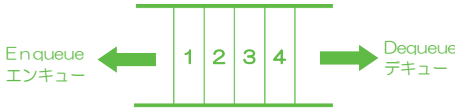


正 誤 表

2023-2024 基本情報技術者 科目 B の重点対策（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。

訂正箇所			誤	正																																																															
Chapter 3	1	P.89 下の表四つ目の項目	add_message(文字型)	add_message(文字 列 型) （「列」が抜けている）																																																															
	2	P.93 図の下 1 行目 P.94 図の下 1, 3 行目	ListEement	List E lement （1（エル）が抜けている）																																																															
	3	P.95 右下の図	キュー：先入れ先出し 	キュー：先入れ先出し 																																																															
Exercise 3-3	1	P.129 吹出し 8 の左図	<table><tr><td>listHead</td><td>prev</td><td>curr</td></tr><tr><td>(あ)</td><td>(あ)→(い)</td><td>(い)</td></tr></table>	listHead	prev	curr	(あ)	(あ)→(い)	(い)	<table><tr><td>listHead</td><td>prev</td><td>curr</td></tr><tr><td>(あ)</td><td>(あ)→(い)</td><td>(う)</td></tr></table>	listHead	prev	curr	(あ)	(あ)→(い)	(う)																																																			
listHead	prev	curr																																																																	
(あ)	(あ)→(い)	(い)																																																																	
listHead	prev	curr																																																																	
(あ)	(あ)→(い)	(う)																																																																	
Exercise 3-5	1	P.142,144,146,148 [プログラム] 上から 11, 12 行目	<pre>if (top が 0 より大きい) ret ← stack[c] stack[c] ← 未定義 endif top ← d return ret</pre>	<pre>if (top が 0 より大きい) ret ← stack[c] stack[c] ← 未定義 top ← d endif return ret</pre>																																																															
	2	P.145 上の吹出し 1	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。	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 後入れ先出し のデータ構造です。																																																															
	3	P.145 下の吹出し 4	（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。	「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）																																																															
	4	P.147 の表 8 行目 (push(5)), 9 行目 (pop())	<table><tr><th>呼出</th><th>呼出前の top の値</th><th colspan="4">stack</th><th>呼出後の top の値</th><th>pop の戻り値</th></tr><tr><th></th><th></th><th>1</th><th>2</th><th>3</th><th>4</th><th></th><th></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）	呼出	呼出前の top の値	stack				呼出後の top の値	pop の戻り値			1	2	3	4			push(5)	3	1	2	3	5	5		pop()	5	1	2	3		3	5	<table><tr><th>呼出</th><th>呼出前の top の値</th><th colspan="4">stack</th><th>呼出後の top の値</th><th>pop の戻り値</th></tr><tr><th></th><th></th><th>1</th><th>2</th><th>3</th><th>4</th><th></th><th></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>	呼出	呼出前の top の値	stack				呼出後の top の値	pop の戻り値			1	2	3	4			push(5)	3	1	2	3	5	4		pop()	4	1	2	3		3
呼出	呼出前の top の値	stack				呼出後の top の値	pop の戻り値																																																												
		1	2	3	4																																																														
push(5)	3	1	2	3	5	5																																																													
pop()	5	1	2	3		3	5																																																												
呼出	呼出前の top の値	stack				呼出後の top の値	pop の戻り値																																																												
		1	2	3	4																																																														
push(5)	3	1	2	3	5	4																																																													
pop()	4	1	2	3		3	5																																																												
Chapter 4	1	P.182 図の⑧	⑧ <table><tr><td>1</td><td>4</td><td>8</td><td>7</td><td>10</td></tr></table>	1	4	8	7	10	⑧ <table><tr><td>1</td><td>4</td><td>7</td><td>8</td><td>10</td></tr></table>	1	4	7	8	10																																																					
	1	4	8	7	10																																																														
	1	4	7	8	10																																																														
	2	P.185 擬似言語の表記例 上から 12, 14 行目	<pre>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</pre>	<pre>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</pre>																																																															
	3	P.188 擬似言語の表記例 上から 3 行目	for (i を 0 まで 1 ずつ減らす) // ②	for (i を 1 まで 1 ずつ減らす) // ②																																																															
	4	P.202 擬似言語の表記例 上から 6 行目	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	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)																																																															
	5	P.203 ずらし表	○ずらし表 <table><tr><th>A</th><th>B</th><th>A</th><th>B</th><th>C</th></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	A	B	A	B	C	1	2	3	4	2	0	0	0	0	2	○ずらし表 <table><tr><th>A</th><th>B</th><th>A</th><th>B</th><th>C</th></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	A	B	A	B	C	1	1	3	3	2	0	0	0	0	2																																	
	A	B	A	B	C																																																														
1	2	3	4	2																																																															
0	0	0	0	2																																																															
A	B	A	B	C																																																															
1	1	3	3	2																																																															
0	0	0	0	2																																																															
6	P.204 擬似言語の表記例 上から 6 行目	整数型の配列: shift ← { 1, 2, 3, 4, 2 } // ②	整数型の配列: shift ← { 1, 1 , 3, 3 , 2 } // ②																																																																
7	P.204 擬似言語の表記例 上から 9 行目	while (i が str の要素数 － p の要素数 以下)	while (i が str の要素数 － p の要素数 ＋ 1 以下)																																																																
8	P.204 擬似言語の表記例 右列の解説⑤	パターンを先頭から順に比較する	パターンを先頭 ＋スキップ文字数 から順に比較する																																																																

