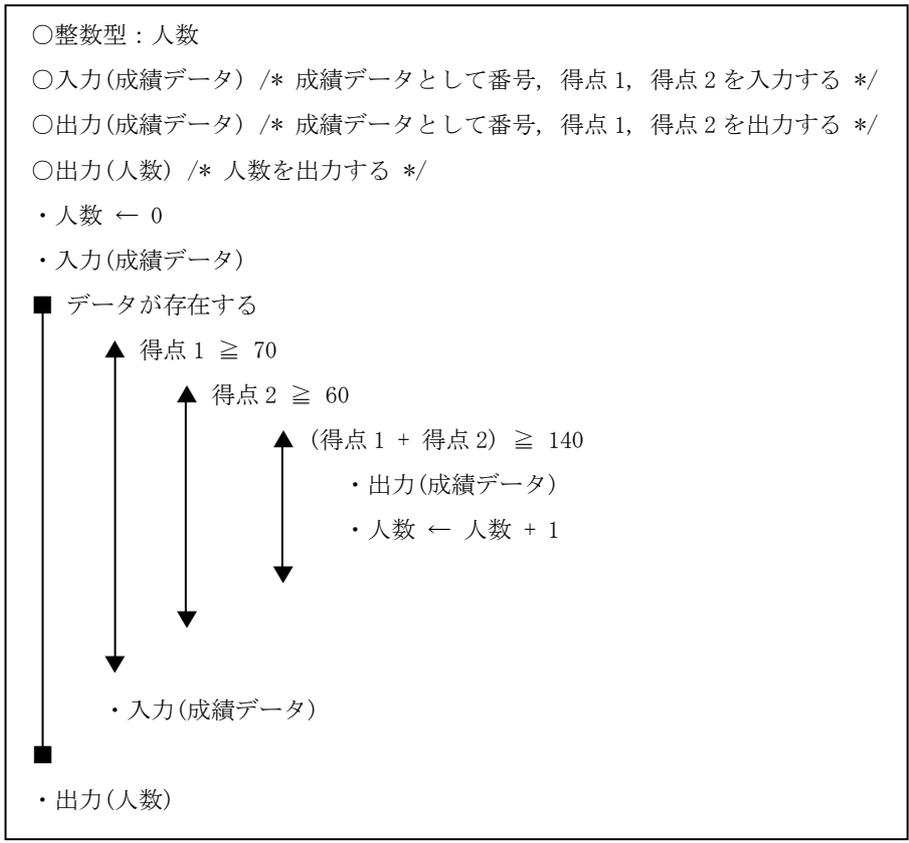
(3)



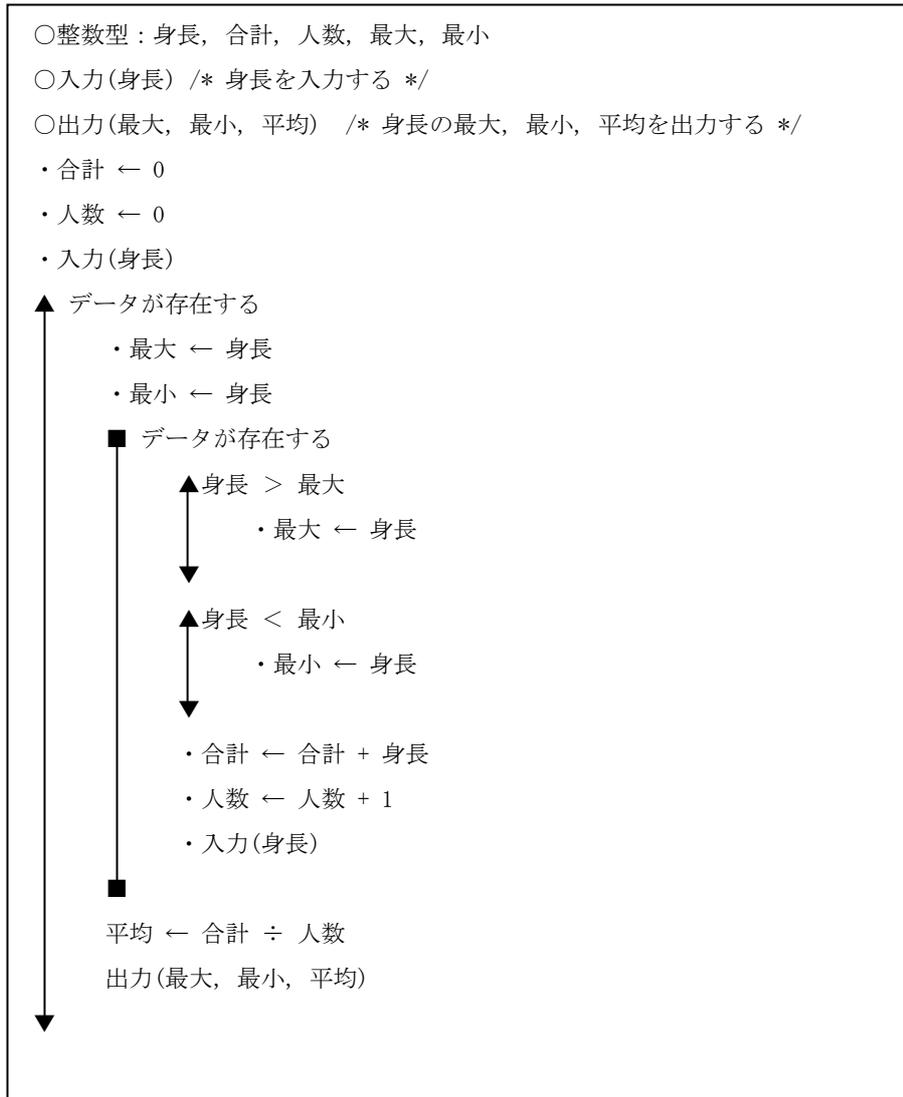
練習 3.18 (解答例)



練習 3.19 (解答例)



練習 3.20 (解答例)

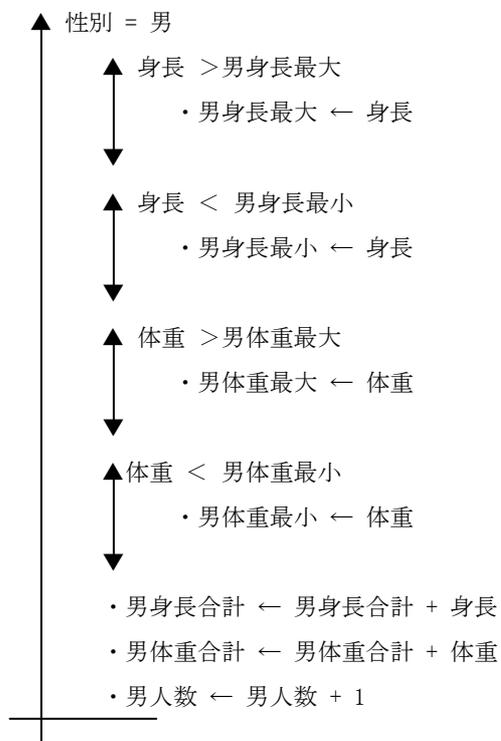


練習 3.21 (解答例)

