PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2513 — Tecnologías y Aplicaciones Web

Tarea 2

Entrega

- **Fecha y hora:** Sábado 29 de abril de 2023, 20:00 hrs
- **Lugar:** Buzón de Tareas en Canvas

Indicaciones

El objetivo de esta entrega es programar usando JavaScript en el ambiente dado por NodeJs accediendo a recursos de la máquina, trabajando con asincronía y objetos según corresponda. Para lo anterior se deberá entregar su solución a 2 problemas (menores) de programación, la respuesta de cada desafío debe incluir:

- Diseño de su solución
- Código que solucione el problema planteado
- Código documentado (con comentarios, no es requerido más por ahora)
- Seguir los lineamientos de programación dados por airbnb, para eso vean: https://github.com/airbnb/javascript
- Incluir un archivo README (por desafío) con las consideraciones, supuestos y observaciones que ustedes consideren relevantes para la instalación (si procede) y ejecución de sus implementaciones

Desafío 1: Sistema de aviso de riesgo sanitario para viajeros

Este sistema se encarga de determinar si cierto lugar (destino) posee restricciones sanitarias (malaria, covid, dengue u otra restricción sanitaria) que impidan el viaje al destino.

El proceso es el siguiente:

- Se recibe un código de destino, este código es alfanumérico, tres letras y dos dígitos, por ejemplo CLP23 (deben programar la función que reciba el código alfanumérico recién descrito)
- Se debe conectar al sistema e inicializarlo usando la función abrirSistemas
- Inicializado el sistema (no antes), se debe buscar el destino de acuerdo al código entregado. Para ello deberá usted implementar el código y las funciones necesarias
- Asociado al destino, estarán las restricciones sanitarias (en el mismo objeto).
- Se debe invocar a la función que busca restricciones entregando como parámetro el código de restricción recuperado (restriccionesSanitarias). Debe modificar la función callback invocada por el setTimeout(). Esta función debe entregar un número random entre 10 y 100 y compararlo con el código de restricción (pasado como parámetro) si el número random es menor que el código de restricción, entonces quiere decir que NO hay restricción de viaje. De lo contrario, si es mayor o igual, la función retorna que si hay restricciones. **NOTA:** el código de restriccion siempre será un entero entre 0 y 90

- Si no hay problema con restricciones en el lugar de destino, se imprimee en consola (console.log) el atributo status del objeto retornado.
- Si todo está bien con el destino se procede, entonces, a emitir un certificado de viaje seguro. Para ello invoca a la función emiteCertificado()
- Si hay problemas, se debe indicar que hay restricciones en el destino. Para ello se imprime el status del objeto retornado. El status contiene el mensaje de restricción
- Terminado lo anterior, se procede a cerrar el sistema invocando a la funcion cerrarSistemas

Usted debe:

- Utilizar la biblioteca planificacion.js
- Utilizar la siguiente estructura de datos un arreglo de objetos llamado restricciones que tiene la siguiente forma [codigo: ABC12, destino: "SANTIAGO", restriccion: 13, codigo: XYZ23, destino:"ARUBA", restriccion: 57, ..., codigo: RST77,destino:"BRUSELAS, restriccion: 3]. **NOTA:** los valores puestos son de ejemplo, el largo del arreglo es variable.

  En síntesis la estructura es un arreglo de objetos. Los objetos tienen 3 atributos: codigo (alfanumerico, 3 letras y 2 digitos), destino (string) y restriccion (entero)
- Modificar el código de la funcion restriccionesSanitarias para incorporar el reject en caso de algún error (no de restricciones sanitarias, eso no es un error, es una condición)
- Hacer todas las modificaciones y validaciones que estime conveniente, puede crear estructura de datos adicionales. **NOTA:** El código va a tener errores, deberán corregir tanto sintaxis como semántica.
- No puede cambiar la naturaleza de las funciones en cuanto retornan promesas y utilizan (las que lo utilicen) setTimeout junto al objeto que retornan
- Justifiquen cualquier cambio de las funciones y documéntenlo en su diseño
- Para corregir su tarea el ayudante hará node index.js eso quiere decir que su programa principal ha de estar en index.js (puede tener otros .js que sean invocados por index.js, no es problema. En particular su programa principal deberá leer planificacion.js). Adicionalmente usted deberá declarar al inicio de su código, las estructuras solicitadas de tal forma que se puedan modificar con los set de pruebas que el ayudante tendrá para probar su código.