Chapter 4	9	P.204 擬似言語の表記例 右列の解説⑧	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	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																																									
	10	P.207 擬似言語の表記例 下から 4 行目	if (match) return i // ⑥ endif	if (match) return i + 1 // ⑥ endif																																									
	11	P.207 擬似言語の表記例 ⑥	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	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																																									
Exercise 4-4	1	P.244 上から 4 行目	整数型の配列: ret ← {} 整数型: i, j ← 0	整数型の配列: ret ← {} 整数型: i, j ← 1																																									
Exercise 4-7	1	P.270,P.272,P.274 〔プログラム〕 上から 11, 13 行目	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	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																																									
Exercise 4-9	1	p.290,p.294,p.296,p.298, p.300 〔拡張後のプログラム〕 7～10 行目	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)	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)																																									
	2	P.295 吹出し 7	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。	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。																																									
	3	P.297 吹出し 3 の 右図の名称	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>	要素番号	1	2	3	値	4	1	4	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>	要素番号	1	2	3	値	4	1	4																									
	要素番号	1	2	3																																									
	値	4	1	4																																									
	要素番号	1	2	3																																									
値	4	1	4																																										
4	P.299 吹出し 5	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。 <table><tr><td>tmp</td><td>要素番号</td><td>1</td><td>2</td><td>3</td><td>tmp</td><td>要素番号</td><td>2</td><td>3</td></tr><tr><td>値</td><td>4</td><td>1</td><td>4</td><td></td><td>値</td><td>1</td><td>4</td><td></td></tr></table>	tmp	要素番号	1	2	3	tmp	要素番号	2	3	値	4	1	4		値	1	4		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。 <table><tr><td>list</td><td>要素番号</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>tmp</td><td>要素番号</td><td>1</td><td>2</td></tr><tr><td>値</td><td>2</td><td>3</td><td>4</td><td>4</td><td>1</td><td>4</td><td></td><td>値</td><td>1</td><td>4</td><td></td></tr></table>	list	要素番号	1	2	3	4	5	6	tmp	要素番号	1	2	値	2	3	4	4	1	4		値	1	4	
tmp	要素番号	1	2	3	tmp	要素番号	2	3																																					
値	4	1	4		値	1	4																																						
list	要素番号	1	2	3	4	5	6	tmp	要素番号	1	2																																		
値	2	3	4	4	1	4		値	1	4																																			
5	P.299 吹出し 6 上図の要素番号	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>	要素番号	2	3	値	1	4	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>	要素番号	1	2	値	1	4																														
要素番号	2	3																																											
値	1	4																																											
要素番号	1	2																																											
値	1	4																																											
6	P.300 解答群 (エ) の b	Index	index (i (アイ) が小文字)																																										
Exercise 4-11	1	P.312 上から 3 行目	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，	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，																																									
	2	P.312, P.316, P.318, P.320 〔プログラム〕 ○整数型: hash_search の 3～4 行目	○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)	○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)																																									
	3	P.313,P.320 解答群	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		a	b	c	ア	list[index] が 未定 義	list[index] と val が等しい	list[index] と val が等しくない	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)		a	b	ア	list[index] が 未定 義	list[index] と val が等しくない																											
		a	b	c																																									
	ア	list[index] が 未定 義	list[index] と val が等しい	list[index] と val が等しくない																																									
	a	b																																											
ア	list[index] が 未定 義	list[index] と val が等しくない																																											
4	P.317 吹出し 6 の 1 行目	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、	if 文の条件が空欄となっていますが、																																										
5	P.321 上から 8, 11 行目	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。	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)																																										

