問8　ハフマン符号化を用いた文字列圧縮（データ構造及びアルゴリズム）

（H31春-FE 午後問8）

【解答】

［設問1］　a－ア，b－イ
［設問2］　c－ウ，d－エ
［設問3］　e－オ，f－イ

【解説】

ハフマン符号化という少し難しいテーマの問題だが，符号化の方法と具体的な例を使った説明が最初にあり，この内容を理解すれば解答できるようになっている。

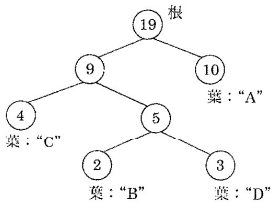
設問1は提示された文字列のハフマン符号化，及び圧縮率を求める問題である。説明に従って配列，ハフマン木，ビット列を順に作成していくことで解答を導き出せる。圧縮率も式が示されているので各文字のビット表現が導き出せれば求めることができる。

設問2は，プログラムの空欄を埋める問題だが，配列が多数出てくるので，それぞれが何を示すのかを確認しながら考えていく必要がある。

設問3はハフマン木から文字のビット表現を作成して表示するプログラムの空欄穴埋め問題である。ここでは再帰呼出しが使われているので，次にプログラムが呼ばれる前の処理，戻ってきた後の処理がどの部分になるかを意識して，説明を読むようにするとよい。

まず，図1「文字列αに対応するハフマン木の例」を参考に作成した図Aを用いてハフマン木の定義を確認する。

・○が節，直線が枝を表す。
・親である節は，子である節を常に二つもち，子の節の和を値としてもつ。
・子をもたない節（葉）は文字に対応し，出現回数を値としてもつ
・親をもたない節（根）は文字列の文字数をもつ。



図A

［設問1］

・空欄a：問題文に示された「(2)　文字の出現回数表に基づいてハフマン木を作成する」の手順に従って文字列“ABBBBBBBCCCDD”の文字の出現回数表の配列を作成する。

①　節の値を格納する1次元配列の用意

②　出現回数に基づく葉の値の格納

“A”が1文字，“B”が7文字，“C”が3文字，“D”が2文字出現するので，各文字に対応する葉の値を配列の先頭要素から順に格納すると，次のようになる。

文字	“A”	“B”	“C”	“D”
要素番号	0	1	2	3
葉（節）の値	1	7	3	2

注記：配列の要素番号は添字のことである。

③　親が作成されていない二つの節の親の節の作成

値の小さい順に二つ，同じ値の場合は配列の先頭に近い要素に格納されている節を選択する。選択した順に左側の子，右側の子とする親の節を作成し（親の節の値は二つの子の値の和），値が格納されている最後の要素の次に格納する。

最初，全ての節は親をもたない。

(a)　はじめに，節の値が小さい二つの節は要素番号0の“A”と3の“D”で，値は1と2である。親の節の値は1+2=3となり，要素番号4に格納する。

文字	“A”	“B”	“C”	“D”	
要素番号	0	1	2	3	4
節の値	1	7	3	2	3

(b)　次に親をもたず値の小さい節は要素番号2の“C”と4の節で，値は3と3である。親の節の値は3+3=6となり，要素番号5に格納する。

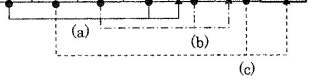
文字	“A”	“B”	“C”	“D”		
要素番号	0	1	2	3	4	5
節の値	1	7	3	2	3	6

(c)　その次に親をもたず値の小さい節は要素番号1の“B”と5の節で，値は7と6である。親の節の値は7+6=13となり，要素番号6に格納する。

文字	“A”	“B”	“C”	“D”			
要素番号	0	1	2	3	4	5	6
節の値	1	7	3	2	3	6	13

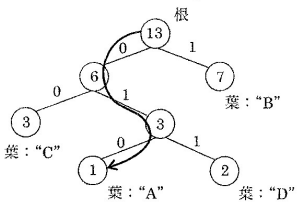
④　ここで，親が作成されていない節が要素番号6の節だけになったので，処理を終了する。

