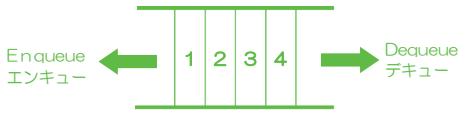
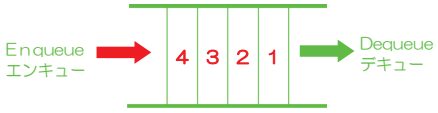
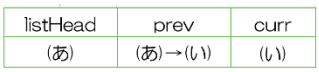
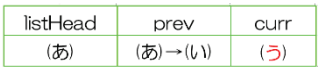
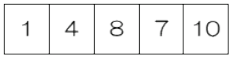
正　誤　表

2023-2024 基本情報技術者　科目 B の重点対策　第 1 版　第 1 刷（電子書籍版含む）

下記の部分に誤りがありましたので訂正させていただきます。ご迷惑をおかけし大変申し訳ございません。

<table>
<tr><th colspan="3">訂正箇所</th><th>誤</th><th>正</th></tr>
<tr><td rowspan="3">Chapter 3</td><td>1</td><td>P.89 下の表四つ目の項目</td><td>add_message(文字型)</td><td>add_message(文字列型)　（「列」が抜けている）</td></tr>
<tr><td>2</td><td>P.93 図の下 1 行目
P.94 図の下 1, 3 行目</td><td>ListEement</td><td>ListElement　（l（エル）が抜けている）</td></tr>
<tr><td>3</td><td>P.95
右下の図</td><td>キュー：先入れ先出し
 
</td><td>キュー：先入れ先出し
 

（数字の順番，エンキューの矢印の方向）</td></tr>
<tr><td>Exercise 3-3</td><td>1</td><td>P.129 吹出し 8 の左図</td><td> 
</td><td> 
</td></tr>
<tr><td rowspan="4">Exercise 3-5</td><td>1</td><td>P.142,144,146,148
〔プログラム〕
上から 11, 12 行目</td><td>if (top が 0 より大きい)
　ret ← stack[　c　]
　stack[　c　] ← 未定義
endif
top ← 　d　
return ret</td><td>if (top が 0 より大きい)
　ret ← stack[　c　]
　stack[　c　] ← 未定義
　top ← 　d　
endif
return ret</td></tr>
<tr><td>2</td><td>P.145 上の吹出し 1</td><td>スタックに関するプログラムが問われていることが分かります。スタックとは先入れ後出しのデータ構造です。</td><td>スタックに関するプログラムが問われていることが分かります。スタックとは後入れ先出しのデータ構造です。</td></tr>
<tr><td>3</td><td>P.145 下の吹出し 4</td><td>（前略）問題文にある「スタックに要素がないとき」に対応する処理だと想像できます。</td><td>「スタックに要素があるとき」に対応する処理だと想像できます。
（「誤」の青字部分を削除，「正」の赤字部分を修正）</td></tr>
<tr><td>4</td><td>P.147 の表
8 行目（push(5)），
9 行目（pop( )）</td><td><table><tr><th rowspan="2">呼出</th><th rowspan="2">呼出前の top の値</th><th colspan="4">stack</th><th rowspan="2">呼出後の top の値</th><th rowspan="2">pop の戻り値</th></tr><tr><th>1</th><th>2</th><th>3</th><th>4</th></tr><tr><td>push(5)</td><td>3</td><td>1</td><td>2</td><td>3</td><td>5</td><td>5</td><td></td></tr><tr><td>pop( )</td><td>5</td><td>1</td><td>2</td><td>3</td><td></td><td>3</td><td>5</td></tr></table>（表の 1〜7 行目，10 行目は記載を省略）</td><td><table><tr><th rowspan="2">呼出</th><th rowspan="2">呼出前の top の値</th><th colspan="4">stack</th><th rowspan="2">呼出後の top の値</th><th rowspan="2">pop の戻り値</th></tr><tr><th>1</th><th>2</th><th>3</th><th>4</th></tr><tr><td>push(5)</td><td>3</td><td>1</td><td>2</td><td>3</td><td>5</td><td>4</td><td></td></tr><tr><td>pop( )</td><td>4</td><td>1</td><td>2</td><td>3</td><td></td><td>3</td><td>5</td></tr></table></td></tr>
<tr><td>Exercise 3-6</td><td>1</td><td>P.150,152,154
〔プログラム〕
上から 13 行目</td><td>if (last が 0 より大きい)</td><td>if ((last が 0 より大きい) and (last が first 以上)</td></tr>
<tr><td rowspan="8">Chapter 4</td><td>1</td><td>P.182
図の⑧</td><td>⑧　 
</td><td>⑧　 
</td></tr>
<tr><td>2</td><td>P.185 擬似言語の表記例
上から 12, 14 行目</td><td>if (list[i] が pivot 以下) // ⑦
　left の末尾に left[i]を追加する
else
　right の末尾に left[i]を追加する
endif</td><td>if (list[i] が pivot 以下) // ⑦
　left の末尾に list[i]を追加する
else
　right の末尾に list[i]を追加する
endif</td></tr>
<tr><td>3</td><td>P.188 擬似言語の表記例
上から 3 行目</td><td>for (i を 0 まで 1 ずつ減らす) // ②</td><td>for (i を 1 まで 1 ずつ減らす) // ②</td></tr>
<tr><td>4</td><td>P.202 擬似言語の表記例
上から 6 行目</td><td>for (i を 1 から strの要素数 − pの要素数 まで 1 ずつ増やす) // ②</td><td>for (i を 1 から (str の要素数 − p の要素数 ＋ 1) まで 1 ずつ増やす) // ②
（内側のカッコは分かりやすさのため追加）</td></tr>
<tr><td>5</td><td>P.203
ずらし表</td><td>○ずらし表
<table><tr><td>A</td><td>B</td><td>A</td><td>B</td><td>C</td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>2</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>2</td></tr></table>ずらす文字数
スキップ文字数</td><td>○ずらし表
<table><tr><td>A</td><td>B</td><td>A</td><td>B</td><td>C</td></tr><tr><td>1</td><td>1</td><td>3</td><td>3</td><td>2</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>2</td></tr></table>ずらす文字数
スキップ文字数</td></tr>
<tr><td>6</td><td>P.204 擬似言語の表記例
上から 6 行目</td><td>整数型の配列: shift ← { 1, 2, 3, 4, 2 } // ②</td><td>整数型の配列: shift ← { 1, 1, 3, 3, 2 } // ②</td></tr>
<tr><td>7</td><td>P.204 擬似言語の表記例
上から 9 行目</td><td>while (i が str の要素数 − p の要素数 以下)</td><td>while (i が str の要素数 − p の要素数 ＋ 1 以下)</td></tr>
</table>