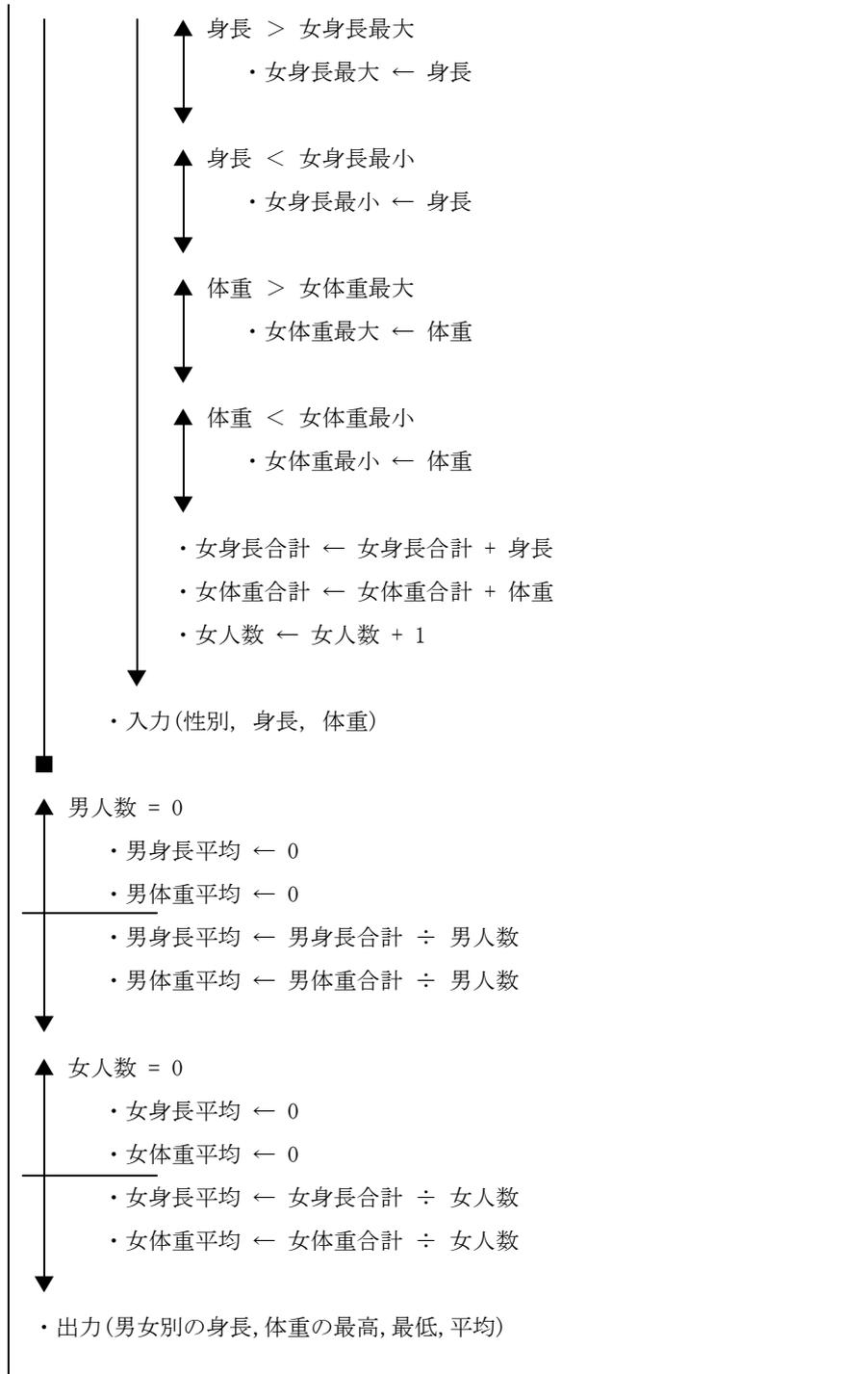
- 整数型：性別，身長，体重
- 整数型：男身長最大 = {0}，男身長最小 = {999}，男身長平均
- 整数型：男体重最大 = {0}，男体重最小 = {999}，男体重平均
- 整数型：男身長合計 = {0}，男体重合計 = {0}，男人数 = {0}
- 整数型：女身長最大 = {0}，女身長最小 = {999}，女身長平均
- 整数型：女体重最大 = {0}，女体重最小 = {999}，女体重平均
- 整数型：女身長合計 = {0}，女体重合計 = {0}，女人数 = {0}
- 入力(性別，身長，体重) /* 性別，身長，体重を入力する */
- 出力(男女別の身長，体重の最高，最低，平均) /* 男女別の身長，体重の最高，最低，平均を出力する */

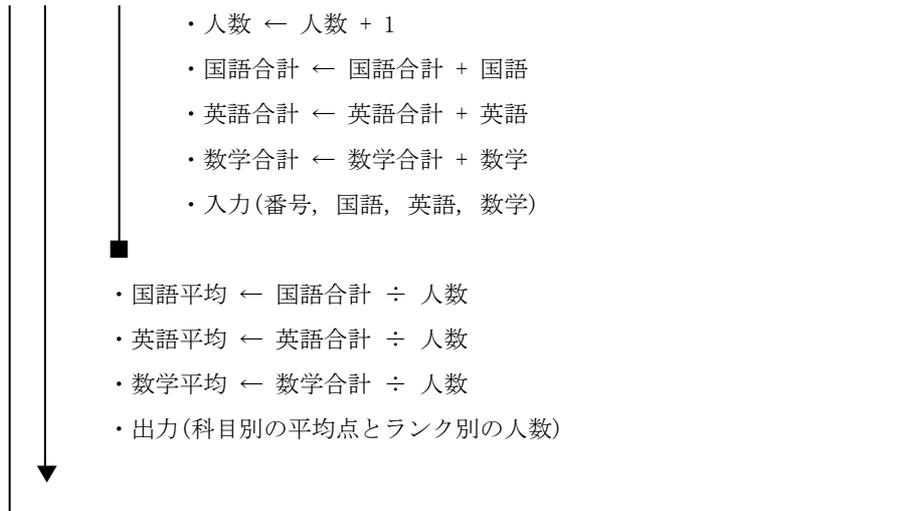
・入力(性別，身長，体重)

■ データが存在する



次ページに続く





練習 3.23

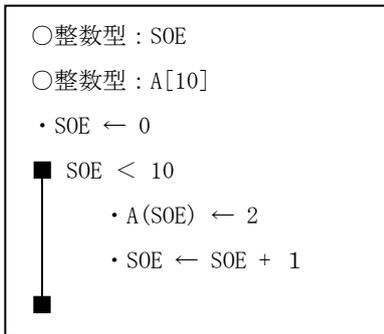
- (ア) $S \leftarrow 0$
- (イ) $M \leftarrow 100$
- (ウ) $N \leq M$
- (エ) $M \leftarrow M - N$
- (オ) $S \leftarrow S + 1$

練習 3.24

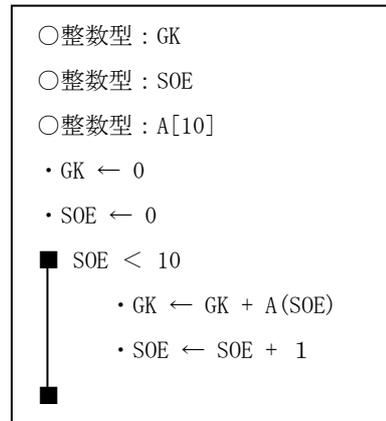
I	C	S
1	1	1
2	1	2
3	2	4
4	3	7
5	5	12
6	8	20
7	13	33
8	21	54

練習 3.25 (解答例)

