文部科学省後援

情報システム試験

令和元年度後期 情報検定

＜実施　令和2年2月9日（日）＞

プログラミングスキル

（説明時間　10：00～10：10）

（試験時間　10：10～11：40）

・試験問題は試験開始の合図があるまで開かないでください。

・解答用紙（マークシート）への必要事項の記入は，試験開始の合図と同時に行いますので，それまで伏せておいてください。
・試験開始の合図の後，次のページを開いてください。**＜受験上の注意＞**が記載されています。必ず目を通してから解答を始めてください。
・試験問題は，すべてマークシート方式です。正解と思われるものを１つ選び，解答欄の()をＨＢの黒鉛筆でぬりつぶしてください。２つ以上ぬりつぶすと，不正解になります。
・辞書，参考書類の使用および筆記用具の貸し借りは一切禁止です。
・電卓の使用が認められます。ただし，下記の機種については使用が認められません。

＜使用を認めない電卓＞

１．電池式（太陽電池を含む）以外の電卓
２．文字表示領域が複数行ある電卓（計算状態表示の一行は含まない）
３．プログラムを組み込む機能がある電卓
４．電卓が主たる機能ではないもの
　＊パソコン（電子メール専用機等を含む），携帯電話（ＰＨＳ），スマートフォン，タブレット，電子手帳，電子メモ，電子辞書，翻訳機能付き電卓，音声応答のある電卓，電卓付き腕時計，時計型ウェアラブル端末等
５．その他試験監督者が不適切と認めるもの

一般財団法人　職業教育・キャリア教育財団

＜受験上の注意＞

１．この試験問題は29ページあります。ページ数を確認してください。
乱丁等がある場合は，手をあげて試験監督者に合図してください。
※問題を読みやすくするために空白ページを設けている場合があります。

２．解答用紙（マークシート）に，受験者氏名・受験番号を記入し，受験番号下欄の数字をぬりつぶしてください。正しく記入されていない場合は，採点されませんので十分注意してください。

３．試験問題についての質問には，一切答えられません。自分で判断して解答してください。

４．試験中の筆記用具の貸し借りは一切禁止します。筆記用具が破損等により使用不能となった場合は，手をあげて試験監督者に合図してください。

５．試験を開始してから30分以内は途中退出できません。30分経過後退出する場合は，もう一度，受験番号・マーク・氏名が記載されているか確認して退出してください。なお，試験終了５分前の合図以降は退出できません。試験問題は各自お持ち帰りください。

６．試験後にお知らせする合否結果（合否通知），および合格者に交付する「合格証・認定証」はすべて，Ｗｅｂページ（ＰＣ，モバイル）での認証によるディジタル「合否通知」，ディジタル「合格証・認定証」に移行しました。
①団体宛にはこれまでと同様に合否結果一覧ほか，試験結果資料一式を送付します。
②合否等の結果についての電話・手紙等でのお問い合わせには，一切応じられませんので，ご了承ください。

＜問題の構成＞

必須問題　全員解答

問題１～問題４	2 ページ～15 ページ

選択問題　次の問題から１問選択し解答せよ。

（選択した問題は解答用紙「選択欄」に必ずマークすること）

※選択欄にマークがなく，解答のみマークした場合は採点を行いません。

・Ｃ言語の問題	17 ページ～20 ページ
・表計算の問題	21 ページ～26 ページ
・アセンブラの問題	27 ページ～29 ページ

必須問題

問題１　次のスタックとキューの説明を読み，各設問に答えよ。

［スタックについて］

スタックとは後入先出法(LIFO)によりデータを管理するメモリ領域のことである。プログラム中で副プログラムや関数を呼び出すときのアドレス情報などを一時的に蓄える場所として使われる。ここでは，スタックにデータを格納する場合は push，スタックからデータを取り出す場合には pop を使う。

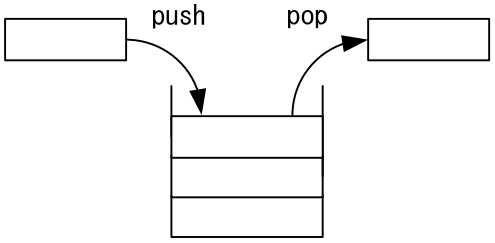


図１　スタック

・push　書式：push(データ)
　　例：　push(100)　※スタックに 100 を格納
・pop　書式：pop()
　　例：　x ← pop()　※スタックからデータを取り出して変数 x へ代入

［キューについて］

キューとは先入先出法(FIFO)によりデータを管理するメモリ領域のことである。OS の待ち行列管理などに用いられる。ここでは，キューにデータを格納する場合は enqueue，キューからデータを取り出す場合は dequeue を使う。

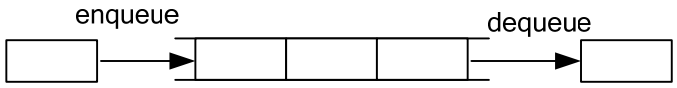


図２　キュー

・enqueue　書式：enqueue(データ)
　　例：　enqueue(100)　※キューに 100 を格納
・dequeue　書式：dequeue()
　　例：　x ← dequeue()　※キューから取り出したデータを変数 x へ代入

＜設問１＞　次のスタックとキューを操作する処理を実行後，変数 x に格納される値を解答群から選べ。ここではスタックとキューは異なるメモリ領域を使っており，スタックおよびキューの領域は空の状態から各操作を始めるものとする。

(1)
```
push(100)
push(200)
push(300)
push(400)
x ← pop()
x ← pop()
```

(2)
```
enqueue(100)
enqueue(200)
enqueue(300)
enqueue(400)
x ← dequeue()
x ← dequeue()
```

(3)
```
enqueue(100)
enqueue(200)
push(300)
push(400)
push(dequeue())
x ← pop()
```

(1) ～ (3) の解答群

ア．100　　イ．200　　ウ．300　　エ．400

＜設問２＞　次の処理の説明を読み，処理の［　　］に入れるべき適切な字句を解答群から選べ。なお，処理の「print」は，後に続く値を標準出力装置へ出力するための命令である。

［処理の説明］

・スタックおよびキューは空の状態から始める
・キューへ順番に 30，20，10 を格納する
・スタックを利用してキューの値を入れ替え，10，20，30 の順に出力する

［処理］

```
enqueue(30)
enqueue(20)
enqueue(10)
[ (4) ]
push(dequeue())
[ (5) ]
enqueue(pop())
```

※ スタックを利用してデータを入れ替える

```
print dequeue()  ※ 10 が表示される
print dequeue()  ※ 20 が表示される
print dequeue()  ※ 30 が表示される
```

(4)，(5) の解答群

ア．enqueue(dequeue())　　イ．enqueue(pop())
ウ．push(dequeue())　　エ．push(pop())

