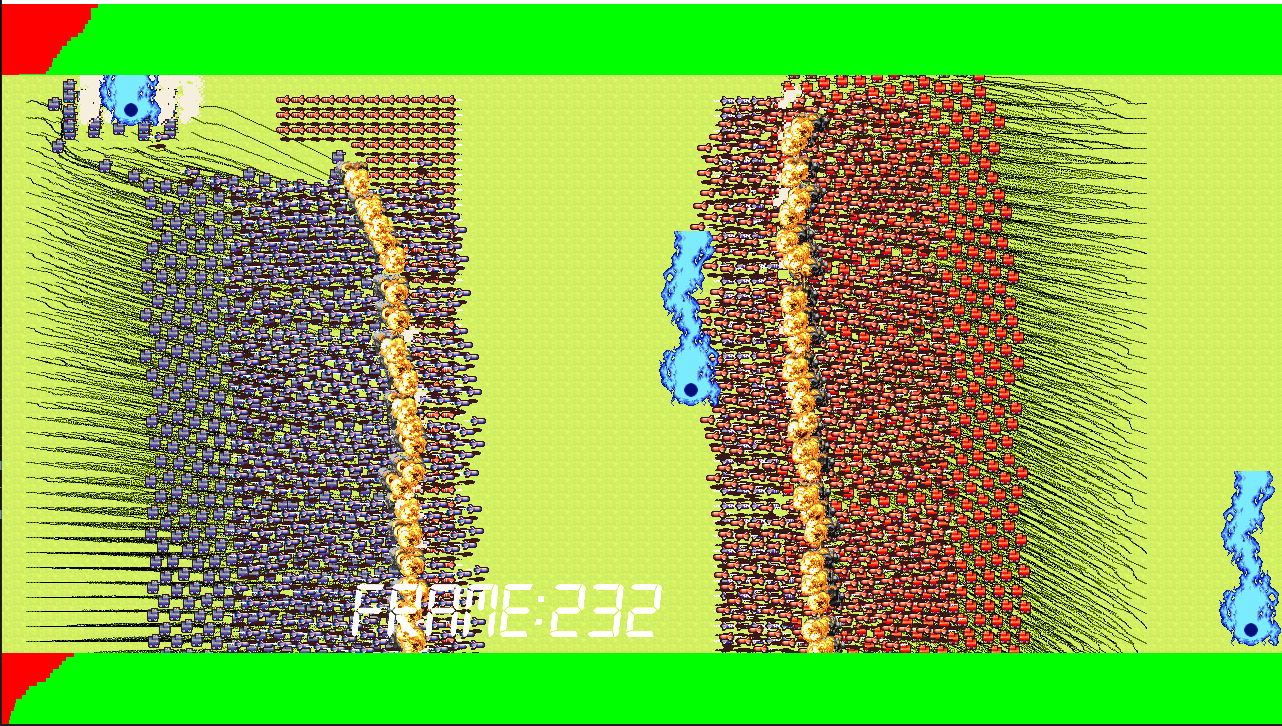
# Machine intelligence



## Inhoud

[Machine intelligence 1](#_Toc76014771)

[Inhoud 2](#_Toc76014772)

[Samenvatting 4](#_Toc76014773)

[BIG O ANALYSIS 5](#_Toc76014774)

[Big o van de meegegeven code. 5](#_Toc76014775)

[Tank collision detection: 5](#_Toc76014776)

[FindClosestEnemy: 5](#_Toc76014777)

[Update rockets: 6](#_Toc76014778)

[Update Particle\_Beams: 6](#_Toc76014779)

[InsertionSortTanksHealth: 6](#_Toc76014780)

[Toegepaste verbeteringen 7](#_Toc76014781)

[De tank collision detection: 7](#_Toc76014782)

[Draw Sorted Healthbars: 7](#_Toc76014783)

[Update Particle\_Beams: 8](#_Toc76014784)

[Insertion Sort Tank Health: 8](#_Toc76014785)

[Concurrency 8](#_Toc76014786)

## Samenvatting

Eerste run. Zonder optimalisatie.