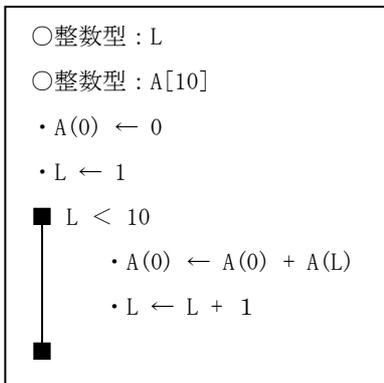
問 1



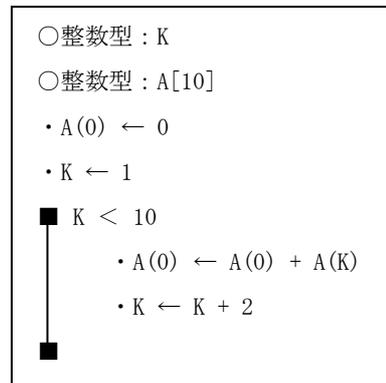
問 2



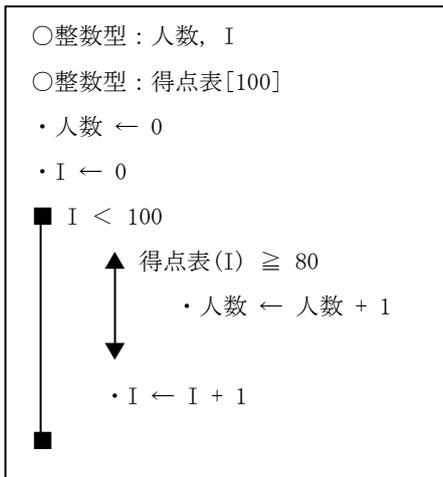
問 3



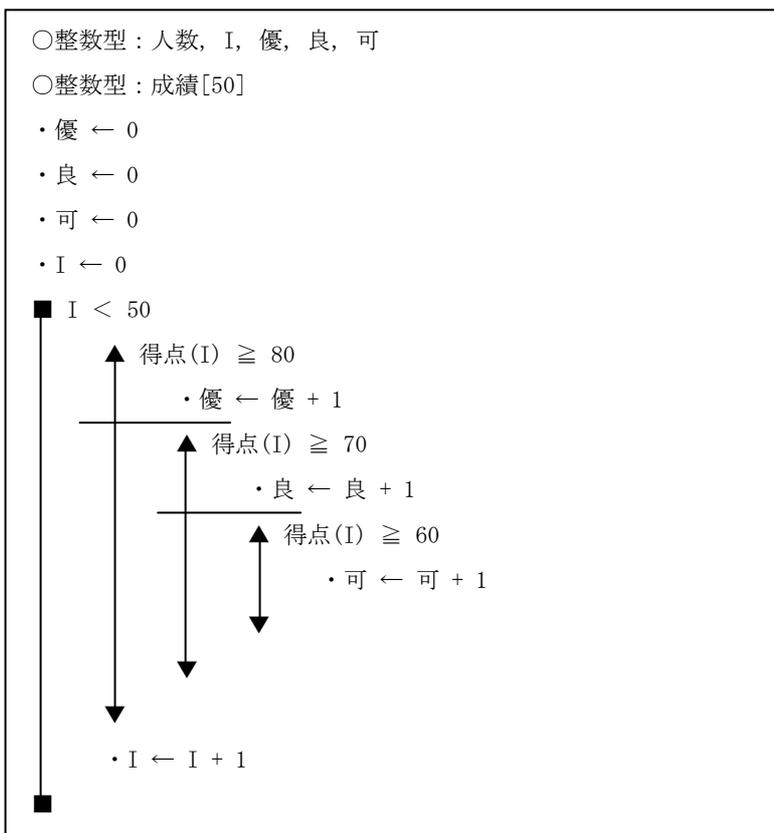
問 4



練習 3.26 (解答例)

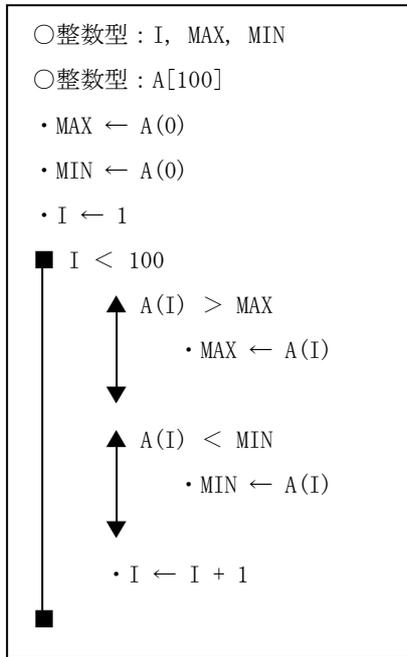


練習 3.27 (解答例)

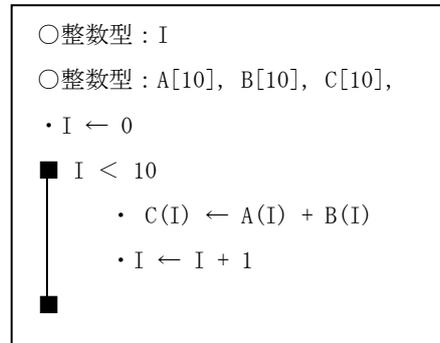


練習 3.28 (解答例)

問 1



問 2



練習 3.29 (解答例)

○整数型：最大 = {0}, 最小 = {100}, 平均, 合計 = {0}

○整数型：I, J

○整数型：得点[5, 6]

• I ← 0

■ I < 5

• J ← 0

■ J < 6

▲ 得点(I, J) > 最大

• 得点(I, J) ← 最大

▼

▲ 得点(I, J) < 最小

• 得点(I, J) ← 最小

▼

• 合計 ← 合計 + 得点(I, J)