問題２　次の図形の回転に関する記述を読み，流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

［図形の回転の説明］

10行10列の２次元配列zに対して，時計回りに90度回転させる。図形は白黒であり，白は0，黒は1として，２次元配列zに格納されており，添字は0から始まる。

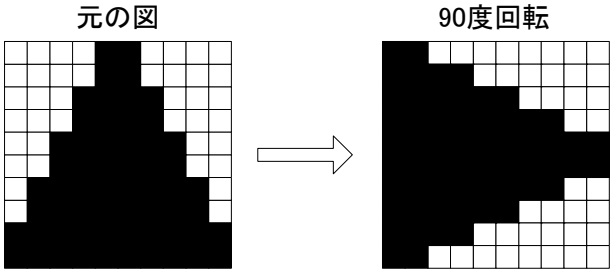


図１　回転の例

配列ｚ内での移動は，①～④の順序で，外側から中央に向けて行う。

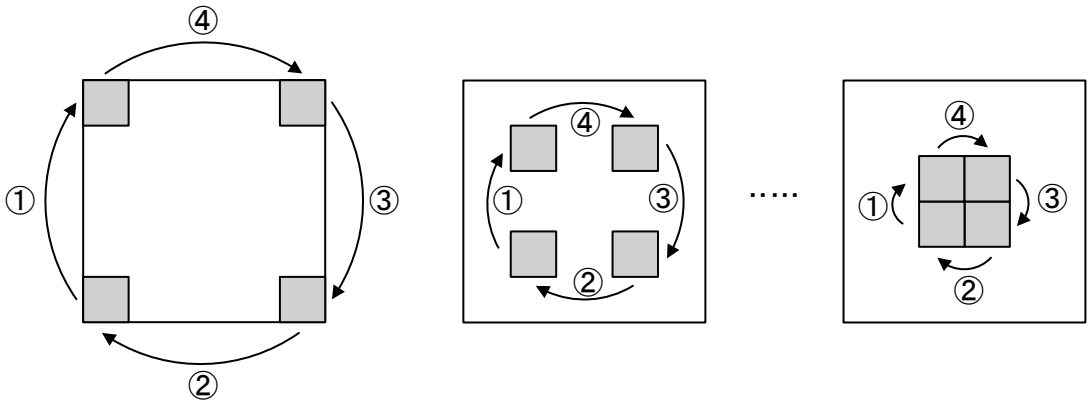


図２　回転の手順

退避領域をwとして，一番外の周についての配列要素の移動は，次のようになる。

一番外の周 1個目

w ← z[0][0]
z[0][0] ← z[9][0]
z[9][0] ← z[9][9]
z[9][9] ← z[0][9]
z[0][9] ← w

一番外の周 2個目

w ← z[0][1]
z[0][1] ← z[8][0]
z[8][0] ← z[9][8]
z[9][8] ← z[1][9]
z[1][9] ← w

…

一番外の周 9個目

w ← z[0][8]
z[0][8] ← z[1][0]
z[1][0] ← z[9][1]
z[9][1] ← z[8][9]
z[8][9] ← w

次に，一つ内側の周についての配列要素の移動は，次のようになる。

一つ内側の周 1個目

w ← z[1][1]
z[1][1] ← z[8][1]
z[8][1] ← z[8][8]
z[8][8] ← z[1][8]
z[1][8] ← w

一つ内側の周 2個目

w ← z[1][2]
z[1][2] ← z[7][1]
z[7][1] ← z[8][7]
z[8][7] ← z[2][8]
z[2][8] ← w

…

一つ内側の周 7個目

w ← z[1][7]
z[1][7] ← z[2][1]
z[2][1] ← z[8][2]
z[8][2] ← z[7][8]
z[7][8] ← w

このように，配列の中心に向かって繰り返し，最も内側の4個の要素についての移動は，次のようになる。

最も内側の要素

w ← z[4][4]
z[4][4] ← z[5][4]
z[5][4] ← z[5][5]
z[5][5] ← z[4][5]
z[4][5] ← w

［流れ図］

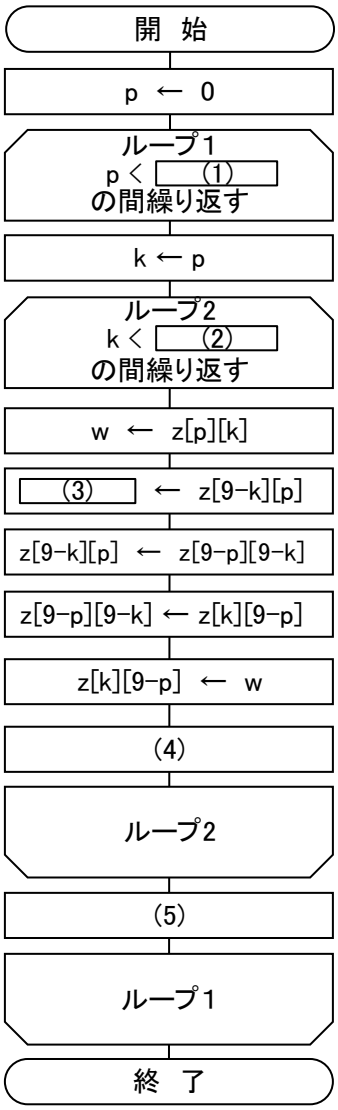


図３　回転の流れ図

(1)，(2) の解答群

ア．1　　イ．5　　ウ．5 － p　　エ．9 － p

(3) の解答群

ア．z[k][p]　　イ．z[k][9－p]

ウ．z[p][k]　　エ．z[p][9－k]

(4) の解答群

ア．k ← k － 1　　イ．k ← k ＋ 1　　ウ．k ← k ＋ p

(5) の解答群

ア．p ← p － 1　　イ．p ← p ＋ 1　　ウ．p ← p ＋ k

問題３　次のヒープに関する記述を読み，各設問に答えよ。

［ヒープについて］

二分木構造において，親の値が子の値より常に大きいか等しい，または小さいか等しいという制約を満たすものをヒープと呼ぶ。この問題では，親の値が子の値より常に大きいか等しいものを扱う。

図１にヒープの例を示す。図は「○」で節を表しており，「○」の中の数値が節の値である。また，子を持つ節を親と呼び，親を持たない節を根と呼ぶ。ヒープの制約に従えば，根の値は節の中で一番大きな値となる。なお，子の節が１つの場合は左の子とし，子の節が２つある場合の並びは値と関係ない。

