■教育実施記録帳

管理No.	アルビョン・メックス FND27	Day10	Dago1						
	for…inループ、	Day10Page1値渡しと参照渡し			_				
実施内容	for…Inルーノ、	個波しと参照	<i>ii</i> 支し						
場所 実施日	2024/5/40		0.00	12.00	_				
	2024/5/10	実施時間	9:00	<u>~ 12:00</u>		:			
実施者	toraさん Masaさん	受講者		i 					
作成者	本田		作成日(開始)	2024/5/15	作成日(完了)	2024/5/15			
<time></time>	<contents></contents>		-	-		•			
	・オブジェクト入門 ウォーミングアップ								
	[. js] JSON-block								
	/**								
	* @param {Arra	* @param { Array<object></object> } arrMultipleObjects 複数のオブジェクトが入った配列							
	* @param {stri	* @param {string} +-							
	* @returns { Array(any) } pluckArrayObjs 配列の中のオブジェクトから、								
	* 第 2 引数と同じキーに対応する値だけを拾って(= pluck して)、配列に入れたもの								
	*/								
	・第1引数[参	・第1引数「参考]arrayOfobjects //関連性を持たせる(どこから、何から)							
		⇒配列の中の3つのオブジェクト:「複数形」にして解りやすくする							
	•第2引数[参	 考]key //どこ	こ書くのか、どう書くのた	が迷った箇所					
		- , , ,			己法で書く				
		⇒key&valueペアのキーが変数(key)の場合、アクセスはブロック記法で書く ・問題の逆読み(本来は必要に応じて自分が書くもの): @paraが二つ⇒引数が二つ							
		・返り値[参考] resultArray							
	[. js] test-code								
		test(pluck(arrayOfObjects, "a"), [1, 4, 7]);							
	<u> </u>								
		関数pluckは、変数arrayOfObjects内の3つの オブジェクトが入った配列を1個 と、							
	Man Production & South of Organical State of the Control of the Co								
	test(plu	ck(array0f0	bjects, "a").	[1, 4, 7]):					
	(test(pluck(arrayOfObjects, "a"), [1, 4, 7]);							
	const arrayOfObjects = [
	{ a: 1, b: 2, c: 3 },								
	{ a: 4 , b: 5 , c: 6 } , -3つのオブジェクトが入った 配列								
	{ a: 7, b: 8, c: 9 },								
];								
	変数arrayOfObjects内のオブジェクトにアクセスするための 文字列型のキーを1個 とり、								
					<u>†</u>				
	test(pluck(arrayOfObjects, "a"), [1, 4, 7]);								
	<u>第2引数と同じキーに対応する値だけを拾って入れた配列を返す</u>								
	[. js] 関数宣	[. js] 関数宣言							
	Function plu	Function pluck(arrMultipleObjects, key) {							
	const pluc	const pluckArrayObjs = [];							
	for (const	for (const arrayOfObject of arrMultipleObjects) {							
	//キーに変数(key)を使う時はブラケット記法を使う ドット記法ではうまくいかない								
	pluckArrayObjs. push(arrayOfObject[key]); Masa@DIG から全員 952								
	console.	console log("pluckArrayObis:" pluckArrayObis):							
	}	り場合は、ループ内 pluckArrayObjs: ▶ [1] で、console.lo <mark>1</mark> ()を使うと便利							
	return pluckArrayObjs; pluckArrayObjs: ► (2) [1, 4]								
	}			js: ▶ (3) [1, 4)デバッグ用の			
			Yay! Test PA:			ole.log()は後で消してくださ			
				(3) [1, 4, 7]	u				
			*			··· 👍 2			
			expected: >	(3) [1, 4, 7]					