• J ← J + 1

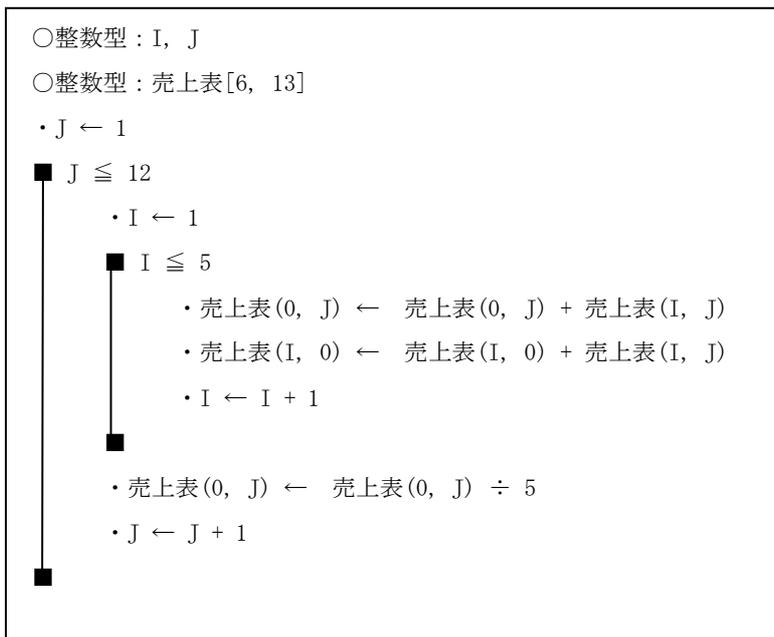
■

• I ← I + 1

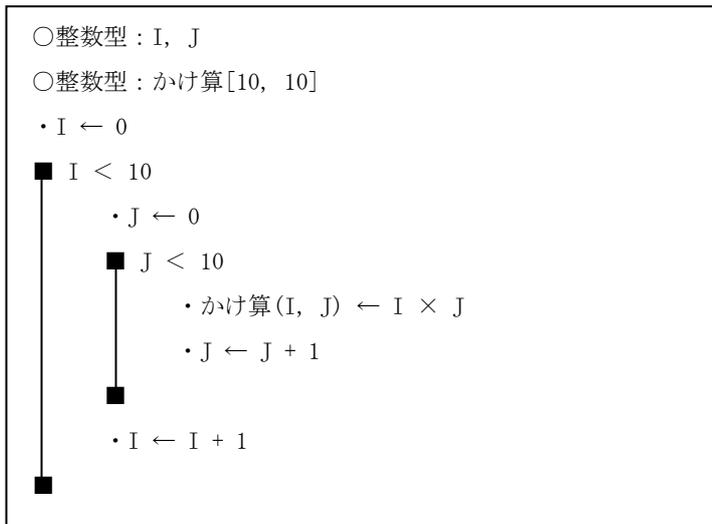
■

• 平均 ← 合計 ÷ 30

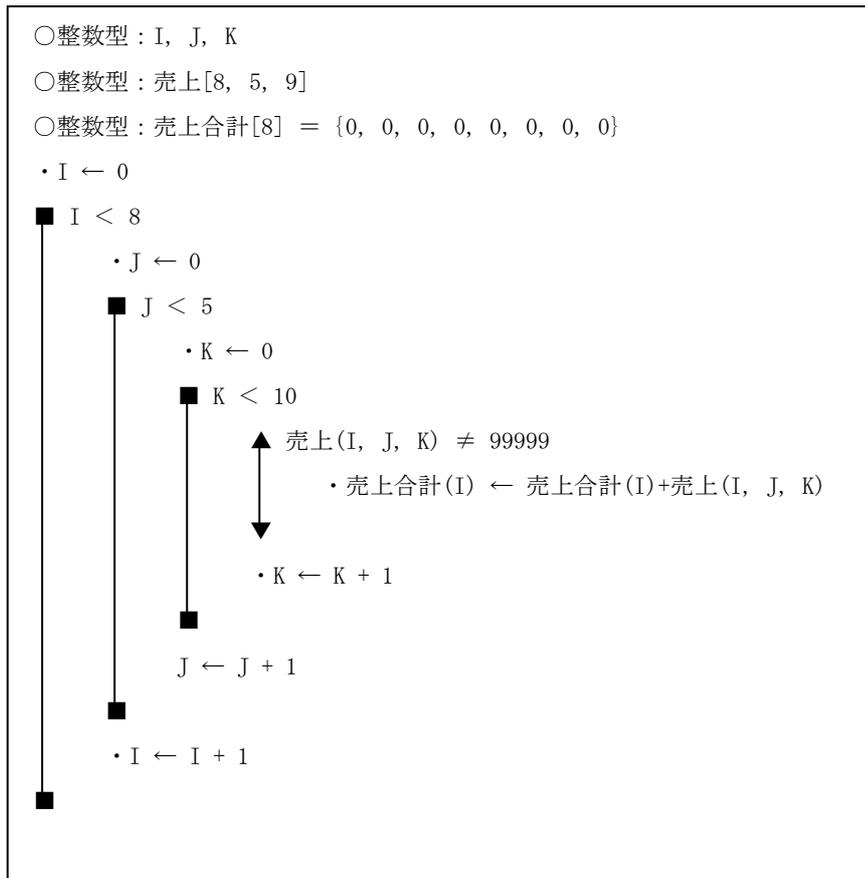
練習 3.30 (解答例)



練習 3.31 (解答例)



練習 3.32 (解答例)



第4章

練習 4.1 (解答例)

○整数型：支店コード, 売上金額

○整数型：対応表[5, 4] = {02, 05, 03, 08,
14, 16, 17, 11,
20, 23, 29, 25,
33, 35, 31, 38,
44, 41, 42, 48 }

○整数型：合計表[5] = {0, 0, 0, 0, 0}

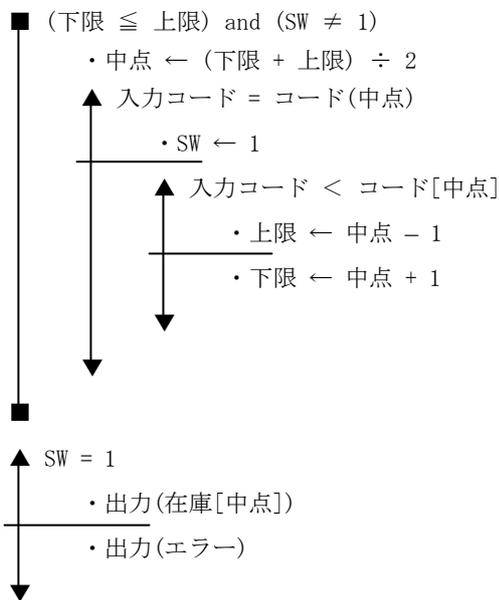
○手続：入力(売上データ) /* 売上データとして支店コード, 売上金額を入力する */

- ・入力(売上データ)
- 支店コード ≠ 99
 - ・SW ← 0
 - ・I ← 0
 - (I < 5) and (SW ≠ 1)
 - ・J ← 0
 - (I < 4) or (SW ≠ 1)
 - ↑ 支店コード = 対応表[I, J]
 - ・合計表[I] ← 合計表[I] + 売上金額
 - ・SW ← 1
 - ↓
 - ・J ← J + 1
 -
 - ・I ← I + 1
 -
 - ・入力(売上データ)

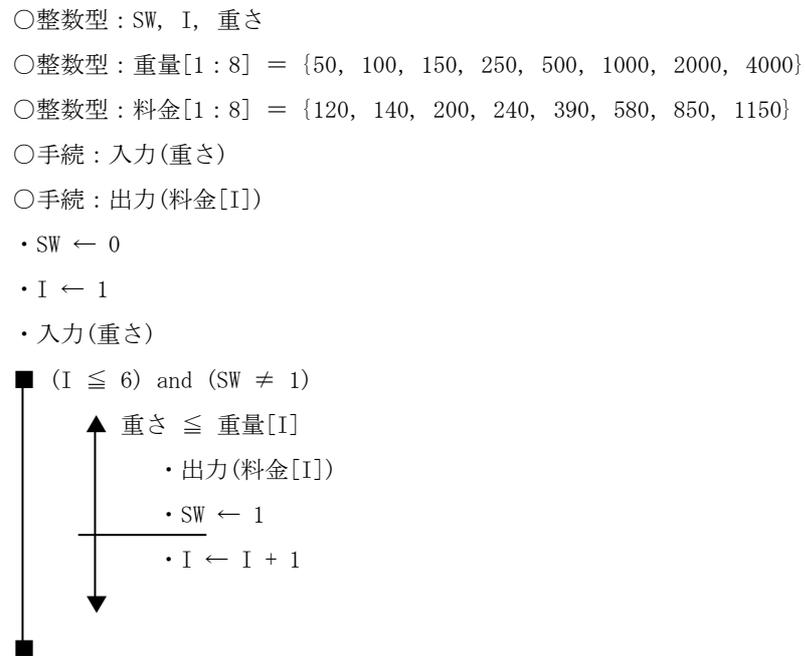
練習 4.2 (解答例)

- 整数型 : コード[1 : 100] = {15, 20, , 195, 200, 203}
- 整数型 : 在庫[1 : 100] = {100, 140, , 103, 50, 90}
- 整数型 : 上限, 下限, 中点, SW
- 手続 : 入力(入力コード)
- 手続 : 出力(在庫[中点])
- 手続 : 出力(エラー)

- SW ← 0
- 下限 ← 1
- 上限 ← 100
- 入力(入力コード)



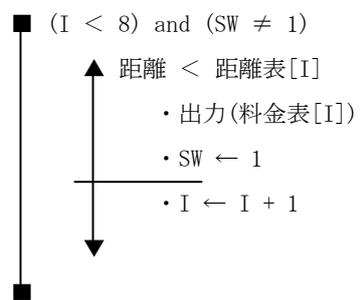
練習 4.3 (解答例)



練習 4.4 (解答例)