Exercise 4-13	1	P.341,P.343 の変数名	queue	q （変数名を全て queue→q へ）																																																																					
Exercise 4-15	1	P.356, P.360, P.362, P.364 〔プログラム〕 8 行目	while (i が str の要素数 - p の要素数 以下)	while (i が str の要素数 - p の要素数 + 1 以下)																																																																					
	2	P.357,P.358,P.366 問題文の下から 5 行目	ずらし表: {1, 2, 3, 4, 2} スキップ表: {0, 0, 0, 0, 2}	ずらし表: {1, 1, 3, 3, 2} スキップ表: {0, 0, 0, 0, 2}																																																																					
	3	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	<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	A	B	A	B	C	1	2	3	4	2	0	0	0	0	2	<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	A	B	A	B	C	1	1	3	3	2	0	0	0	0	2																																							
	A	B	A	B	C																																																																				
1	2	3	4	2																																																																					
0	0	0	0	2																																																																					
A	B	A	B	C																																																																					
1	1	3	3	2																																																																					
0	0	0	0	2																																																																					
4	P.361 吹出し 3 の 2 行目	先頭から str の要素数 - p の要素数まで	先頭から str の要素数 - p の要素数 + 1 まで																																																																						
Exercise 4-16	1	P.368,P.372,P.374,P.376 〔プログラム〕 8 行目	for (s を 1 から p の要素数 まで 1 ずつ増やす)	for (s を 1 から p の要素数-1 まで 1 ずつ増やす)																																																																					
	2	P.373 吹出し 2 の 3 行目	次の for 文では、パターン文字列 p を順に参照して、	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、																																																																					
	3	P.375 吹出し 2 の 3 行目	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。	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）																																																																					
	4	P.375 吹出し 2 の図	<table><tr><td colspan="11">skip</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>	skip											1:A	2:B	3:C	4	5	6	・	・	・	・	・	256	2	1	0	5	5	5	5	5	5	5	5	5	<table><tr><td colspan="11">skip</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>・</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><td>5</td></tr></table> （「誤」の青字部分を削除／「正」の赤字部分を修正）	skip											1:A	2:B	3	4	5	・	・	・	・	・	・	256	2	1	5	5	5	5	5	5	5	5	5
skip																																																																									
1:A	2:B	3:C	4	5	6	・	・	・	・	・	256																																																														
2	1	0	5	5	5	5	5	5	5	5	5																																																														
skip																																																																									
1:A	2:B	3	4	5	・	・	・	・	・	・	256																																																														
2	1	5	5	5	5	5	5	5	5	5	5																																																														
Chapter 5	1	P.386 (1)行列積 囲みの中 7 行目	for (k を 1 から m の行数 まで 1 ずつ増やす) // ③	for (k を 1 から m の列数 まで 1 ずつ増やす) // ③																																																																					

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