

問題

次の説明文のうち、間違っているものをひとつ選べ。図 1 は和の計算を行う再帰的なアルゴリズムを示す。図 2 はクイックソートのアルゴリズムを示す。partition はクイックソートの分割位置を返す関数とする。

- (a) アルゴリズム 3.4 は分割統治の考え方に基づくアルゴリズムの例である。配列を 2 分割し、分割した配列を再度 recursive_sum2 関数に引数と渡して呼び出すことで、大きな配列を小さな配列にして計算させているからである。
- (b) アルゴリズム 3.4 は分割統治の考え方に基づくアルゴリズムの例である。recursive_sum2 関数は、分割した配列 A1 と A2 の計算結果を戻り値として受け取り、その和を返す。recursive_sum 関数の引数 A の要素数が 3 個で渡された場合、この関数の内部で合計 3 回呼び出される。
- (c) アルゴリズム 6.1 は分割統治の考え方に基づくアルゴリズムの例である。クイックソートが分割統治の考え方に基づくアルゴリズムである理由は、ソート対象のデータを pivot_index で分割し、徐々に小さなソート対象にしていく並べ替えるからである。
- (d) アルゴリズム 6.1 は分割統治の考え方に基づくアルゴリズムの例である。クイックソートはソートのために再帰的に quicksort 関数を呼び出すが、実際にプログラムとしてコンピュータで実行する場合は、ソート対象のデータ個数は物理メモリ量は制約にならない。

アルゴリズム 3.4 和の計算を行う再帰的なアルゴリズム(その 2)

```
recursive_sum2(A[0], A[1], ..., A[n-1]) {  
  if (入力の引数がA[k]という1つの配列要素のみである) { return A[k]; }  
  else {  
    配列Aを半分ずつの以下の2つの配列に分割する;      --- (1)  
    A1={A[0], A[1], ..., A[(n-1)/2]},  
    A2={A[(n-1)/2+1], A[(n-1)/2+2], ..., A[n-1]}  
    x=recursive_sum2(A1);                                --- (2)  
    y=recursive_sum2(A2);                                --- (2)  
    return x+y;                                          --- (3)  
  }  
}
```

図1 和の計算を行う再帰的なアルゴリズム

アルゴリズム 6.1 クイックソート

```
quicksort(D, left, right) {  
  if (left < right) {  
    pivot_index = partition(D, left, right); --- (1)  
    quicksort(D, left, pivot_index - 1); --- (2)  
    quicksort(D, pivot_index + 1, right); --- (2)  
  }  
}
```

図2 クイックソート