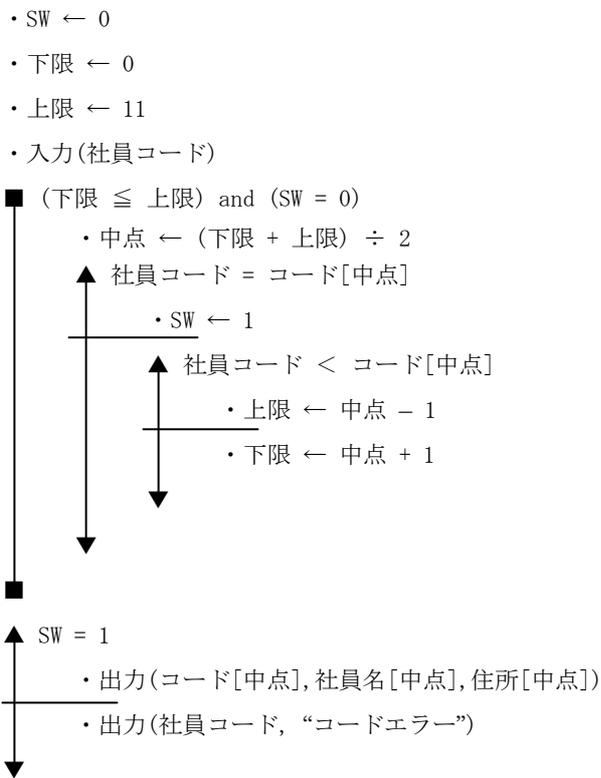
- 整数型 : 距離表[8] = {3, 10, 15, 20, 25, 30, 35, 40}
- 整数型 : 料金表[8] = {110, 130, 150, 170, 190, 210, 230, 250}
- 整数型 : SW, I
- 手続 : 入力(距離)
- 手続 : 出力(料金[I])

- SW ← 0
- I ← 0
- 入力(距離)



練習 4.5 (解答例)

- 整数型：コード[10] = {101, 203, 204, 312, , 908}
- 文字列：社員名[10] = {xxxx, xxxx, xxxx, xxxx}
- 文字列：住所[10] = {xxxx, xxxx, xxxx, xxxx}
- 整数型：社員コード
- 整数型：上限, 下限, 中点, SW
- 手続：入力(社員コード)
- 手続：出力(コード[中点], 社員名[中点], 住所[中点]) /* 検索結果を出力 */
- 手続：出力(社員コード, “コードエラー”) /* エラーを出力 */



練習 4.6 (解答例)

- 文字列：名前[1 : 40] = {xxxx, xxxx, xxxx, xxxx}
- 文字列：得点[1 : 40] = {xxxx, xxxx, xxxx, xxxx}
- 手続：入力(出席番号)
- 手続：出力(出席番号, 名前[出席番号], 得点[出席番号])
- 手続：出力(“エラー”)

・入力(出席番号)

▲ (出席番号 ≥ 1) and (出席番号 ≤ 40)

・出力(出席番号, 名前[出席番号], 得点[出席番号])

・出力(“エラー”)

練習 4.7

エ

練習 4.8

N = 2

SW = 0

練習 4.9

(ア) $J \leq 7$

(イ) $K \leftarrow J + 1$

(ウ) $K \leq 8$

(エ) W商品名 \leftarrow 商品名(J)

(オ) 商品名(J) \leftarrow 商品名(K)

(カ) 商品名(K) \leftarrow W商品名

練習 4.10 (解答例)

○整数型 : I, J, W 年, W 月日

○構造体型 : 生年月日表[50] {整数型 : 年, 整数型 : 月日, 文字列 : 氏名}

○文字列型 : W 氏名

○主プログラム : 整列プログラム

・ I ← 0

■ I < 49

・ J ← I + 1

■ J < 50

↑ 生年月日表[I]. 月日 > 生年月日表[J]. 月日

・ 交換()

↓
・ J ← J + 1

■
・ I ← I + 1

○副プログラム : 交換()

・ W 年 ← 生年月日表[I]. 年

・ 生年月日表[I]. 年 ← 生年月日表[J]. 年

・ 生年月日表[J]. 年 ← W 年

・ W 月日 ← 生年月日表[I]. 月日

・ 生年月日表[I]. 月日 ← 生年月日表[J]. 月日

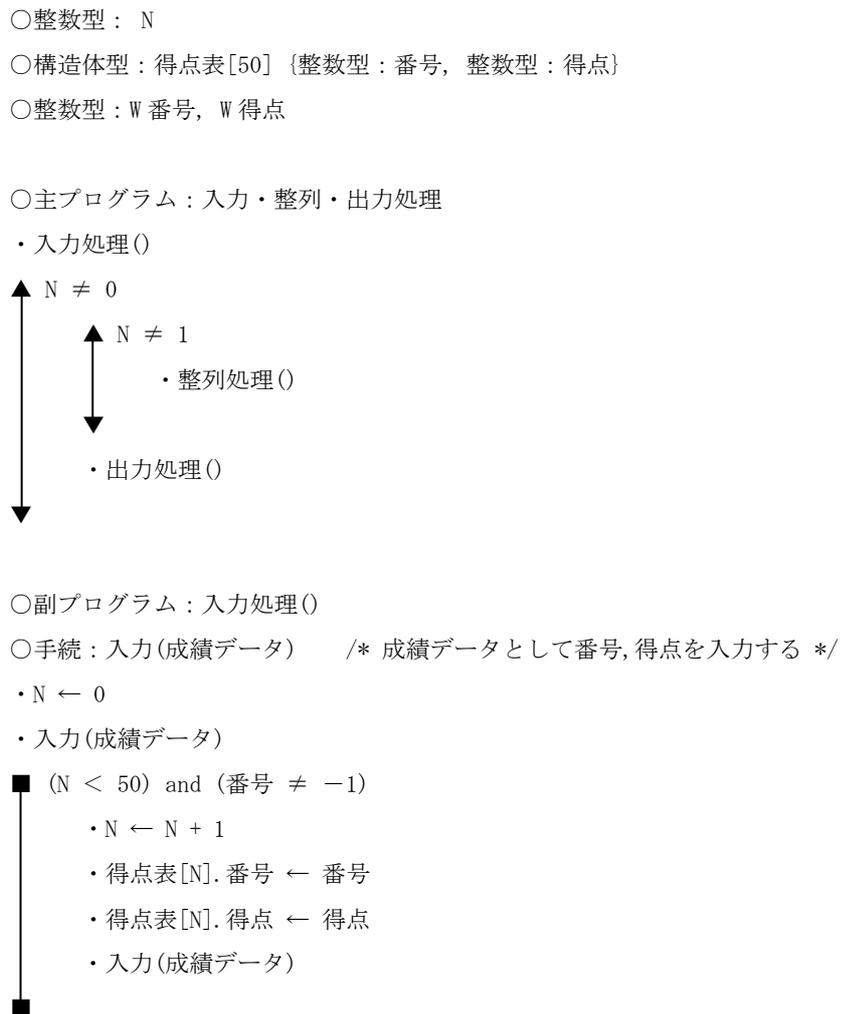
・ 生年月日表[J]. 月日 ← W 月日

・ W 氏名 ← 生年月日表[I]. 氏名

・ 生年月日表[I]. 氏名 ← 生年月日表[J]. 氏名

・ 生年月日表[J]. 氏名 ← W 氏名

練習 4.11 (解答例)



次ページに続く

○副プログラム：整列処理()

○整数型：n, SW, I

• n ← N

• SW ← 1

■ (n > 1) and (SW = 1)

• SW ← 0

• I ← 1

■ I < n

↑ 得点表[I].得点 ≠ 得点表[I+1].得点

↑ 得点表[I].得点 < 得点表[I+1].得点

• 交換()

