L3: King Kong

Required compiler flags: `-Wall -fsanitize=address,undefined`

Write a program to know how King Kong will hike through the New York city. The program should accept two arguments:

- N – No of buildings in the New York city
- T – the time of the simulation given in seconds.

($N > 2$ and $T \geq 5$) – you should check the correctness of the arguments provided.

The city of New York is represented as an array of N elements. King Kong has to go from one building to another starting from the beginning (index at 0) to the end (index at N-1). King Kong can't look below so he can only jump to a taller building from a shorter one.

First the Program creates an N element array initialized with zeroes and prints it. Then the program starts N threads and each thread sleeps for a random time in range $[100, 500]$ milliseconds and increments the respective value of the array by 1. Thread 1 increments `cell[0]`, thread 2 increments `cell[1]` and so on.

The main thread of the program keeps the position of King Kong starting from zero and every 1 second King Kong checks if the `[i+1]`-th building is taller than the `[i]`-th building and if so King Kong jumps from `[i]` to `[i+1]` and prints the array and new position after each jump. The program stops when King Kong reaches at the end of the array or T seconds passed.

After King Kong reaches the last cell or T seconds have passed, the main thread cancels all the building threads (N) and prints the final position of King Kong and the array.

Each cell of the array must be protected by a dedicated (separate) mutex.

When the program receives `SIGINT` the all array values are reset to 0.

In this task you cannot use global/ static variables.

Stages:

1. 4 p. Check the arguments passed by the user. Creating N threads. Each thread is printing 'Created'. Main thread waits for all threads to finish.
2. 4 p. Create the array initialize with zero. Start N threads and thread's logic. Implement the King Kong Jump and Building height code. Main thread loops end when King Kong reaches last building of the array. All threads are canceled and the program ends. No mutexes are required yet.
3. 3 p. Add mutex protection to the array.
4. 3 p. Main Thread counts down the simulation time. After T seconds cancels the building threads, waits for them to finish and prints the final state of the array and King Kong's position.
5. 3 p. SIGINT is supported. (it's a good idea to use alarms to wait a second so both actions will be performed in reaction to a signal).

Copy your solution to a directory: `/home2/samba/karwowskij/unix/`
☐ **The solution was copied to the directory** (ticked by the student).

Stage	1	2	3	4	5	Sum
Points	4	4	3	3	3	17
Result						