<table>
<tr><td rowspan="4">Chapter
4</td><td>8</td><td>P.204 擬似言語の表記例
右列の解説⑤</td><td>パターンを先頭から順に比較する</td><td>パターンを先頭＋スキップ文字数から順に比較する</td></tr>
<tr><td>9</td><td>P.204 擬似言語の表記例
右列の解説⑧</td><td>繰返しの最後まで到達した場合（パターンに一致する文字がなく⑤で return されなかった場合）は－1 を返す</td><td>繰返しの最後まで到達した場合（パターンに一致する文字がなく⑦で return されなかった場合）は－1 を返す</td></tr>
<tr><td>10</td><td>P.207 擬似言語の表記例
下から 4 行目</td><td>if (match)
　return i // ⑥
endif</td><td>if (match)
　return i ＋ 1 // ⑥
endif</td></tr>
<tr><td>11</td><td>P.207 擬似言語の表記例
⑥</td><td>パターンと全ての文字が一致した場合は，i を返す
このとき，i はパターンの先頭の位置となっている</td><td>パターンと全ての文字が一致した場合は，i＋1 を返す
このとき，i＋1 はパターンの先頭の位置となっている</td></tr>
<tr><td>Exercise
4-4</td><td>1</td><td>P.244 上から 4 行目</td><td>整数型の配列: ret ← {}
整数型: i, j ← 0</td><td>整数型の配列: ret ← {}
整数型: i, j ← 1</td></tr>
<tr><td>Exercise
4-7</td><td>1</td><td>P.270,P.272,P.274
〔プログラム〕
上から 11, 13 行目</td><td>if (list[i] が pivot 以下)
　left の末尾 に left[i] を追加する
else
　right の末尾 に left[i] を追加する
endif</td><td>if (list[i] が pivot 以下)
　left の末尾 に list[i] を追加する
else
　right の末尾 に list[i] を追加する
endif</td></tr>
<tr><td rowspan="6">Exercise
4-9</td><td>1</td><td>p.290,p.294,p.296,p.298,
p.300
〔拡張後のプログラム〕
7～10 行目</td><td>while (a)
　ret の末尾 に b を追加する
　j ← j ＋ 1
　tmp ← {}
　for (i を index ＋ 1 から list の要素数 まで 1 ずつ増やす)</td><td>while (a)
　if (ret が空)
　　ret の末尾 に index を追加する
　else
　　ret の末尾 に b を追加する
　endif
　j ← j ＋ 1
　tmp ← {}
　for (i を ret[j] ＋ 1 から list の要素数 まで 1 ずつ増やす)</td></tr>
<tr><td>2</td><td>P.295 吹出し 7</td><td>index ＋ 1 以降の list の要素を tmp に追加して，tmp を引数に再度関数 linear_search を呼出しています。index には一致する値の要素番号が代入されているので，index ＋ 1 以降の要素を取り出すことで，既に発見した一致する値を飛ばして，探索を行います。</td><td>ret[j] ＋ 1 以降の list の要素を tmp に追加して，tmp を引数に再度関数 linear_search を呼出しています。ret[j] には一致する値の要素番号が代入されているので，ret[j] ＋ 1 以降の要素を取り出すことで，既に発見した一致する値を飛ばして，探索を行います。</td></tr>
<tr><td>3</td><td>P.297 吹出し 3 の
右図の名称</td><td>ret
<table><tr><td>要素番号</td><td>1</td><td>2</td><td>3</td></tr><tr><td>値</td><td>4</td><td>1</td><td>4</td></tr></table></td><td>tmp
<table><tr><td>要素番号</td><td>1</td><td>2</td><td>3</td></tr><tr><td>値</td><td>4</td><td>1</td><td>4</td></tr></table></td></tr>
