次の問８は必須問題です。必ず解答してください。

問８　次のプログラムの説明及びプログラムを読んで，設問１，２に答えよ。

ヒープの性質を利用して，データを昇順に整列するアルゴリズムを考える。ヒープは二分木であり，本問では，親は一つ又は二つの子をもち，親の値は子の値よりも常に大きいか等しいという性質をもつものとする。ヒープの例を図１に示す。図１において，丸は節を，丸の中の数値は各節が保持する値を表す。子をもつ節を，その子に対する親と呼ぶ。親をもたない節を根と呼び，根は最大の値をもつ。

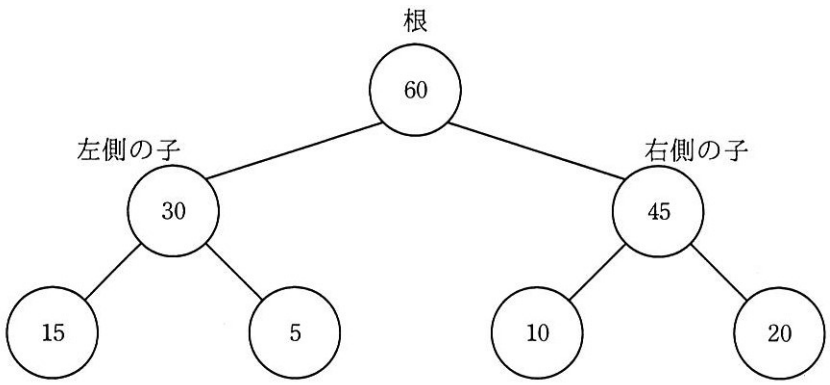


図１　ヒープの例

〔プログラム１の説明〕

(1)　配列の要素番号は，0 から始まる。

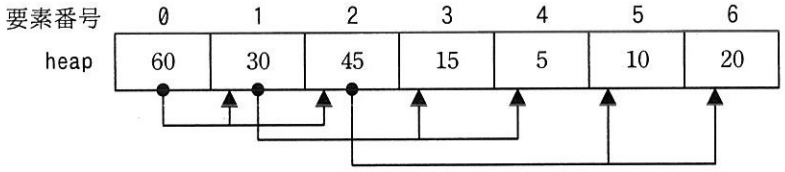
(2)　副プログラム makeHeap は，整数型の１次元配列 data に格納されている hnum 個（hnum ＞ 0）のデータを，次の①～③の規則で整数型の１次元配列 heap に格納して，ヒープを配列で実現する。この状態を，“配列 heap は，ヒープの性質を満たしている”という。

①　配列要素 heap[i]（i = 0, 1, 2, …）は，節に対応する。配列要素 heap[i]には，節が保持する値を格納する。

②　配列要素 heap[0]は，根に対応する。

③　配列要素 heap[i]（i = 0, 1, 2, …）に対応する節の左側の子は配列要素 heap[2×i＋1]に対応し，右側の子は配列要素 heap[2×i＋2]に対応する。子が一つの場合，左側の子として扱う。

(3) 図1のヒープの例に対応した配列 heap の内容を，図2に示す。



注記　矢印 ●→ は，始点，終点の二つの配列要素に対応する節が，親子関係にあることを表す。

図2　図1のヒープの例に対応した配列 heap の内容

(4) 親の要素番号と子の要素番号を関係付ける三つの関数がある。

① 整数型：lchild(整数型：i)

要素番号 i の配列要素に対応する節の左側の子の配列要素の要素番号 2×i+1 を計算して返却する。

② 整数型：rchild(整数型：i)

要素番号 i の配列要素に対応する節の右側の子の配列要素の要素番号 2×i+2 を計算して返却する。

③ 整数型：parent(整数型：i)

要素番号 i の配列要素に対応する節の親の配列要素の要素番号 (i−1)÷2（小数点以下切捨て）を計算して返却する。

(5) 副プログラム swap は，二つの配列要素に格納されている値を交換する。

(6) 副プログラム makeHeap の引数の仕様を表1に，副プログラム swap の引数の仕様を表2に示す。

表1　副プログラム makeHeap の引数の仕様

引数	データ型	入出力	説明
data[]	整数型	入力	データが格納されている1次元配列
heap[]	整数型	出力	ヒープの性質を満たすようにデータを格納する1次元配列
hnum	整数型	入力	データの個数

表2　副プログラム swap の引数の仕様

引数	データ型	入出力	説明
heap[]	整数型	入力／出力	交換対象のデータが格納されている1次元配列
i	整数型	入力	交換対象の要素番号
j	整数型	入力	交換対象の要素番号

〔プログラム1〕

```
○副プログラム: makeHeap( 整数型: data[], 整数型: heap[], 整数型: hnum )
○整数型: i, k
■ i: 0, i ＜ hnum, 1
  ・heap[i] ← data[i]                    /* heap にデータを追加 */
  ・k ← i
  ■ k ＞ 0
    ┌ a ┐
    ・swap(heap, k, [ b ] )
    ・k ← parent(k)
    ・break                              /* 内側の繰返し処理から抜ける */
```

```
○副プログラム: swap( 整数型: heap[], 整数型: i, 整数型: j )
○整数型: tmp
・tmp ← heap[i]
・heap[i] ← heap[j]
・heap[j] ← tmp
```

設問1　プログラム1中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア　heap[k] ＞ heap[lchild(k)]　　イ　heap[k] ＞ heap[parent(k)]

ウ　heap[k] ＞ heap[rchild(k)]　　エ　heap[k] ＜ heap[lchild(k)]