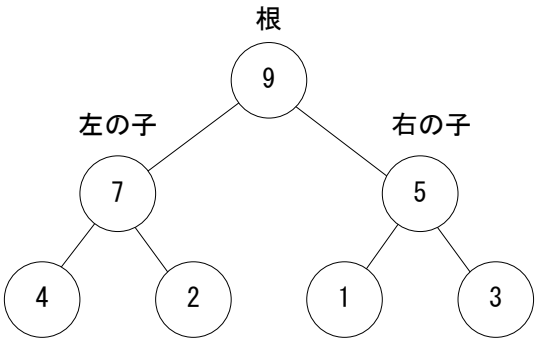


図１　ヒープの例

［ヒープの実装について］

・ヒープ構造を一次元配列 h に構築する。
・配列の添え字は 0 から始まるものとする。
・根は h[0]に格納する。
・親の節の位置を i とすれば，左の子が格納される位置は i×2＋1，右の子が格納される位置は i×2＋2 となる。

次の図２は，図１を配列 h に格納したものである。

位置	0	1	2	3	4	5	6
h	9	7	5	4	2	1	3

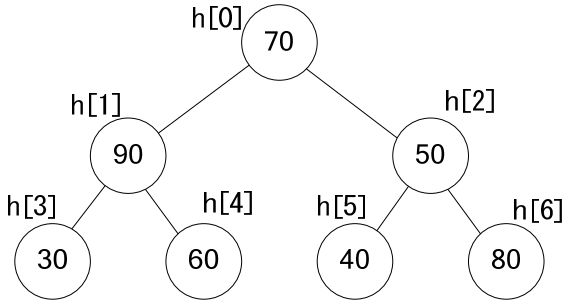
図２　ヒープを一次元配列に格納した例

・根である h[0]の左の子は h[1]，右の子は h[2]に格納される。
・h[1]の左の子は h[3]，右の子は h[4]に格納される。
・h[2]の左の子は h[5]，右の子は h[6]に格納される。

＜設問１＞　次のヒープの構築に関する記述中［　　］に入れるべき適切な字句を解答群から選べ。

ランダムな値が格納された配列でヒープを構築するには，「親と子要素の中での最大値が親の位置に格納されるようにデータを入れ替える」という構築操作を繰り返す。この時，データの入れ替えが発生した場合は，入れ替えた子の位置を新たな親の位置として構築操作を続け，入れ替えが発生しなければ構築操作を終える。

ここで，ランダムな値を格納した二分木が図３のような場合において，ヒープを構築することを考える。



位置	0	1	2	3	4	5	6
h	70	90	50	30	60	40	80

図３　ヒープが構築されていない二分木と配列の状態

ヒープの構築を開始する位置は，末端の親(位置が一番大きな場所にある親)から始め，配列の前方(位置の小さい方)の親へと進める。図３では，末端の親はh[2]である。これは，「(配列の要素数－2)÷2」で計算できる。なお，除算の結果は小数点以下を切り捨てた整数値にする。

最初に，h[2]と h[2]を親とした子要素の中での最大値を h[2]に格納する。この場合は，h[2]と［　(1)　］を入れ替える。さらに［　(1)　］を新たな親の位置として構築操作を進めようとすれば，新しい子の位置は配列の要素を超えた位置になるため，構築操作を終える。

次に，h[1]と h[1]を親とした子要素の中での最大値を h[1]に格納する。ここで，h[1]に格納されている値は子要素の値と比較しても一番大きな値である。よって，入れ替えは行わないため，構築操作を終える。

最後に，h[0]を親とした子要素の中での最大値をh[0]に格納する。この場合は，h[0]と［　(2)　］を入れ替える。さらに［　(2)　］を新しい親の位置として構築操作を続ける。新しい親の位置に格納されている値が子要素の値と比較して一番大きな値となる。よって，入れ替えは行わないため，構築操作を終了する。

以上の操作により構築したヒープは図４になる。

位置	0	1	2	3	4	5	6
h	90	70	80	30	60	40	50

図４　ヒープを構築した状態

(1)，(2) の解答群

ア．h[1]　　イ．h[2]　　ウ．h[3]
エ．h[4]　　オ．h[5]　　カ．h[6]

＜設問２＞　配列に構築されたヒープとして正しいものを(3)の解答群から選べ。

(3) の解答群

ア．	700	200	300	400	500	600	100
イ．	600	500	700	400	300	200	100
ウ．	700	500	600	300	400	100	200
エ．	700	500	400	600	100	200	300

＜設問３＞　次の流れ図の説明を読み，流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

［流れ図の説明］

要素数 n の一次元配列 h[i](i＝0,1,…,n－1)に格納されたランダムな値に対してヒープを構築する。流れ図に示したものは，ヒープの構築を制御する makeHeap，ヒープを構築する処理を行う downHeap，配列内の値を入れ替える swap である。

・makeHeap は，ヒープを構築するための操作を行う donwHeap を呼び出す。この時，親の要素位置と配列内の最後の位置を引数で与える。
・downHeap は，配列内の値を入れ替える swap を呼び出す。この時，入れ替える配列内の２つの位置を引数で与える。