• SW ← 1

↑ 得点表[I].番号 > 得点表[I+1].番号

• 交換()

• SW ← 1

↓
• I ← I + 1

■

• n ← n - 1

■

○副プログラム：交換()

• W番号 ← 得点表[I].番号

• 得点表[I].番号 ← 得点表[I+1].番号

• 得点表[I+1].番号 ← W番号

• W得点 ← 得点表[I].得点

• 得点表[I].得点 ← 得点表[I+1].得点

• 得点表[I+1].得点 ← W得点

次ページに続く

○副プログラム：出力処理()

○整数型：I, 順位

○手続：出力(順位, 得点表[I].番号, 得点表[I].得点)

/* 順位, 得点表[I].番号, 得点表[I].得点を出力する */

・ I ← 1

・ 順位 ← 1

・ 出力(順位, 得点表[I].番号, 得点表[I].得点)

■ I < N

・ I ← I + 1

↑ 得点表[I].得点 ≠ 得点表[I+1].得点

・ 順位 ← I



・ 出力(順位, 得点表[I].番号, 得点表[I].得点)



練習 4.12

(ア) 最小 ← I

(イ) 最小 ← J

(ウ) 最小 > I

練習 4.15

ウ

練習 4.16

ウ

練習 4.17

ア

練習 4.18

設問1 エ 設問2 a オ b ウ

第5章

練習 5.1

a ○ b × c × d × e ○ f ○ g ○ h × i ×

練習 5.2

ア

練習 5.3

エ

練習 5.4

イ

練習 5.5

ウ

練習 5.6

a-ウ b-キ c-キ d-ウ e-イ f-オ g-ク
h-イ i-ク j-オ

練習 5.7

a-ウ または エ b-ケ c-ア

練習 5.8

ウ

練習 5.9

ウ

練習 5.10

a-ク b-オ c-ウ

練習 5.11

設問1 ウ ($a_1 \rightarrow a_2 \rightarrow a_3 = 70$ 分)

設問2 a—イ b—エ

練習 5.12

ア

第6章

練習 6.1 (解答例)

- 成績一覧表作成
 - ファイル：成績ファイル
 - 整数型：入力状態
 - 整数型：人数，合計点，平均点
 - 構造体型：成績レコード {文字列型：氏名，整数型：点数}
 - 構造体型：成績一覧表 {文字列型：氏名，整数型：点数}
 - 手続：レコード入力(file, record, status) /* file で指定したファイルから
1レコードを読み込み、record で指定した領域に格納する。
status で指定した変数には、レコードが入力されたときは0、
レコードがないときは1が格納される。 */
 - 手続：プリント出力(record) /* record で指定した項目をプリント出力する */
-
- ・人数 ← 0
 - ・合計点 ← 0
 - ・成績ファイルオープン
 - ・プリント出力(“ 氏名 点数 ”)
 - ・レコード入力(成績ファイル，成績レコード，入力状態)
- 入力状態 = 0
- ・成績一覧表.氏名 ← 成績レコード.氏名
 - ・成績一覧表.点数 ← 成績レコード.点数
 - ・プリント出力(成績一覧表)
 - ・人数 ← 人数 + 1
 - ・合計点 ← 合計点 + 成績レコード.点数
 - ・レコード入力(成績ファイル，成績レコード，入力状態)
-
- ・平均点 ← 合計点 ÷ 人数
 - ・プリント出力(“ 平均点 ”，平均点)
 - ・成績ファイルクローズ

練習 6.2 (解答例)

- 成績一覧表作成
- 大域：ファイル：成績ファイル
- 大域：整数型：入力状態
- 大域：整数型：合計点，印字件数
- 大域：構造体型：成績レコード{文字列型：氏名，整数型：試験点，整数型：実習点}
- 大域：構造体型：成績一覧表
 {文字列型：氏名，整数型：試験点，整数型：実習点，整数型：合計点}
- 手続：レコード入力(file, record, status) /* file で指定したファイルから
 1レコードを読み込み、record で指定した領域に格納する。
 status で指定した変数には、レコードが入力されたときは0、
 レコードがないときは1が格納される。 */
- 手続：プリント出力(record) /* record で指定した項目をプリント出力する */

- ・成績ファイルオープン
- ・レコード入力(成績ファイル，成績レコード，入力状態)
- 入力状態 = 0
 - ・成績出力ページ処理()
-
- ・成績ファイルクローズ

- 副プログラム：成績出力ページ処理()
 - ・見出し印字()
 - ・印字件数 ← 0
- (印字件数 < 25) and (入力状態 = 0)
 - ・成績明細行処理()
-

- 副プログラム：成績明細行処理()
 - ・成績一覧表編集()
 - ・プリント出力(成績一覧表)
 - ・印字件数 ← 印字件数 + 1
 - ・レコード入力(成績ファイル，成績レコード，入力状態)

次ページに続く

○副プログラム：成績一覧表編集()

- ・合計点 ←成績レコード. 試験点 + 成績レコード. 実習点
- ・成績一覧表. 氏名 ← 成績レコード. 氏名
- ・成績一覧表. 試験点 ← 成績レコード. 試験点
- ・成績一覧表. 実習点 ← 成績レコード. 実習点
- ・成績一覧表. 合計点 ← 合計点

○副プログラム：見出し印字()

- ・プリント出力(“成績一覧表”)
- ・プリント出力(“氏名 試験点 実習点 合計点”)

練習 6.3 (解答例)

○エラー検査処理

○大域：ファイル：成績ファイル，正常成績ファイル

○大域：整数型：入力状態

○大域：構造体型：成績レコード{整数型：学籍番号，整数型：点数}

○大域：構造体型：エラーチェック表
 {整数型：学籍番号，整数型：点数，文字列型：エラー表示}

○手続：レコード入力(file, record, status) /* file で指定したファイルから
 1レコードを読み込み、record で指定した領域に格納する。
 status で指定した変数には、レコードが入力されたときは0、
 レコードがないときは1が格納される。 */

○手続：レコード出力(file, record) /* file で指定したファイルへ
 record で指定した領域の内容を書き込む。*/

○手続：プリント出力(record) /* record で指定した項目をプリント出力する */

・ファイルオープン /* 成績ファイル，正常成績ファイル */

・プリント出力(“ 学籍番号 点数”)

・レコード入力(成績ファイル，成績レコード，入力状態)

■ 入力状態 = 0

 |

 | ・明細出力処理()

■

・ファイルクローズ /* 成績ファイル，正常成績ファイル */

○副プログラム：明細出力処理()

▲ (成績レコード.点数 \leq 100) and (成績レコード.点数 \geq 0)

 |

 | ・正常明細行出力()

 | ・正常レコード出力()

 | ・エラー明細行出力()

▼

・レコード入力(成績ファイル，成績レコード，入力状態)

次ページに続く

○副プログラム：正常明細行出力()

- ・エラーチェック表. 学籍番号 ← 成績レコード. 学籍番号
- ・エラーチェック表. 点数 ← 成績レコード. 点数
- ・エラーチェック表. エラー表示 ← ” ”
- ・プリント出力(エラーチェック表)

○副プログラム：正常レコード出力()

- ・レコード出力(正常成績ファイル, 成績レコード)

○副プログラム：エラー明細行出力()

- ・エラーチェック表. 学籍番号 ← 成績レコード. 学籍番号
- ・エラーチェック表. 点数 ← 成績レコード. 点数
- ・エラーチェック表. エラー表示 ← ”** エラー ** ”
- ・プリント出力(エラーチェック表)

練習 6.4 (解答例)