文字	“A”	“B”	“C”	“D”			
要素番号	0	1	2	3	4	5	6
節の値	1	7	3	2	3	6	13



注記：矢印 ●⟶ は，始点，終点の二つの要素に対応する節が子と親の関係にあることを示す。

文字列に対応したハフマン木は，図Bのように作成できる。



図B

次に，「(3)　ハフマン木から文字のビット列を次の手順で作成する」に従って各文字を表すビット列を作成する。

①　親の左側の子をつなぐ枝に0，右側の子をつなぐ枝に1を割り当てる。

②　文字ごとに根から対応する葉までたどると，“A”のビット表現は根の⑬から左側の子⑥へは0，続いて右側の子の要素③へは1，更に，左側の子①へは0となるので「010」となる（図Bの矢線）。

“B”のビット表現は根の⑬→⑦で「1」，“C”のビット表現は根の⑬→⑥→③で「00」，“D”のビット表現は⑬→⑥→③→②で「011」となる。したがって，空欄aには（ア）が入る。

・空欄b：文字“A”～“D”をそれぞれ2ビットの固定長で表現したときの当該文字列の総ビット長に対する圧縮率を求める問題である。文字列は全部で13文字なので固定長で表現したときのビット長は2×13=26ビットである。

圧縮後は，“A”が3ビットで1文字，“B”が1ビットで7文字，“C”が2ビットで3文字，“D”が3ビットで2文字となり，それぞれのビット長と文字数と掛け合わせると，

3×1+1×7+2×3+3×2=22ビット

となる。

問題で示された圧縮率の式に当てはめると，

$$\text{圧縮率}=\frac{3\times1+1\times7+2\times3+3\times2}{2\times13}=\frac{22}{26}=0.846$$

小数第3位を四捨五入して求めるので，圧縮率は0.85となる。したがって，空欄bには（イ）の「0.85」が入る。

［設問2］

〔プログラムの1説明〕に従って，図3の例で四つの1次元配列や用語の確認をしておく。全ての要素は－1で初期化されており，節が葉のとき，配列 left，right ともに－1である。また，節が根のときは配列 parent の値が－1となる。

文字		“A”	“B”	“C”	“D”			
配列名		要素番号						
		0	1	2	3	4	5	6
parent	親の要素番号	6	4	5	4	5	6	－1
left	左側の子の要素番号	－1	－1	－1	－1	1	2	5
right	右側の子の要素番号	－1	－1	－1	－1	3	4	0
freq	節の値	10	2	4	3	5	9	19

一つの節を表す要素組　　節が葉を示す　　節が根を示す

図C

副プログラム Huffman は葉である節の個数 size，初期化された配列 parent，left，right と，初期化された後に要素番号0から文字の出現回数が順に格納された配列 freq を受け取り，完成したハフマン木を表現する四つの配列と節の個数 size を戻す。

・行番号4：最初に実行される副プログラム SortNode は，葉の節の個数 size，配列 parent と freq，出力用の nsize，配列 node を引数として渡される。そして，親が作成されていない節を抽出し，節の値の昇順に整列して，節を表す要素組の要素番号を配列 node に格納し，その個数（親が作成されていない節の個数）を変数 nsize に格納する。

・行番号5～15：繰返し処理

・行番号6，7：SortNode で節の値の昇順に配列 node の要素が整列されているので，node[0]と node[1]が，最も小さい値をもつ要素組の要素番号と2番目に小さい値をもつ要素組の要素番号である。最も小さい値をもつ要素組の要素番号を i に，2番目に小さい値をもつ要素組の要素番号を j に格納する。

・行番号8～10：i，j で示される節の親の節の要素に値を追加する。要素番号は0から始まるため，「配列中で値が格納されている最後の要素の次の要素」の要素番号は size で示される位置になる。左側の子は値の小さい方の節なので要素番号 i，右側の子は2番目に小さい節なので要素番号は j となる。親の節の値は，二つの子の節の値の和となるので，freq[i]と freq[j]を加算した値が親の節 freq[size]の値になる。