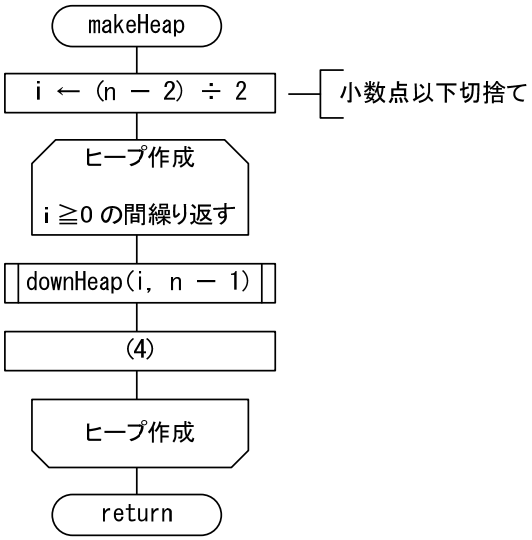


図５　makeHeap の流れ図

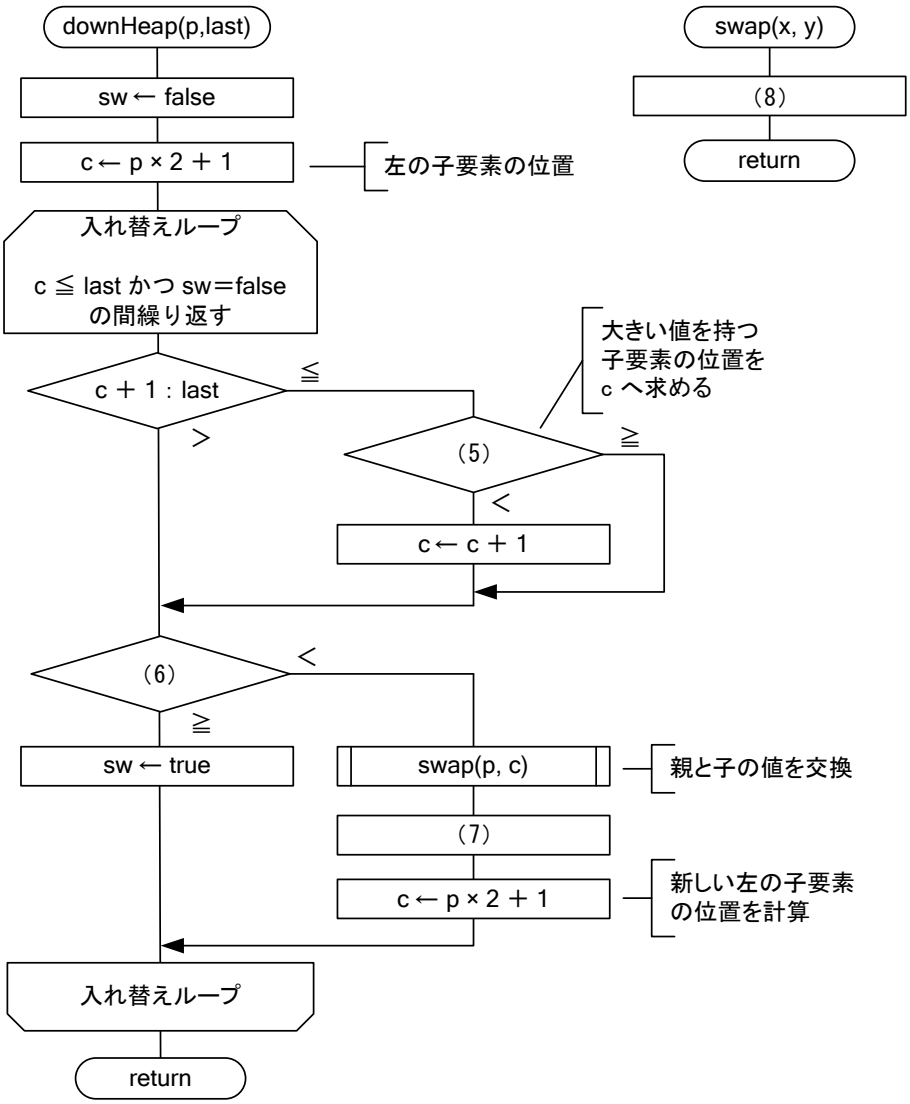


図６　downHeap と swap の流れ図

(4) の解答群

ア．i ← i － 1　　イ．i ← i ＋ 1
ウ．i ← i × 2　　エ．i ← i ＋ 2

(5)，(6) の解答群

ア．h[c]：h[c＋1]　　イ．h[c＋1]：h[c]
ウ．h[c]：h[p]　　エ．h[c＋1]：h[p]
オ．h[p]：h[c]　　カ．h[p＋1]：h[c]

(7) の解答群

ア．p ← c　　イ．p ← c ＋ 1
ウ．p ← p ＋ 1　　エ．p ← p × 2

(8) の解答群

ア．
```
h[x] ← h[y]
h[y] ← h[x]
```

イ．
```
h[x] ← h[x＋1]
h[y] ← h[y＋1]
```

ウ．
```
work ← h[x]
h[y] ← work
h[x] ← h[y]
```

エ．
```
work ← h[x]
h[x] ← h[y]
h[y] ← work
```

＜設問４＞　次のヒープソートに関する記述を読み，流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

ヒープが構築された配列では，配列の先頭に格納されるのは一番大きな値である。これを利用して配列内を並び替える方法がヒープソートである。

例えば，図７のヒープが構築されている配列で考える。

	0	1	2	3	4	5	6
h	90	70	80	30	60	40	50

図７　ヒープが構築されている配列

h[0]に格納されている値が最大値であり，配列の最後である h[6]と入れ替える。

	0	1	2	3	4	5	6
h	50	70	80	30	60	40	90

図８　h[0]と h[6]を入れ替える

要素を入れ替えたことによりヒープ構造が崩れるため，未整列の領域である h[0]～h[5]でヒープを再構築する。再構築では，h[0]からの構築操作を行う。

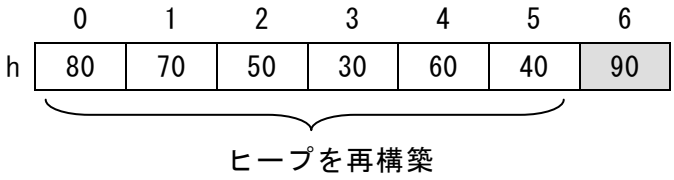


図９　h[0]～h[5]でヒープを再構築

再構築を終えると，h[0]～h[5]の最大値が h[0]に格納されるため，h[0]と h[5]を入れ替える。

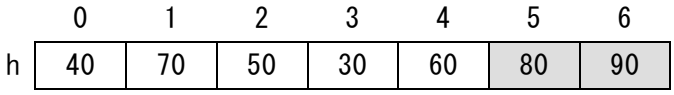


図 10　h[0]と h[5]でヒープを入れ替える

再びヒープ構造が崩れるため，未整列の領域である h[0]～h[4]でヒープを再構築する。再構築では，h[0]からの構築操作を行う。

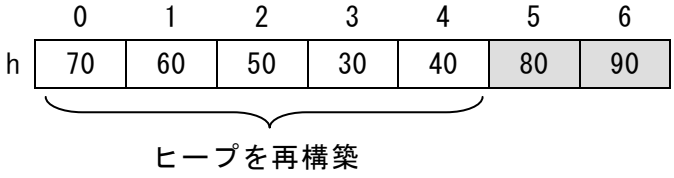


図 11　h[0]～h[4]でヒープを再構築

このように，配列の先頭と並べ替える対象範囲の末尾を入れ替え，対象範囲を１つ縮めてヒープを再構築するという操作を繰り返すことで，配列内のデータを昇順に並べ替えることができる。

次の流れ図は，ランダムに格納された配列 h[i](i＝0，1，…，n－1)をヒープソートにより並べ替えるものである。流れ図中で呼び出す makeHeap，downHeap，swap は，設問３の流れ図で用いたものである。