オ　heap[k] ＜ heap[parent(k)]　　カ　heap[k] ＜ heap[rchild(k)]

bに関する解答群

ア　heap[hnum－1]　　イ　heap[k]

ウ　parent(hnum－1)　　エ　parent(k)

設問2　〔プログラム2の動作〕の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

〔プログラム2の説明〕

(1) 副プログラム heapSort は，最初に副プログラム makeHeap を使って，配列 heap にデータを格納する。配列 heap は，整列対象領域と整列済みデータ領域に分かれている（図3参照）。last は，整列対象領域の最後の配列要素の要素番号を示している。最初は，配列 heap 全体が整列対象領域であり，このとき last の値は hnum－1 である。

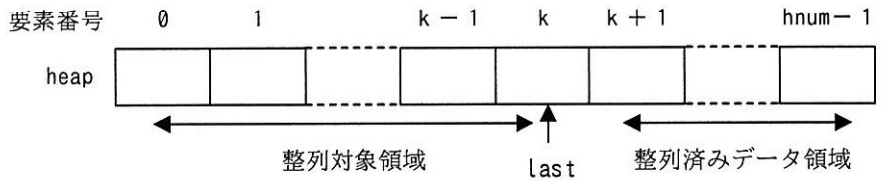


図3　配列 heap における整列対象領域と整列済みデータ領域

(2) 整列対象領域がヒープの性質を満たすとき，配列要素 heap[0] の値は，この領域での最大の値となっている。そこで，配列要素 heap[0] の値と配列要素 heap[last] の値を交換し，last の値を 1 減らして，整列対象領域の範囲を狭め，整列済みデータ領域を広げる。値の交換によって，整列対象領域はヒープの性質を満たさなくなるので，副プログラム downHeap を使って，整列対象領域のデータがヒープの性質を満たすように再構成する。これを繰り返すことによって，整列済みデータ領域には昇順に整列されたデータが格納されることになる。

(3) 副プログラム heapSort の引数の仕様を表3に，副プログラム heapSort で使用する副プログラム downHeap の引数の仕様を表4に示す。

表3　副プログラム heapSort の引数の仕様

引数	データ型	入出力	説明
data[]	整数型	入力	整列対象のデータが格納されている1次元配列
heap[]	整数型	出力	整列済みのデータを格納する1次元配列
hnum	整数型	入力	データの個数

表4　副プログラム downHeap の引数の仕様

引数	データ型	入出力	説明
heap[]	整数型	入力／出力	整列対象のデータを格納する1次元配列
hlast	整数型	入力	整列対象領域の最後の要素番号

〔プログラム2〕

```
(行番号)
 1  ○副プログラム: heapSort( 整数型: data[], 整数型: heap[], 整数型: hnum )
 2  ○整数型: last
 3  ・makeHeap(data, heap, hnum)
 4  ■ last: hnum−1, last ＞ 0, −1          ←― α
 5  |  ・swap(heap, 0, last)               /* heap[0]と heap[last]の値を交換 */
 6  |  ・downHeap(heap, last−1)            /* heap を再構成 */
 7  ■
```

```
(行番号)
 1  ○副プログラム: downHeap( 整数型: heap[], 整数型: hlast )
 2  ○整数型: n, tmp
 3  ・n ← 0
 4  ■ lchild(n) ≦ hlast
 5  |  ・tmp ← lchild(n)
 6  |  ▲ rchild(n) ≦ hlast
 7  |  |  ▲ heap[tmp] ≦ heap[rchild(n)]
 8  |  |  |  ・tmp ← rchild(n)
 9  |  |  ▼
10  |  ▼
11  |  ▲ heap[tmp] ＞ heap[n]
12  |  |  ・swap(heap, n, tmp)
13  |  |  ・return                         /* downHeap から抜ける */
14  |  ▼
15  |  ・n ← tmp
16  ■
```

〔プログラム２の動作〕

副プログラム heapSort の行番号３の実行が終了した直後のαにおける配列 heap の内容は，図２のとおりであった。このとき，副プログラム heapSort の行番号４から行番号７までの１回目の繰返し処理について考える。

副プログラム heapSort の行番号５の副プログラム swap の実行が終了した直後の配列要素 heap[0] の値は，［ c ］となる。このため，配列 heap の要素番号 0 から hnum－2 までのデータは，根に対応する配列要素 heap[0] が最大の値をもつというヒープの性質を満たさなくなる。

副プログラム heapSort の行番号６で呼び出している副プログラム downHeap は，配列 heap の整列対象領域の要素番号 0 から hlast までのデータがヒープの性質を満たすように，その領域のデータを次の手順で再構成する。

(1) 配列要素の値の大きさを比較する際に使用する要素番号を n とし，n の初期値を 0 とする。

(2) 要素番号 n の配列要素に対応する節の左側の子の要素番号を tmp に代入する。要素番号 n の子が二つあり（rchild(n) ≦ hlast），右側の子の値が左側の子の値［ d ］，右側の子の要素番号を tmp に代入する。

(3) 子に対応する配列要素 heap[tmp] の値と，その親に対応する配列要素 heap[n] の値とを比較し，配列要素 heap[tmp] の値が大きければ，配列要素 heap[n] の値と配列要素 heap[tmp] の値を交換し，tmp を次の n として (2) に戻る。ここで，副プログラム downHeap の行番号 15 において最初に n に代入する tmp の値は，［ e ］である。

c に関する解答群

ア 5　　イ 10　　ウ 15　　エ 20

d に関する解答群

ア 以下のときには　　イ 以上のときには

ウ よりも大きいときには　　エ よりも小さいときには

e に関する解答群

ア 1　　イ 2　　ウ 3

エ 4　　オ 5　　カ 6