<tr><td>4</td><td>P.299 吹出し 5</td><td>再度，繰返しに入り，index は 2 となります。配列 tmp から，index より先の要素を取り出した配列を作成し，tmp を更新します。
tmp
<table><tr><td>要素番号</td><td>1</td><td>2</td><td>3</td></tr><tr><td>値</td><td>4</td><td>1</td><td>4</td></tr></table>tmp
<table><tr><td>要素番号</td><td>2</td><td>3</td></tr><tr><td>値</td><td>1</td><td>4</td></tr></table></td><td>再度，繰返しに入り，j は 2 となります。配列 list から，ret[j]＋1 以降の要素を取り出した配列を作成し，tmp を更新します。
list
<table><tr><td>要素番号</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr><tr><td>値</td><td>2</td><td>3</td><td>4</td><td>4</td><td>1</td><td>4</td></tr></table>tmp
<table><tr><td>要素番号</td><td>1</td><td>2</td></tr><tr><td>値</td><td>1</td><td>4</td></tr></table></td></tr>
<tr><td>5</td><td>P.299 吹出し 6
上図の要素番号</td><td>tmp
<table><tr><td>要素番号</td><td>2</td><td>3</td></tr><tr><td>値</td><td>1</td><td>4</td></tr></table></td><td>tmp
<table><tr><td>要素番号</td><td>1</td><td>2</td></tr><tr><td>値</td><td>1</td><td>4</td></tr></table></td></tr>
<tr><td>6</td><td>P.300 解答群（エ）の b</td><td>Index</td><td>index　（i（アイ）が小文字）</td></tr>
<tr><td rowspan="4">Exercise
4-11</td><td>1</td><td>P.312
上から 3 行目</td><td>問 11　次のプログラム中の a ～ c に入れる正しい答えの組合せを，</td><td>問 11　次のプログラム中の a と b に入れる正しい答えの組合せを，</td></tr>
<tr><td>2</td><td>P.312, P.316, P.318,
P.320
〔プログラム〕
○整数型: hash_search
の 3～4 行目</td><td>○整数型: hash_search(整数型の配列: list, 整数型: val)
　整数型: index ← hash(val)
　while ((b) and (index が list の要素数以下))
　　if (c)</td><td>○整数型: hash_search(整数型の配列: list, 整数型: val)
　整数型: index ← hash(val)
　while (index が list の要素数以下)
　　if (b)</td></tr>
<tr><td>3</td><td>P.313,P.320
解答群</td><td>解答群
<table><tr><td></td><td>a</td><td>b</td><td>c</td></tr><tr><td>ア</td><td>list[index] が 未定義</td><td>list[index] と val が等しい</td><td>list[index] と val が等しくない</td></tr></table>※選択肢（イ）以下は記載を省略</td><td>解答群
<table><tr><td></td><td>a</td><td>b</td></tr><tr><td>ア</td><td>list[index] が 未定義</td><td>list[index] と val が等しくない</td></tr></table>（元の空欄 b の列を削除／空欄 c→空欄 b に）</td></tr>
<tr><td>4</td><td>P.317 吹出し 6 の 1 行目</td><td>while 文の継続条件と，途中で while 文を抜ける条件が空欄となっていますが，</td><td>if 文の条件が空欄となっていますが，</td></tr>
</table>