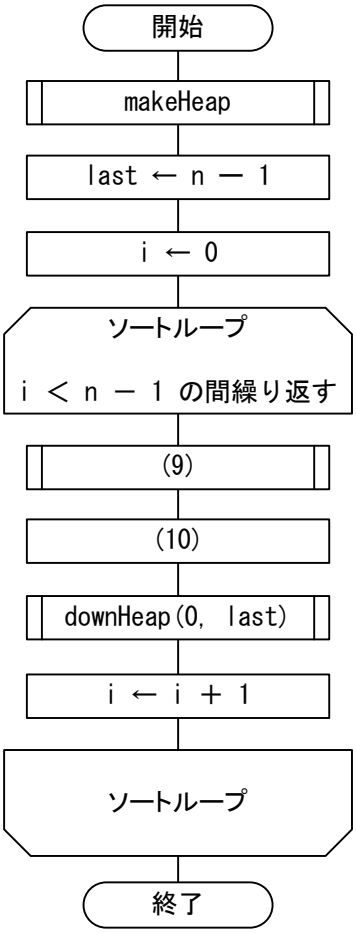


図 12　ヒープソートの流れ図

(9) の解答群

ア. swap(0, last)　　イ. swap(0, i)

ウ. swap(i, n)　　エ. swap(i, last)

(10) の解答群

ア. last ← last − 1　　イ. last ← last ＋ 1

ウ. last ← last − i　　エ. last ← last ＋ i

問題４　次のプログラムの説明を読み，各設問に答えよ。

［プログラムの説明］

階乗を再帰的に求めるプログラムFactである。例えば，ｎの階乗はｎ！と表し，

ｎ！＝ｎ×(ｎ－1)×(ｎ－2)×・・・×2×1　　（ｎ＝0 の場合は 1)

と求めることができる。また，

(ｎ－1)！＝(ｎ－1)×(ｎ－2)×・・・×2×1

を用いて，

ｎ！＝ｎ×(ｎ－1)！　　（ただし，0！＝ 1)

と再帰的に求めることもできる。

［擬似言語の記述形式の説明］

記述形式	説明
○	手続き，変数などの名前，型などを宣言する
・変数 ← 式	変数に式の値を代入する
/* 文 */	注釈を記述する
▲ 条件式 ・処理１ ―― ・処理２ ▼	選択処理を示す。 条件式が真の時は処理１を実行し， 偽の時は処理２を実行する。
■ 条件式 ・処理 ■	前判定繰り返し処理を示す。 条件式が真の間，処理を実行する。

［演算子と優先順位］

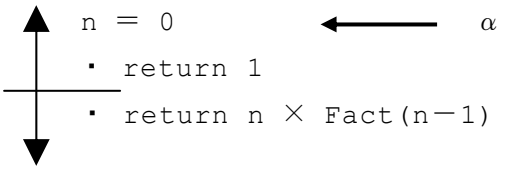
演算の種類	演算子	優先順位
単項演算	＋，－，not	高 ▲
乗除演算	×，÷，％	
加減演算	＋，－	
関係演算	＞，＜，≧，≦，＝，≠	
論理積	and	
論理和	or	▼ 低

注記　整数同士の除算では，整数の商を結果として返す。％演算子は剰余算を表す。

［プログラム］

○Fact（整数型：ｎ）

/* 階乗の計算をする */



＜設問１＞　4 の階乗を求める場合，プログラム中のαを実行するときの変数ｎをトレースした表の［　　］に入れるべき適切な字句を解答群から選べ。

表　トレースの内容

順番	１回目	２回目	３回目	４回目	５回目
ｎ	(1)	3	(2)	1	(3)

(1) ～ (3) の解答群

ア．０　　イ．１　　ウ．２　　エ．３　　オ．４

＜設問２＞　プログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

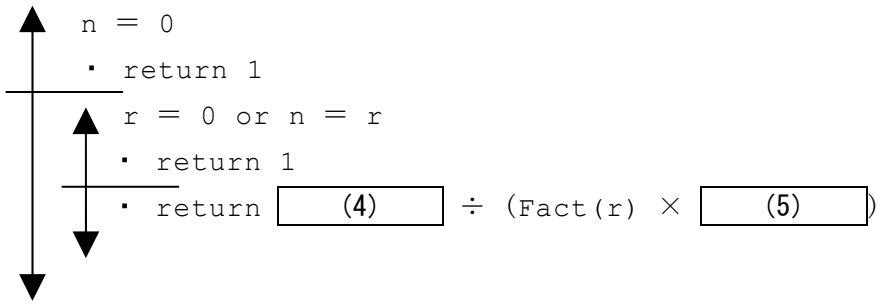
次は，階乗を求めるプログラムを利用して，組合せ(Combination)の総数を求めるプログラムComb_nである。組合せとは，異なるｎ個のものの中から，異なるｒ個のものを取り出し，順序を考えず１組にしたものであり，ｎ個からｒ個取る組合せといい，その総数を${}_{n}C_{r}$で表し，次の式で求めることができる。

$${}_{n}C_{r} = n! \div \{r! \times (n-r)!\}$$　（ｒ＝ 0 またはｒ＝ｎ の場合は 1）

［プログラム］

○Comb_n（整数型：ｎ，整数型：ｒ）

/* 組合せの計算をする */



(4)，(5) の解答群

ア．Fact(ｎ)　　イ．Fact(ｎ－ｒ)　　ウ．Fact(ｒ)

エ．(ｎ－1) × Fact(ｎ)　　オ．ｎ × Fact(ｎ－ｒ)　　カ．ｎ × Fact(ｎ)

＜ 選 択 問 題 ＞

選択問題は問題から１つ選択し解答せよ。
選択した問題は必ず，解答用紙「選択欄」にマークすること。
※選択欄にマークがなく，解答のみの場合は採点を行いません。

各構成は以下のとおり。

選択問題

選択問題　C言語の問題

次の線形リストの説明を読み，各設問に答えよ。

［線形リストの説明］

線形リストとは，データと次のデータが格納されている位置(アドレス)を示すポインタによる要素で構成するデータ構造である。ここでは，線形リストの先頭の位置は root に格納し，最後の要素のポインタには NULL を格納する。