- 地区別町別ガス使用一覧表作成
- 大域：ファイル：検針ファイル
- 大域：整数型：入力状態，行数
- 大域：整数型：使用量，料金，全合計料金，地区合計料金，町合計料金
- 大域：整数型：地区退避，町退避
- 大域：構造体型：検針レコード{
構造体型：需要家コード{整数型：地区，整数型：町名，整数型：家屋番号}，
整数型：区分コード，整数型：前月指針，整数型：当月指針}
- 大域：構造体型：ガス使用一覧表{
構造体型：需要家コード{整数型：地区，整数型：町名，整数型：家屋番号}，
整数型：区分コード，整数型：使用量，整数型：料金}
- 手続：レコード入力(file, record, status) /* file で指定したファイルから
1 レコードを読み込み、record で指定した領域に格納する。
status で指定した変数には、レコードが入力されたときは0、
レコードがないときは1が格納される。 */
- 手続：プリント出力(record) /* record で指定した項目をプリント出力する */
 - ・全料金 ← 0
 - ・全使用量 ← 0
 - ・検針ファイルオープン
 - ・レコード入力(検針ファイル，検針レコード，入力状態)
 - 入力状態 = 0
 - ・全データ処理()
 -
 - ・全合計印字()

次ページに続く

○副プログラム：全データ処理()

- ・地区合計使用量 ← 0
- ・地区合計料金 ← 0
- ・地区退避 ← 検針レコード.需要家コード.地区

■ (入力状態 = 0) and (検針レコード.需要家コード.地区 = 地区退避)

- ・地区データ処理()

- ・地区合計印字()

○副プログラム：地区データ処理()

- ・町合計使用量 ← 0
- ・町合計料金 ← 0
- ・町退避 ← 検針レコード.需要家コード.町

■ (入力状態 = 0) and (検針レコード.需要家コード.地区 = 地区退避)

and (検針レコード.需要家コード.町 = 町退避)

- ・町データ処理()

- ・町合計印字()

○副プログラム：町データ処理()

- ・見出し処理()

■ (入力状態 = 0) and (検針レコード.需要家コード.地区 = 地区退避)

and (検針レコード.需要家コード.町 = 町退避) and (行数 < 30)

- ・ページ印字処理()

○副プログラム：ページ印字処理()

- ・料金計算()
- ・明細印字()
- ・町合計計算()
- ・レコード入力(検針ファイル, 検針レコード, 入力状態)

次ページに続く

○副プログラム：見出し処理()

- ・プリント出力(“需要家コード 区分コード 使用量 料金”)
- ・行数 ← 0

○副プログラム：料金計算()

▲ 区分コード = "A"

- ・基本料 ← 1000
- ・単価 ← 75

▲ 区分コード = "B"

- ・基本料 ← 1000
- ・単価 ← 75

▲ 区分コード = "C"

- ・基本料 ← 800
- ・単価 ← 30

▲ 区分コード = "D"

- ・基本料 ← 700
- ・単価 ← 20
- ・基本料 ← 300
- ・単価 ← 63

- ・使用量 ← 検針レコード. 当月指針 - 検針レコード. 前月指針
- ・料金 ← 基本料 + (使用量 × 単価)

次ページに続く

○副プログラム：明細印字()

- ・ガス使用一覧表. 需要家コード ← 検針レコード. 需要家コード
- ・ガス使用一覧表. 区分コード ← 検針レコード. 区分コード
- ・ガス使用一覧表. 使用量 ← 使用量
- ・ガス使用一覧表. 料金 ← 料金
- ・プリント出力(ガス使用一覧表)
- ・行数 ← 行数 + 1

○副プログラム：町合計計算()

- ・町合計料金 ← 町合計料金 + 料金

○副プログラム：町合計印字()

- ・プリント出力("** 町合計 **", 町合計料金)
- ・地区合計料金 ← 地区合計料金 + 町合計料金

○副プログラム：地区合計印字()

- ・プリント出力("** 地区合計 **", 地区合計料金)
- ・全合計料金 ← 全合計料金 + 地区合計料金

○副プログラム：全合計印字()

- ・プリント出力("** 全合計 **", 全合計料金)

練習 6.5 (解答例)

○ファイルの併合
○大域：ファイル：男子ファイル，女子ファイル，男女併合ファイル
○大域：整数型：入力状態，最小値 = {0}
○大域：構造体型：男子レコード{文字列型：氏名，整数型：トータルピン}
○大域：構造体型：女子レコード{文字列型：氏名，整数型：トータルピン}
○大域：構造体型：男女併合レコード{文字列型：氏名，整数型：トータルピン}
○手続：レコード入力(file, record, status) /* file で指定したファイルから
1 レコードを読み込み、record で指定した領域に格納する。
status で指定した変数には、レコードが入力されたときは0、
レコードがないときは1が格納される。 */
○手続：レコード出力(file, record) /* file で指定したファイルへ
record で指定した領域の内容を書き込む。 */

・ファイルオープン /* 男子ファイル，女子ファイル，男女併合ファイル */
・男子ファイル入力()
・女子ファイル入力()
■ (男子レコード.トータルピン ≠ 最小値) or (女子レコード.トータルピン ≠ 最小値)
 ・男女併合レコード処理()
■

・ファイルクローズ /* 男子ファイル，女子ファイル，男女併合ファイル */

○副プログラム：男女併合レコード処理()
▲ (男子レコード.トータルピン) > (女子レコード.トータルピン)
 ・男女併合レコード.氏名 ← 男子レコード.氏名
 ・男女併合レコード.トータルピン ← 男子レコード.トータルピン
 ・レコード出力(男女併合ファイル，男女併合レコード)
 ・男子ファイル入力()
 ・男女併合レコード.氏名 ← 女子レコード.氏名
 ・男女併合レコード.トータルピン ← 女子レコード.トータルピン
 ・レコード出力(男女併合ファイル，男女併合レコード)
 ・女子ファイル入力()
▼

次ページに続く

○副プログラム：男子ファイル入力()

・レコード入力(男子ファイル, 男子レコード, 入力状態)

▲ 入力状態 = 1

↓
・男子レコード, トータルピン ← 最小値

○副プログラム：女子ファイル入力()

・レコード入力(女子ファイル, 女子レコード, 入力状態)

