# Lab: Error Handling and Exceptions

Problems for in-class lab for the [“JavaScript Advanced” course @ SoftUni](https://softuni.bg/courses/javascript-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/344/>.

## Sub Sum

Write a JS function to sum a **range** of numeric elements from array. The function takes three parameters – the first is an **array**, the second is **start index** and the third is an **end index**. Both indexes are **inclusive**. Assume array elements may not be of type Number and cast everything. Implement the following error handling:

* if the **first element** is not an array, return NaN
* if the **start index** is less than zero, assume it is zero
* if the **end index** is outside the bounds of the array, assume it points to the last index of the array

### Input / Output

Your function must take three **parameters**. As output, **return** the resulting **sum** as instructed.

### Examples

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| subsum([10, 20, 30, 40, 50, 60], 3, 300) | 150 |
| subsum([1.1, 2.2, 3.3, 4.4, 5.5], -3, 1) | 3.3 |
| subsum([10, 'twenty', 30, 40], 0, 2) | NaN |
| subsum([], 1, 2) | 0 |
| subsum('text', 0, 2) | NaN |

## Playing Cards

Create a JS class **Card** to hold a card’s face and suit, both set trough the constructor. **Throw** an error if the card is initialized with invalid **face** or **suit** or if an attempt is made to change the **face** or **suit** of an existing instance to an invalid value.

* Valid card faces are: 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A
* Valid card suits are: S (♠), H (♥), D (♦), C (♣)

Both face and suit are expected as an uppercase string. The class also needs to have a toString() method that prints the card’s face and suit as a string. Use the following UTF code literals to represent the suits:

* \u2660 – Spades (♠)
* \u2665 – Hearts (♥)
* \u2666 – Diamonds (♦)
* \u2663 – Clubs (♣)

### Input / Output

The constructor function must take two string parameters. The toString() method of the class must return a string.

Submit the class definition.

### Examples

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| console.log('' + new Card('A', 'S')); | A♠ |
| console.log('' + new Card('10', 'H')); | 10♥ |
| console.log('' + new Card('1', 'C')); | Error |

## Deck of Cards

Write a JS function that takes a deck of cards as a string array and print them as a sequence of cards (space separated). Print "Invalid card: [card]" when an invalid card definition is passed as input. Use the solution from the previous task to generate the cards.

### Input / Output

The function must take an array of strings as parameter. As output, print on the console the list of cards as strings, separated by space.

Submit a function that contains the Card class definition and other logic.

|  |
| --- |
| deck.js |
| function printDeckOfCards(cards) {  class Card {  *// TODO use class definition from previous taks*  }  *// TODO process input*  } |

### Examples

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| printDeckOfCards(['AS', '10D', 'KH', '2C']); | A♠ 10♦ K♥ 2♣ |
| printDeckOfCards(['5S', '3D', 'QD', '1C']); | Invalid card: 1C |

## Time Span

Create a JS class TimeSpan that holds a seconds property. The constructor takes three numeric parameters for hours, minutes and seconds that initialize the value of the internal property. The class must work with negative values and overflow values (e.g. more than 59 seconds), but throws a RangeError if any of the parameters is invalid with the following error message:

Invalid {hours/minutes/seconds}: {parameter value}

Replace the parts in curly braces with the relevant value. See the examples for details.

The class must have a toString() method that prints a string with the held timespan in format 'hh:mm:ss'. The minutes and seconds must have leading zeros if they are less than 10.

### Input / Output

The class constructor must take three parameters. As output of the toString() method **return** a string.

### Examples

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| console.log('' + new TimeSpan(7, 8, 5)); | 7:08:05 |
| console.log('' + new TimeSpan(12, 76, -5)); | 13:15:55 |
| console.log('' + new TimeSpan('', 2.5, {})); | RangeError: Invalid hours: |
| console.log('' + new TimeSpan(3, 2.5, {})); | RangeError: Invalid minutes: 2.5 |
| console.log('' + new TimeSpan(3, 2, {})); | RangeError: Invalid seconds: [object Object] |