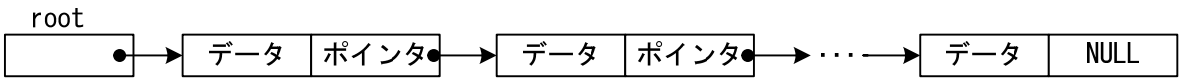


図１　線形リスト

＜設問１＞　次の線形リストへの追加・削除に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

何も格納されていない状態の線形リストに新しい要素を追加することを考える。

追加する要素は線形リストにおける最初の要素となるため，root は要素が格納された位置を保持し，追加する要素のポインタを NULL とする。

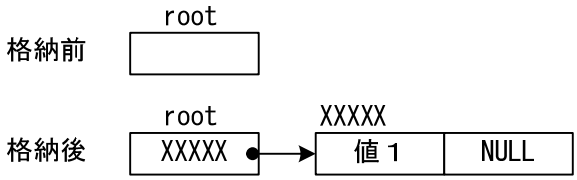


図２　最初の要素を格納

さらに新しい要素を追加することを考える。

格納前の線形リストにおける末尾要素のポインタには，新しく追加する要素が格納された位置を持ち，追加する要素のポインタは NULL とする。

図３　新しい要素を追加

このように線形リストの末尾に要素を追加することを繰り返した図４の状態から，要素を削除することを次に考える。

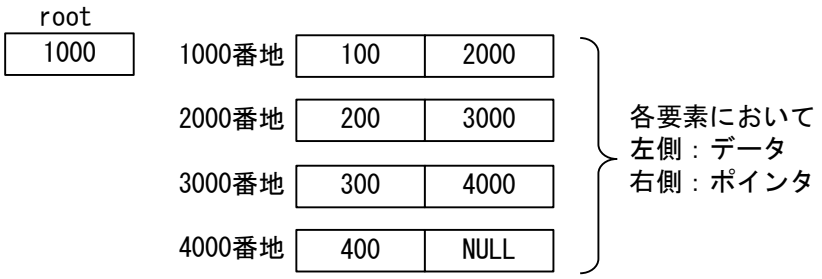


図４　要素を格納した線形リスト

要素の削除は，root から線形リストをたどった場合に参照されないよう，ポインタの値を変更することで実現する。

図４の状態からデータ 200 を削除する場合は，［　(1)　］番地のポインタを［　(2)　］に変更する。

また，線形リストの先頭要素を削除する場合は，root の内容を変更する。図４の状態から先頭要素を削除するには，root の値を［　(3)　］に変更する。

(1) ～ (3) の解答群

ア．1000　　イ．2000　　ウ．3000　　エ．4000　　オ．root

＜設問２＞　次のプログラムの説明を読み，プログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

［プログラムの説明］

線形リストに要素を追加する関数 appendList，要素を削除する関数 deleteList である。なお，線形リストが空の場合は，線形リストの先頭を示す変数 root に NULL が格納される。

［関数の説明］

appendList 関数

引　数：data(整数型)，*root(構造体のポインタ　線形リストの先頭位置)

機　能：data を格納した要素を線形リストの末尾に追加する。

戻り値：線形リストの先頭位置(root の値)

deleteList 関数

引　数：data(整数型)，*root(構造体のポインタ　線形リストの先頭位置)

機　能：線形リストのポインタを変更して data を線形リストから削除する。ただし，data が存在しない場合はエラーメッセージを出力する。

戻り値：線形リストの先頭位置(root の値)

［プログラム］

```
/*  線形リストに格納する要素の構造体  */
struct LIST {
    int data;
    struct LIST *next;
};

struct LIST * appendList(int data, struct LIST *root) {
    struct LIST *p;
    struct LIST *cell;
    /* 要素の記憶域を確保 */
    cell = (struct LIST *)malloc(sizeof(struct LIST));
    cell->data = data;       /* データの設定 */
    cell->next = NULL;       /* 次ポインタの設定(NULL) */
    if (root == NULL) {
        return cell;         /* 線形リストが空であれば先頭に設定 */
    } else {
        /* 線形リストの末尾までたどり要素を追加する */
        p = root;
        while(p->next != NULL) {
            p = p->next;
        }
        (4) ; /* 線形リストの末尾に要素を追加 */
        return root;
    }
}

struct LIST * deleteList(int data, struct LIST *root) {
    struct LIST *p, *old;
    /* 線形リストが空であれば戻る */
    if (root == NULL) return root;
    /* 削除するデータを探す */
    p = root;
    old = NULL;
    /* 該当するデータが見つかるか線形リストの末尾になるまで繰り返す */
    while(p->next != NULL && p->data != data) {
        (5) ;
        p = p->next;
    }
```

```
    if (old == NULL) {
        /* 削除データがリストの先頭の場合 */
        return (6) ;
    } else {
        if ( (7) ) {
            /* データが存在しない場合のメッセージ出力 */
            printf("not found¥n");
        } else {
            /* ポインタを変更 */
            (8) ;
        }
        return root;
    }
}
```

(4) の解答群

ア. p = cell　　イ. p = cell->next

ウ. p->next = cell　　エ. p->next = cell->next

(5) の解答群

ア. old = p　　イ. old = p->next

ウ. old->next = p　　エ. old->next = p->next

(6) の解答群

ア. old　　イ. root

ウ. p　　エ. p->next

(7) の解答群

ア. old == NULL　　イ. p == NULL

ウ. p->data == data　　エ. p->data != data

(8) の解答群

ア. old = p->next　　イ. old->next = p->next

ウ. p = old->next　　エ. p->next = old->next

選択問題　表計算の問題

次の表計算ソフトの記述を読み，各設問に答えよ。

この問題で使用する表計算ソフトの仕様は下記のとおりである。

COUNTIF 関数

ある範囲に含まれるセルのうち，指定された単一の検索条件に一致するセルの個数を返す。

書式：COUNTIF(範囲, 検索条件)

IFERROR 関数

値がエラー以外のときは値を返す。エラーのときはエラーの場合の値を返す。

書式：IFERROR(値, エラーの場合の値)

INDEX 関数

範囲の中から行位置と列位置を 1 から始まる相対値で指定したセルの値を返す。

書式：INDEX(範囲, 行位置, 列位置)

ISERROR 関数

値がエラーのときに TRUE を返す。それ以外のときは FALSE を返す。

書式：ISERROR(値)

LOOKUP 関数

検索範囲内で検索値が見つかると，対応範囲の行または列の同じ位置にある値を返す。なお，検索範囲は，必ず昇順に並べ替えておく必要がる。

書式：LOOKUP(検索値, 検索範囲, 対応範囲)

MATCH 関数

検査範囲内での相対的な位置を返す。位置は検査範囲で指定した範囲内の最も左上に位置するセルが1となる。照合の型で0を指定し，検査範囲に指定した検査値が含まれない場合，エラー値#N/Aが返される。

書式：MATCH(検査値, 検査範囲, 照合の型)

RANK 関数

範囲内の数値を並べたときに何番目になるか(順位)を返す。順序は，降順の場合は0，昇順の場合は1 を設定する。なお，範囲内の検査値に同じものがあれば同じ順位を返し，以降の順位に欠番が生じる。

書式：RANK(検査値, 範囲, 順序)

SUM 関数

指定した範囲に含まれる数値の合計値を返す。

書式：SUM(範囲)

VLOOKUP関数

検索値を左端に含む行を範囲の中から検索し，指定した列位置の値を返す。検索の型に0を指定すると検索値と完全に一致する値を検索し，1を指定すると検索値と一致する値がない場合に，検索値未満で一番大きい値を検索する。

書式：VLOOKUP(検索値，範囲，列位置，検索の型)

式

=に続けて計算式や関数などを入力する。

セル番地の参照

セル番地に$を付けることで，絶対番地(絶対参照)を表す。

他のワークシートの参照

「ワークシート名！セル番地」とすることで他のワークシートのセルを参照することができる。また，ワークシート名を"sheet1:sheet5"のように記述すると，ワークシート範囲を指定することができる。