▲ 入力状態 = 1

↓
・女子レコード, トータルピン ← 最小値

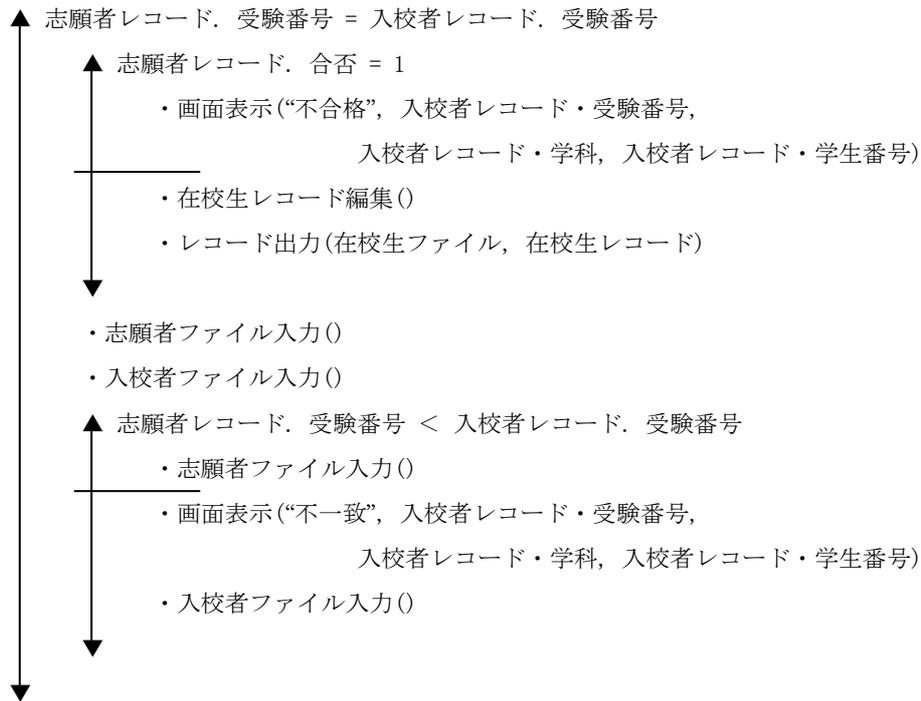
練習 6.6 (解答例)

```
○在校生ファイル作成
○大域：ファイル：志願者ファイル, 入校者ファイル, 在校生ファイル
○大域：整数型：入力状態
○大域：整数型：最大値 = {99999}
○大域：構造体型：志願者レコード{整数型：受験番号, 文字列型：氏名, 整数型：性別,
    整数型：生年月日, 文字列型：現住所, 整数型：出身校コード,
    文字列型：父母住所, 整数型：可否}
○大域：構造体型：入校者レコード{整数型：受験番号, 文字列型：学科,
    整数型：学生番号}
○大域：構造体型：在校生レコード{整数型：受験番号, 整数型：学生番号,
    文字列型：氏名, 整数型：性別, 整数型：生年月日,
    文字列型：現住所, 文字列型：父母住所}
○手続：レコード入力(file, record, status) /* file で指定したファイルから
    1 レコードを読み込み、record で指定した領域に格納する。
    status で指定した変数には、レコードが入力されたときは 0、
    レコードがないときは 1 が格納される。 */
○手続：レコード出力(file, record) /* file で指定したファイルへ
    record で指定した領域の内容を書き込む。 */
○手続：画面表示(data) /* data で指定した領域の内容を画面に表示する。 */

・ファイルオープン /* 志願者ファイル, 入校者ファイル, 在校生ファイル */
・志願者ファイル入力()
・入校者ファイル入力()
■ (志願者レコード. 受験番号≠最大値) or (入校者レコード. 受験番号≠最大値)
    ・突合せ処理()
■
・ファイルクローズ /* 志願者ファイル, 入校者ファイル, 在校生ファイル */
```

次ページに続く

○副プログラム：突合せ処理()

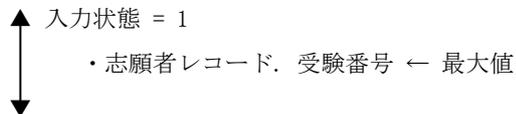


○副プログラム：在校生記録編集()

- ・在校生記録. 受験番号 ← 志願者記録. 受験番号
- ・在校生記録. 学生番号 ← 入校者記録. 学生番号
- ・在校生記録. 氏名 ← 志願者記録. 氏名
- ・在校生記録. 性別 ← 志願者記録. 性別
- ・在校生記録. 生年月日 ← 志願者記録. 生年月日
- ・在校生記録. 現住所 ← 志願者記録. 現住所
- ・在校生記録. 父母住所 ← 志願者記録. 父母住所

○副プログラム：志願者ファイル入力()

- ・記録入力(志願者ファイル, 志願者記録, 入力状態)



次ページに続く

○副プログラム：入校者ファイル入力()

・レコード入力(入校者ファイル，入校者レコード，入力状態)

▲ 入力状態 = 1



・入校者レコード. 受験番号 ← 最大値

練習 6.7 (解答例)

```
○得意先台帳更新処理_1 対 1
○大域：ファイル：入金ファイル，旧得意先台帳，新得意先台帳
○大域：整数型：入力状態
○大域：整数型：更新売掛金残高
○大域：構造体型：入金レコード{整数型：得意先コード， 整数型：伝票日付，
                               整数型：入金額}
○大域：構造体型：旧得意先レコード{整数型：得意先コード， 文字列型：得意先名，
                               整数型：売掛金残高， 整数型：年間売掛合計，
                               構造体型：最終入金情報{整数型：日付， 整数型：金額}}
○大域：構造体型：新得意先レコード{整数型：得意先コード， 文字列型：得意先名，
                               整数型：売掛金残高， 整数型：年間売掛合計，
                               構造体型：最終入金情報{整数型：日付， 整数型：金額}}
○手続：レコード入力(file, record, status) /* file で指定したファイルから
                                             1 レコードを読み込み、record で指定した領域に格納する。
                                             status で指定した変数には、レコードが入力されたときは0、
                                             レコードがないときは1が格納される。 */
○手続：レコード出力(file, record) /* file で指定したファイルへ
                                     record で指定した領域の内容を書き込む。*/
○手続：画面表示(data) /* data で指定した領域の内容を画面に表示する。*/

・ファイルオープン /* 入金ファイル，旧得意先台帳，新得意先台帳 */
・入金ファイル入力()
・旧得意先台帳入力()
■ (旧得意先レコード. 得意先コード≠最大値)
  |
  | or (入金レコード. 得意先コード≠最大値)
  |
  | ・更新情報処理()
  ■
・ファイルクローズ /* 入金ファイル，旧得意先台帳，新得意先台帳 */
```

次ページに続く

○副プログラム：更新情報処理()

▲ 旧得意先レコード. 得意先コード = 入金レコード. 得意先コード

- ・更新売掛金残高 ← 旧得意先レコード. 売掛金残高 - 入金レコード. 入金額
- ・新得意先レコード更新編集()
- ・レコード出力(新得意先台帳, 新得意先レコード)
- ・旧得意先台帳入力()
- ・入金ファイル入力()

▲ 旧得意先レコード. 得意先コード < 入金レコード. 得意先コード

- ・新得意先レコード編集()
- ・レコード出力(新得意先台帳, 新得意先レコード)
- ・旧得意先台帳入力()
- ・画面表示(“該当なしエラー”, 入金レコード. 得意先コード,
入金レコード. 伝票日付,
入金レコード. 入金額)
- ・入金ファイル入力()

○副プログラム：旧得意先台帳入力()

- ・レコード入力(旧得意先台帳, 旧得意先レコード, 入力状態)

▲ 入力状態 = 1

- ・旧得意先レコード. 得意先コード ← 最大値

○副プログラム：入金ファイル入力()

- ・レコード入力(入金ファイル, 入金レコード, 入力状態)

▲ 入力状態 = 1

- ・入金レコード. 得意先コード ← 最大値

次ページに続く

○副プログラム：新得意先レコード編集()

- ・新得意先レコード. 得意先コード ← 旧得意先レコード. 得意先コード
- ・新得意先レコード. 得意先名 ← 旧得意先レコード. 得意先名
- ・新得意先レコード. 売掛金残高 ← 旧得意先レコード. 売掛金残高
- ・新得意先レコード. 年間売掛合計 ← 旧得意先レコード. 年間売掛合計
- ・新得意先レコード. 最終入金情報. 日付 ← 旧得意先レコード. 最終入金情報. 日付
- ・新得意先レコード. 最終入金情報. 金額 ← 旧得意先レコード. 最終入金情報. 金額

○副プログラム：新得意先レコード更新編集()

- ・新得意先レコード. 得意先コード ← 旧得意先レコード. 得意先コード
- ・新得意先レコード. 得意先名 ← 旧得意先レコード. 得意先名
- ・新得意先レコード. 売掛金残高 ← 更新売掛金残高
- ・新得意先レコード. 年間売掛合計 ← 旧得意先レコード. 年間売掛合計
- ・新得意先レコード. 最終入金情報. 日付 ← 旧得意先レコード. 最終入金情報. 日付
- ・新得意先レコード. 最終入金情報. 金額 ← 旧得意先レコード. 最終入金情報. 金額

Memorandum