La salida esperada en caso de poder viajar es del tipo (esto es SOLO ilustrativo)

```
Puede iniciar el proceso
Sin restricciones sanitarias a destino:[destino]
Certificado emitido para destino [destino]
sistemas cerrados de forma segura
```

Entre [ ] se encuentra el destino que se determinó según código entregado.

Desafío 2: Métodos de encriptación caseros

Usted deberá programar un método casero de encriptación de un archivo. Para ello, creará un programa que recibirá tres parámetros:

- Uno es el parámetro de "rotación" que será un número entero,
- El segundo parámetro, un string, es la ruta donde se deberá buscar al archivo (la ruta es RELATIVA al lugar donde se corra el programa. Si el archivo está en el mismo directorio donde se corre el programa la ruta será "*/*"
- El último parámetro, un string, que será el nombre de un archivo que se encontrará en la ruta especificada anteriormente

El programa deberá leer el archivo desde la ruta entregada y aplicar un algoritmo de "rotacion" para cifrarlo. Lo que realiza este algoritmo de cifrado, es considerar las letras del abecedario como en un arreglo circular y procede a reemplazar una letra dada por su correspondiente letra dada por "n" casillas más allá según indica el número entero del parámetro "rotación". Así pues, un parámetro de rotación "1", moverá la letra a de una palabra y la reemplazará por la b, la b por la c y así sucesivamente. Así pues la palabra "zapato", con una rotación "1" quedará como "abqbup" en el caso que solo se consideren las letras del alfabeto. Note que en este caso la z pasa a ser "a" dada la característica circular del arreglo.

Para su algoritmo de rotación (este desafío) considere el alfabeto de la a a la z incluyendo la ñ, además como parte de su arreglo "circular" las letras á, é, í, ó, ú y ü. Así pues, con este "nuevo" arreglo, zapato quedaría, con una rotación de 1, como "ábqbup".

Los dígitos y otros símbolos (como puntuación , paréntesis, etc.) no son "rotados" pero si son considerados. Es decir, si ciframos el string "1.- zapato", el cifrado quedaría como "1.- ábqbup" (sin considerar las comillas, claro está).

La salida de su programa es un archivo, en la misma ruta del archivo original con el mismo nombre del archivo original pero agregando, en el nombre, el texto "_cifrado". Así pues si el nombre del archivo es "ejemplo.txt", el archivo cifrado quedará como: "ejemplo_cifrado.txt".

NOTA: el número del parámetro de rotación puede ser cualquier entero positivo. En caso de entregar un cero, el texto contenido en el archivo quedará igual en el archivo de destino (es decir, en el archivo de sufijo _cifrado)

Para probar este programa el ayudante hará: node index.js eso significa que su programa principal debe estar en index.js

Detalles de la entrega

La entrega de todos los archivos se hará en un buzón de tareas en Canvas. Se debe entregar un zip que contenga todos los archivos, siguiendo la estructura pedida más adelante. En el nombre del zip se debe especificar el número de alumno, nombre y apellido, de la siguiente manera:

12345678_Perez_Juan.zip

NO se aceptarán:

- Entregas por correo electrónico (ya sea al profesor o ayudantes)
- Entrega en otro sistema que no sea el que se ha provisto para estos efectos

Recomendaciones

- Piensen en distintas opciones/algoritmos de solución y la mejor forma de abordar su trabajo. No se queden con la primera idea que les venga a la cabeza
- Pregunten y consulten, usen foro, colaboren entre ustedes (NO COPIEN). Los ayudantes están para apoyarlos
- Trabajen con tiempo, no esperen a último momento para comenzar con la tarea o despejar dudas
- Recuerden que esto es programación de "lado servidor" usando Nodejs. Esto no se correrá ni probará en el browser. Toda interacción es en un "terminal" por lo que no se espera interfaces gráficas de ningun tipo, solo "línea de comandos"

Dudas

Para que todo el curso se vea beneficiado, hagan sus preguntas en las *issues* del repositorio del curso. No se responderá ninguna duda de tareas por correo electrónico.