Ｊテニス協会では，表計算ソフトを利用し，年間国内ランキング表を作成している。選手は男女合わせて100名おり，「登録選手名簿」ワークシートにまとめられている。２行目からA列に選手№，B列に氏名，C列に性別，D列に生年月日，E列に出身地が101行まで入力されている。

	A	B	C	D	E
1	選手№	氏名	性別	生年月日	出身地
2	W83348	今野 綾女	女	1983/12/4	神奈川県
3	W84340	今川 那奈	女	1984/3/21	宮崎県
4	M84311	田崎 将也	男	1984/5/16	和歌山県
5	W84123	石渡 あや子	女	1984/7/13	三重県
6	M84056	瀬戸 良介	男	1984/8/27	大分県
:	:	:	:	:	:
101	M04424	中田 芳正	男	2004/7/4	島根県

図１　「登録選手名簿」のワークシート

大会ごとのランキングへの重み付けと大会結果によるポイントは，「大会重み」と「大会ポイント」ワークシートに示す。なお，「大会重み」ワークシートの大会コードは昇順に並んでいる。

	A	B	C
1	大会コード	大会名	重み
2	T19028	チャレンジ東日本	0.8
3	T19038	チャレンジ西日本	0.8
4	T19125	チャレンジジャパン	1
5	T19173	グランドファイナル	1.2
6	T19188	チャンピオンシップディビジョン	1.5

図２　「大会重み」のワークシート

	A	B
1		ポイント
2	優勝	100
3	2位	80
4	ベスト4	60
5		60
6	ベスト8	40
7		40
8		40
9		40
10	ベスト16	20
11		20
12		20
13		20
14		20
15		20
16		20
17		20

図３　「大会ポイント」のワークシート

＜設問１＞　「チャレンジ東日本」ワークシートの作成に関する次の記述中の［　　］に入れるべき適切な字句を解答群から選べ。

「チャレンジ東日本」ワークシートを作成するため，１行目に大会コードと大会名，B 列，D 列の４行～19 行まで大会結果に基づき選手№を入力する。

	A	B	C	D	E
1	T19028		チャレンジ東日本		
2		男子		女子	
3		選手№	氏名	選手№	氏名
4	優勝	M99163	小町 瞬	W96158	大川 怜奈
5	2位	M96054	成瀬 直人	W96061	畑中 優
6	ベスト4	M97110	山上 陽介	W92208	津川 由美子
7		M03227	菅野 仁	W99232	吹越 優
8	ベスト8	M01121	緒方 亮	W88353	杉野 砂羽
9		M88440	角田 啓介	W90008	雨宮 あい
10		M03185	亀田 博之	W96401	土谷 砂羽
11		M04424	中田 芳正	W97095	中島 一代
12	ベスト16	M90408	有賀 明慶	W89039	浜 美嘉
13		M90477	中西 宏行	W97323	菊池 真帆
14		M94458	北原 輝信	W98320	塚本 麗奈
15		M94280	布川 三省	W98337	水野 遥
16		M00267	富沢 雅彦	W99231	谷川 綾
17		M00432	森口 満	W01398	平沢 恵望子
18		M02063	結城 雅彦	W01384	麻生 コウ
19		M02021	西谷 宏	W03121	奥村 麗奈

図４　「チャレンジ東日本」のワークシート

C 列と E 列に氏名を表示する。セル C4 に次の式を入力し，セル C5～C19，E4～E19 まで複写した。

＝［　(1)　］

(1) の解答群

ア．VLOOKUP($B4, 登録選手名簿!$A2:$B101, 2, 0)

イ．VLOOKUP($B4, 登録選手名簿!A$2:B$101, 2, 0)

ウ．VLOOKUP(B4, 登録選手名簿!A$2:B$101, 2, 0)

エ．VLOOKUP(B4, 登録選手名簿!$A$2:$B$101, 2, 0)

＜設問２＞ 「選手ランキング表(男子)」ワークシートの作成に関する次の記述中の［　　］に入れるべき適切な字句を解答群から選べ。

「選手ランキング表(男子)」ワークシートを作成するため，セル C2～G2 に大会コードを，A 列に男子の選手№を入力し B 列に氏名を検索し表示する。

	A	B	C	D	E	F	G	H	I
1	選手ランキング(男子)		大会結果						
2			T19028	T19038	T19125	T19173	T19188		
3	選手№	氏名	チャレンジ東日本	チャレンジ西日本	チャレンジジャパン	グランドファイナル	チャンピオンシップディビジョン	ポイント合計	順位
4	M88440	角田 啓介	32	16	40	24	60	172	6
5	M89204	志水 光博	0	0	0	0	0	0	24
6	M89348	岡田 洋介	0	0	40	24	30	94	13
7	M89082	岸本 広之	0	16	20	24	60	120	11
8	M90408	有賀 明慶	16	16	0	0	0	32	19
9	M90296	熊沢 丈史	0	0	0	0	0	0	24
:	:	:	:	:	:	:	:	:	:
48	M04455	紺野 一輝	0	0	0	0	0	0	24
49	M04424	中田 芳正	32	32	0	48	60	172	6

図５ 「選手ランキング表(男子)」のワークシート

３行目に大会名を表示する。セル C3 に次の式を入力し，セル D3～G3 まで複写した。

= [(2)]

(2) の解答群

ア．LOOKUP($C2, 大会重み!$A2:A$6, 大会重み!$B2:B$6)

イ．LOOKUP($C2, 大会重み!A2:A6, 大会重み!B2:B6)

ウ．LOOKUP(C2, 大会重み!$A2:$A6, 大会重み!$B2:$B6)

エ．LOOKUP(C2, 大会重み!A2:A6, 大会重み!B2:B6)

C 列にチャレンジ東日本の大会結果のポイントを表示する。セル C4 に次の式を入力し，セル C5～C49 まで複写した。なお，ベスト 16 までに入っていない選手は 0 となる。また，各大会にはランキングへの重み付けが決まっている。

= [(3)](INDEX(大会ポイント!B$2:B$17, [(4)], 1), 0) * [(5)]

(3) の解答群

ア．IFERROR　　イ．INDEX

ウ．ISERROR　　エ．VLOOKUP

(4) の解答群

ア. MATCH($A$4, チャレンジ東日本!$B$4:$B$19, 0)
イ. MATCH(A$4, チャレンジ東日本!B$4:B$19, 0)
ウ. MATCH(A4, チャレンジ東日本!B$4:B$19, 0)
エ. MATCH(A4, チャレンジ東日本!B4:B19, 0)