・行番号11～12：コメントにあるように左右の子の parent の値として親の要素番号 size を設定している。

・行番号13：節を一つ追加したので，節の個数を表す size を1増やす。

・行番号14：作成した親の節も含めて副プログラム SortNode を呼び，配列 node の要素を昇順に整列させる。

・空欄c：行番号6～14の処理を繰り返すための条件が問われている。「(2)　文字出現回数表に基づいてハフマン木を作成する」の④に「親が作成されていない節が一つになるまで③（親の節の作成）を繰り返す」とあるので，この条件を当てはめればよい。親が作成されていない節の個数は，副プログラム SortNode の nsize に格納されて戻るため，繰り返す条件は nsize が2以上となり，空欄cには（ウ）の「nsize ≧ 2」が入る。

・空欄 d：副プログラム SortNode の処理を確認する。

・行番号 18：親が作成されていない節の個数 nsize を 0 で初期化
・行番号 19～24：i を 0 から 1 ずつ増やしながら i＜size を満たす間，すなわち要素番号 0 から size－1 までに設定されている全ての節に対して，親が作成されていない節を抽出し，抽出した節の要素番号を配列 node に格納し，その個数を nsize にカウントする処理を繰り返す。
・行番号 25：親が作成されていない節の要素番号が格納された配列 node の要素を昇順に整列する。

空欄 d は親が作成されていない節を示す条件である。親が作成されていないときは要素組の配列 parent の値が－1 になっている。したがって，空欄 d には（エ）の「parent[i] ＜ 0」が入る。

［設問３］

・空欄 e：〔プログラム２の説明〕から，副プログラム Encode は文字に対応する葉を表す要素組の要素番号を引数 k に与えて呼び出し，ハフマン木からその文字のビット表現を出力するプログラムである。

〔プログラム２の説明〕(3)に「行番号２の条件が成り立つとき，副プログラム Encode を再帰的に呼び出す」とある。再帰的に呼び出せるプログラムとは，プログラムの中から自分自身を呼び出すことができるもので，呼出し元に戻る条件を満たすまで何度も呼び出され，同じ処理が繰り返される。

副プログラム Encode では，親の節がある間，ハフマン木を葉から根までたどりながら再帰呼出しを行う。根までたどり着くと，配列 parent の値は－1 となる。したがって，空欄 e の判定条件には根以外（－1 以外）の節を示す（オ）の「parent[k] ≧ 0」が入る。

・空欄 f：根までたどり着いたら，次は逆に根から子をたどっていき，目的の文字に対応する節（葉）まで戻る。これが再帰呼出しから戻った後に行う処理である。

行番号５の "0" を出力するのは，親の節から左側の子へたどったときである。要素番号 k で示される現在の節が，親の節の左側か右側かを判断するには，親の節の要素組の要素番号を確認する必要がある。

親の節の要素番号は parent[k]であり，その左側の子の要素番号は left[parent[k]]で求めることができ，この値が k と等しければ左側の子であることが分かる。左側の子のときは "0" を表示し，そうでなければ右側の子なので "1" を表示する。したがって，空欄 f には（イ）の「left[parent[k]] ＝ k」が入る。

図３及び図 C の要素番号１の文字“B”の場合を例として処理の遷移を示すと，図 D のようになる。

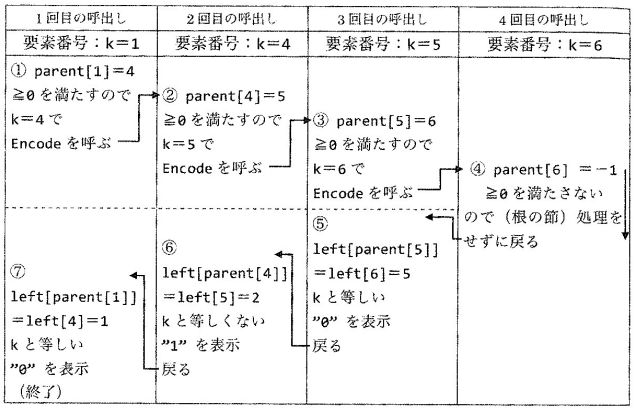


図 D

以上の再帰呼出しによる手順で，文字“B”に対応するビット表現 010 が出力される。