■教育実施記録帳

	しこ 小子 かっこう	D- 10	D2						
管理No.	FND27	Day10	Page2		4				
実施内容	for…inループ、	値渡しと参照渡り	U		4				
場所					4				
実施日	2024/5/10	実施時間	9:00	~ 12:00					
実施者	toraさん	受講者	ļ	¦ -}	- 	¦ -}			
	Masaさん								
作成者	本田		作成日(開始)	2024/5/15	作成日(完了)	2024/5/15			
<time></time>	<contents></contents>								
	★for…in命令								
	[構文] f	or (‡-(変数)	8731	に要素が false					
		…繰り返しの処理		っている					
	}		true						
				任道	8の処理				
	[例] キーで指定されたオブジェクトの要素を1つずつ取出して変数に入れ繰り返し処理をする								
	[戻り値] 繰	り返し処理が終わるま	まで、処理毎の値を変	変数へ都度代入					
	// 配列の作成								
	const infoOf	ject = {							
	name: "Hana",								
	dog: true,								
	age: 12								
	};								
	// 例1								
		y in infoOfjec	t) {	//info	Ofjectkからkey要素	長を一つずつ取出して			
	console. log				key&valueペアのke				
	}			77 =		が終わるまで繰り返す			
	name								
	dog								
	age								
	// 例2								
		ev in infoOfied	t) {	//info	Ofiectkからkev要素	- 			
	for (const key in infoOfject) { //infoOfjectkからkey要素を一つずつ! console. log(infoOfject); //オブジェクト内のkey&valueペアを								
	Consumer Tog (Timour Ject)								
	// 戻り値				/// 3-7-	がべいるのでは水りをう			
		ana" dog: true	age: 12}						
	{name: "Hana", dog: true, age: 12} {name: "Hana", dog: true, age: 12}								
	{name: "Hana", dog: true, age: 12}								
	// 例3	and / dog. true	., ager 123						
		ey in infoOfjec	+) {	//info	Ofiectkからとので更多	まを一つずつ取出して			
	· ·	g(infoOfject[ke		//infoOfjectkからkey要素を一つずつ取出して //オブジェクト内のkey&valueペアのvalue(値)を表示する					
	3	, thirour Jeot [Ne	31/1	// ハンクエンLk30)		ilue(値)を表示する が終わるまで繰り返す			
	// 戻り値				//ブリ手/	ファスイン ひみ て小木ソルベタ			
	// 庆り恒 Hana								
	true								
	12 ◇「"kev"]と文字列にすると"kev"という名前のキー(要素)を控しに行くので注音!!								
	☆["key"]と文字列にすると"key"という名前のキー(要素)を探しに行くので注意!! // 戻り値 "key"という要素はないため…								
		ey Cいり安系はな	0 1/CØ)						
	undefined								
	undefined								
	undefined								

■教育実施記録帳

■狄月天/					\neg				
管理No.	FND27	Day10	Page3		_				
実施内容	for…inループ、	値渡しと参照	値渡しと参照渡し						
場所					4				
実施日	2024/5/10	実施時間	9:00	<u>~ 12:00</u>					
実施者	toraさん Masaさん	受講者		- 					
 作成者	本田	<u> </u>	作成日(開始)	2024/5/15	作成日(完了)	2024/5/15			
<time></time>	<contents></contents>	_ i	11111111(111112)	, , , ,	111111111111111111111111111111111111111	, , , ,			
	★プリミティブ型データ								
	・数値型(number)、文字列型(string)、論理型(boolean)								
	★オブジェクト型データ								
	・配列(array)、オブジェクト(object)、関数(function)								
	★プリミティブ型データとオブジェクト型データの違い								
	! 「値を変数に格納する方法」が異なる								
	☆プリミティブを	☆プリミティブ型: 値そのもの が直接メモリに格納 (値渡し)							
	・ある変数から別の変数へ値を渡すと、変数は 個別に値を持つ ことになる								
		6:	_						
	let x =		x 7						
	x = 7;	^,	у 6						
		.log(x); // 7							
		.log(x); // /							
	Conso te	.tog(y), // 6							
	☆オブジェクト型:参照値(値を実際に格納しているメモリ上のアドレスのようなもの)を格納(参照渡し)								
	・ある変数から別の変数へ参照値を渡すと、両変数ともに 同じ一つの参照元の値 を持つことになる								
	・参照元の値(データ)を変化させたいのか、させたくないのか意識する必要あり								
	let x = [1, 2	2 21.	a0001 [1,	, 2, 3, 4]					
	let y = x;		a0001						
	x.push(4);	У	a0001						
); // [1, 2, 3, 4]							
	console.log(y); // [1, 2, 3, 4]							
	★メモリといえば								
	·RAM (Random Access Memory)								
	⇒コンピュータが一時的にデータを保管(格納)するするためのパーツ。								
	⇒セカンダリメモリ(ストレージ)より高速でアクセスできる								
	☆ ここにコード中のデータが格納される								
	サコンピューム	タの中で起きてい	ハスアと						
	・コードを書		ישכנ						
	・書いたコードを保存する ・メプログラムを実行する								
	- ・メノログラムを美行する ・保存先から コード中のデータをメインメモリに格納する								
	・保存元から コート中のテータをメインメモリに倍約9 る ・CPUがメインメモリ中のデータにアクセスして演算する								
	· CI · O/J·X·			- y W					
	コード		IDD / SSD セカンダリメモリ)	(メインメモリ。単に 「メモリ」とも呼ばれる	5.)	CPU			
	const a = "H const b = "W console.log	world";	_	+	→	Intell piotinam			
	- Consorer tog	, ,							
			G. G.						