<table>
<tr><td>Exercise 4-11</td><td>5</td><td>P.321 上から 8，11 行目</td><td>空欄 b，空欄 c：
（中略）空欄 b には「list[index] と val が等しくない」，空欄 c には「list[index] と val が等しい」が入ることが分かります。</td><td>空欄 b：
（中略）空欄 b には「list[index] と val が等しい」が入ることが分かります。（「誤」の青字部分を削除）</td></tr>
<tr><td>Exercise 4-13</td><td>1</td><td>P.341,P.343 の変数名</td><td>queue</td><td>q　（変数名を全て queue→q へ）</td></tr>
<tr><td rowspan="4">Exercise 4-15</td><td>1</td><td>P.356, P.360, P.362, P.364
〔プログラム〕8 行目</td><td>while（i が str の要素数 − p の要素数 以下）</td><td>while（i が str の要素数 − p の要素数 ＋ 1 以下）</td></tr>
<tr><td>2</td><td>P.357,P.358,P.366
問題文の下から 5 行目</td><td>ずらし表:　{1, 2, 3, 4, 2}
スキップ表: {0, 0, 0, 0, 2}</td><td>ずらし表:　{1, 1, 3, 3, 2}
スキップ表: {0, 0, 0, 0, 2}</td></tr>
<tr><td>3</td><td>P.359 吹出し 3 の上図，
P.363 吹出し 1
ずらし表とスキップ表</td><td><table><tr><td>A</td><td>B</td><td>A</td><td>B</td><td>C</td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>2</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>2</td></tr></table>パターン文字列
ずらし表
スキップ表</td><td><table><tr><td>A</td><td>B</td><td>A</td><td>B</td><td>C</td></tr><tr><td>1</td><td>1</td><td>3</td><td>3</td><td>2</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>2</td></tr></table>パターン文字列
ずらし表
スキップ表</td></tr>
<tr><td>4</td><td>P.361 吹出し 3 の 2 行目</td><td>先頭から str の要素数 − p の要素数まで</td><td>先頭から str の要素数 − p の要素数 ＋ 1 まで</td></tr>
<tr><td rowspan="4">Exercise 4-16</td><td>1</td><td>P.368,P.372,P.374,P.376
〔プログラム〕8 行目</td><td>for（s を 1 から p の要素数 まで 1 ずつ増やす）</td><td>for（s を 1 から p の要素数−1 まで 1 ずつ増やす）</td></tr>
<tr><td>2</td><td>P.373　吹出し 2 の
3 行目</td><td>次の for 文では，パターン文字列 p を順に参照して，</td><td>次の for 文では，パターン文字列 p を p の要素数−1 まで順に参照して，</td></tr>
<tr><td>3</td><td>P.375 吹出し 2 の
3 行目</td><td>ここでは，A，B，C の文字コードを，それぞれ 1，2，3 とします。</td><td>ここでは，A，B の文字コードを，それぞれ 1，2 とします。（「誤」の青字部分を削除）</td></tr>
<tr><td>4</td><td>P.375　吹出し 2 の図</td><td><table><tr><td>skip</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1:A</td><td>2:B</td><td>3:C</td><td>4</td><td>5</td><td>6</td><td>・</td><td>・</td><td>・</td><td>・</td><td>・</td><td>256</td></tr><tr><td>2</td><td>1</td><td>0</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr></table></td><td><table><tr><td>skip</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1:A</td><td>2:B</td><td>3</td><td>4</td><td>5</td><td>・</td><td>・</td><td>・</td><td>・</td><td>・</td><td>256</td></tr><tr><td>2</td><td>1</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td><td>5</td></tr></table>（「誤」の青字部分を削除／「正」の赤字部分を修正）</td></tr>
<tr><td>Chapter 5</td><td>1</td><td>P.386 (1)行列積
囲みの中 7 行目</td><td>for（k を 1 から m の行数 まで 1 ずつ増やす）//③</td><td>for（k を 1 から m の列数 まで 1 ずつ増やす）//③</td></tr>
</table>

この度，弊社書籍『2023-2024　基本情報技術者　科目 B の重点対策』に多数の誤りが発生し，
学習中の皆様に，ご不便やお手数をお掛けしましたことを，心より深くお詫び申し上げます。
皆様からのご意見やご指摘を真摯に受け止め，改善に努めてまいります。

本書籍へのお問い合わせは，下記よりお願い申し上げます。
サービスデスク：https://www.itec.co.jp/contact

この度は誠に申し訳ございません。
今後とも変わらぬご愛顧を賜りますようお願い申し上げます。

株式会社アイテック