const \_ = {

clamp(number, lower, upper) {

const lowerClampedValue = Math.max(number, lower);

const clampedValue = Math.min(lowerClampedValue, upper);

return clampedValue;

},

inRange(number, start, end = 0) {

let isInRange = false;

if (typeof number === 'number' && typeof start === 'number' && typeof end === 'number'){

if (number >= Math.min(start, end) && number < Math.max(start, end)) {

isInRange = true;

}

} else {

console.log('All input arguments must be numbers!');

} return isInRange;

},

words(string) {

const words = string.split('');

return words;

},

pad(string, length) {

if(length <= string.length)

{

return string;

} else {

let padding = length - string.length;

let startPaddingLength = Math.floor(padding / 2);

let endPaddingLength = Math.ceil(padding / 2);

const paddedString = '';

return ''.repeat(startPaddingLength) + string + ''.repeat(endPaddingLength);

}

},

has(object, key) {

return object !== undefined && object[key] != null;

},

invert(object) {

const invertedObject = {};

for(let key in object) {

let originalValue = object[key];

invertedObject[originalValue] = key;

};

return invertedObject;

},

findKey(object, predicate) {

for (let key in object) {

let value = object[key];

let predicateReturnValue = predicate(object[key]);

if (predicateReturnValue) {

return key;

};

return undefined;

};

},

drop(array, number) {

let newArray = [];

if(number === undefined) {

number = 1;

}

else if(number < 0) {

number = 0;

}

for(let i = 0; i < array.length; i++) {

newArray.push(array[i]);

}

return newArray.slice(number);

},

dropWhile(array, predicate) {

let droppedArray = [];

for(let x = 0; x < array.length; x++) {

if(!predicate(array[x], x, array)) {

droppedArray = array.slice(x);

};

}

return droppedArray;

},

chunk(array, size) {

const arrayChunks = [];

if(!size) {

size = 1;

}

for(let i = 0; i < array.length - 2; i++) {

const arrayChunk = array.slice(i + i, i + size);

arrayChunks.push(arrayChunk);

size++;

//console.log(arrayChunks);

}

return arrayChunks;

}

};

// Do not write or modify code below this line.

module.exports = \_;