2715. Timeout Cancellation

Goal: Create a function named cancellable. Input:

- fn → a function to run later
- args → an array of arguments for fn
- t → delay in miliseconds before calling fn.

Behaviour:

- After t ms, fn(...args) should be automatically executed - unless the user cancels it first
- To cancel, cancellable must return a cancelFn, if one calls cacelFn() before t ms have passed, fn should not run.

It will look like:

```
const cancel = cancellable(log, [2], 20);

setTimeout(cancel, 50);
```

setTimeout() ⟹ Allows us to schedule the execution of a function after an amount of time.
"Stop watch"

```
const timeoutId = setTimeout (fn, delay)
```

clearTimeout () ⟹ can cancel a timeout before it triggers.

$\boxed{\text{clearTimeout (timeout Id)}}$

Here we are getting control over delaying execution.

important features like :

→ debouncing

→ timed prompts

→ aborting delaying actions

```
function cancellable = (fn, args, t) {
      const  cancelFn = ( ) => clearTimeout (timer);
      const  timer = setTimeOut (( ) => fn(...args), t);
      return cacelFn;
   }
 function example (x){ console.log (x*5);}


 const cancel = cancellable (example,[2] , 2000);
  setTimeout (cancel, 5000);
```