(5) の解答群

ア. VLOOKUP($C$2, 大会重み!$A$2:$C$6, 2)
イ. VLOOKUP(C$2, 大会重み!A$2:C$6, 2)
ウ. VLOOKUP(C$2, 大会重み!A$2:C$6, 3)
エ. VLOOKUP(C2, 大会重み!A2:C6, 3)

同様に，セル D4～G49 に「チャレンジ西日本」，「チャレンジジャパン」，「グランドファイナル」，「チャンピオンシップディビジョン」の大会結果のポイントを表示した。

H 列に選手ごとの合計ポイントを表示する。セル H4 に次の式を入力し，セル H5～H49 まで複写した。

= (6)

(6) の解答群

ア. SUM($C$4:$G$4)　　イ. SUM($C4:G$4)
ウ. SUM(C$4:$G4)　　エ. SUM(C4:G4)

I 列にポイント合計の高い順に順位を表示する。セル I4 に次の式を入力し，セル I5～I49 まで複写した。

= (7)

(7) の解答群

ア. RANK(H4, H$4:H$49, 0)
イ. RANK(H4, H$4:H$49, 1)
ウ. RANK(H4, H4:H49, 0)
エ. RANK(H4, H4:H49, 1)

選択問題　アセンブラの問題

次のアセンブラ言語CASLⅡプログラムの説明を読み，各設問に答えよ。

［プログラムの説明］

DAT 番地から始まる 10 語の連続した領域に格納済みのデータを，挿入法により降順に整列するプログラム SSORT である。

［挿入法の手順］

DAT+0～DAT+k-1番地まで降順に整列されているとき，DAT+k番地の内容を格納すべき位置を見つけて挿入する手順は，次のようになる。なお，kの値は1～9とする。

① DAT+k番地の内容をwに退避する。

② mをk-1とする。

③ DAT+m番地の内容がw以上であれば，格納すべき位置を見つけたことになるので④に進む。そうでなければ，DAT+m番地の内容をDAT+m+1番地に格納してmから1を引き，③に戻る。

④ wをDAT+m+1番地に格納する。

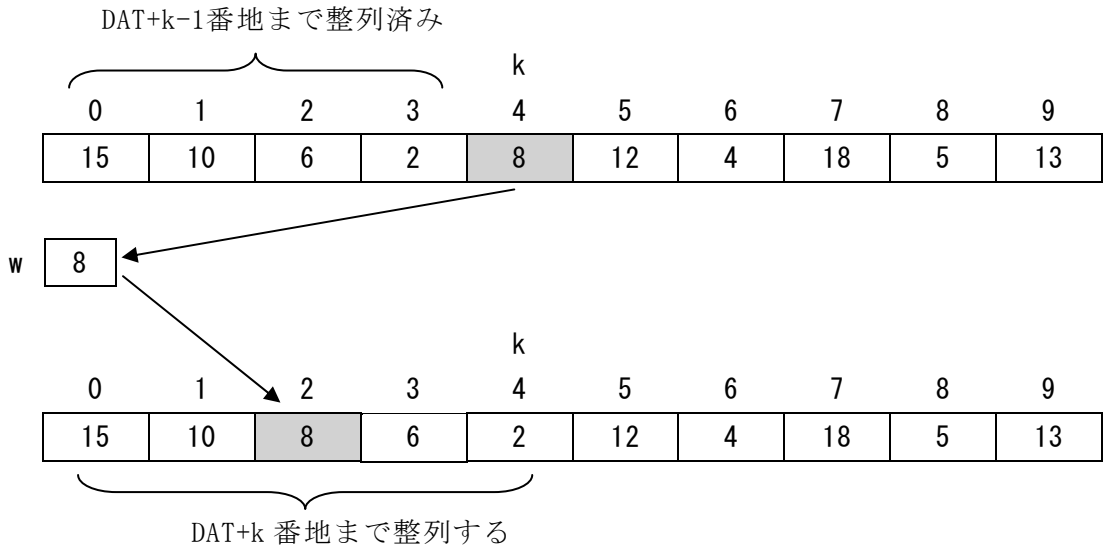


図　挿入法の例(k=4 の場合)

［プログラム］

行番号	ラベル	命令	オペランド	コメント
100	SSORT	START		
110		RPUSH		
120		LD	GR1,=1	
130	LOOP1	LD	GR0,DAT,GR1	; GR0 を w とする
140		(1)		
150	LOOP2	SUBA	GR2,=1	; m の値を 1 減らす
160		JMI	NEXT	
170		CPA	GR0,DAT,GR2	
180		JMI	NEXT	
190		JZE	NEXT	
200		LD	GR4,DAT,GR2	
210		LAD	GR3,1,GR2	
220		(2)		; データを隣りに移動
230		JUMP	LOOP2	
240	NEXT	(3)		; 挿入位置の確定
250		ST	GR0,DAT,GR3	; 挿入位置へ w を挿入
260		(4)		
270		CPA	GR1,=10	
280		JMI	LOOP1	
290		RPOP		
300		RET		
310	DAT	DS	10	
320		END		

＜設問１＞　プログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

(1) の解答群

ア．LD　GR2,=0　　イ．LD　GR2,=1

ウ．LD　GR2,GR0　　エ．LD　GR2,GR1

(2) の解答群

ア．ST　GR3,DAT,GR2　　イ．ST　GR3,DAT,GR4

ウ．ST　GR4,DAT,GR2　　エ．ST　GR4,DAT,GR3

(3) の解答群

ア．LAD　GR3,0,GR2　　イ．LAD　GR3,1,GR2

ウ．LAD　GR4,0,GR2　　エ．LAD　GR4,1,GR2

(4) の解答群

ア．**LAD　GR1,1,GR1**　　イ．**LAD　GR2,1,GR2**

ウ．**LAD　GR3,1,GR3**　　エ．**LAD　GR4,1,GR4**

＜設問２＞　次のプログラムの変更に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

次のように，１命令を JPL 命令に変更するだけで，降順から昇順に変更できる。なお，設問の都合上，ラベルとオペランドは網掛けにしている。

［変更後の命令］

行番号	ラベル	命令	オペランド
(5)		JPL	

(5) の解答群

ア．**160**　　イ．**180**　　ウ．**190**　　エ．**280**

＜設問３＞　次の実行回数に関する記述を読み，記述中の［　　］に入れるべき適切な字句を解答群から選べ。

降順に整列するプログラムにおいて，要素の移動回数が，最も少なくなるデータの並びは［　(6)　］，最も多くなるデータの並びは［　(7)　］である。

(6)，(7) の解答群

ア．	2	4	5	6	8	10	12	13	15	18
イ．	10	10	10	10	10	18	15	13	12	10
ウ．	18	15	13	12	10	2	4	5	6	8
エ．	18	15	13	12	10	8	6	5	4	2