Algoritmische optimalisaties:

* Toevoegen van een grid. Toegepast op tank collision. Speedup 1.0 naar 1.9
* Healthbars worden getekend met alleen benodigde informatie. Ipv alle objecten binnen Tank. Speedup van 1.9 naar 2.0
* Healthbars Sorteren d.m.v een merge sort. In plaats van de huidige insertion sort. Geen grote speedup.
* Particle beams controleren alleen collision op tanks binnen de opgegeven locatie. Ipv alle tanks op het veld. Sim duurt korter maar niet genoeg voor een speedup.
* Toevoegen van grid op rocket collision speedup van 2.0 naar 2.7

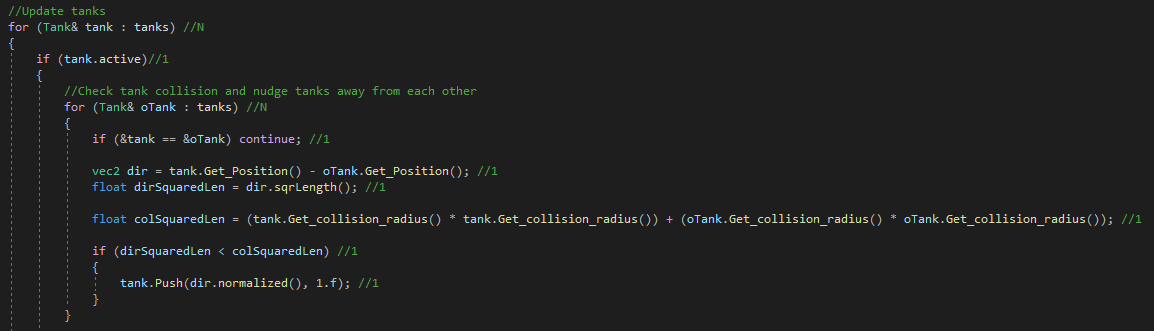
Concurrent optimalisaties:

* Threads worden dynamisch aangemaakt.

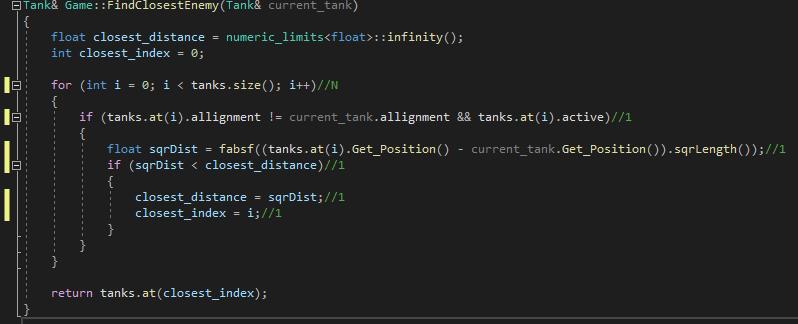
## BIG O ANALYSIS

### Big o van de meegegeven code.

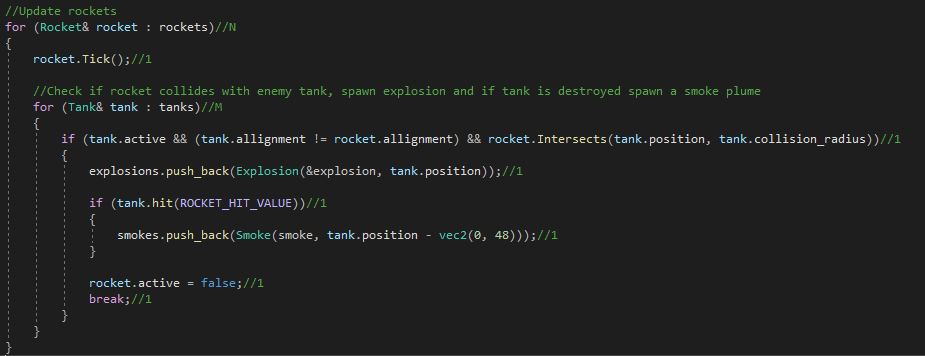
#### Tank collision detection:

O(N6N), O(N^2)

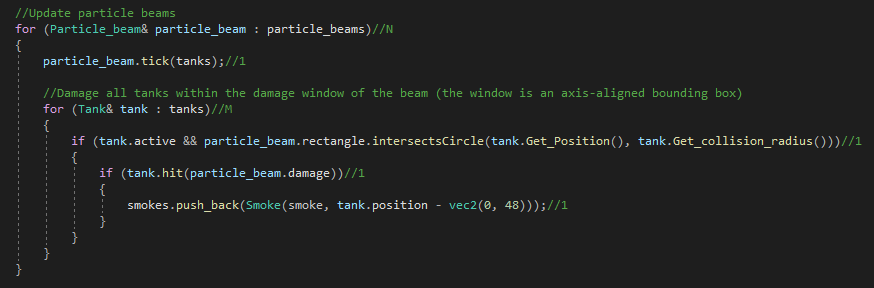
#### FindClosestEnemy:

O(5N), O(N)

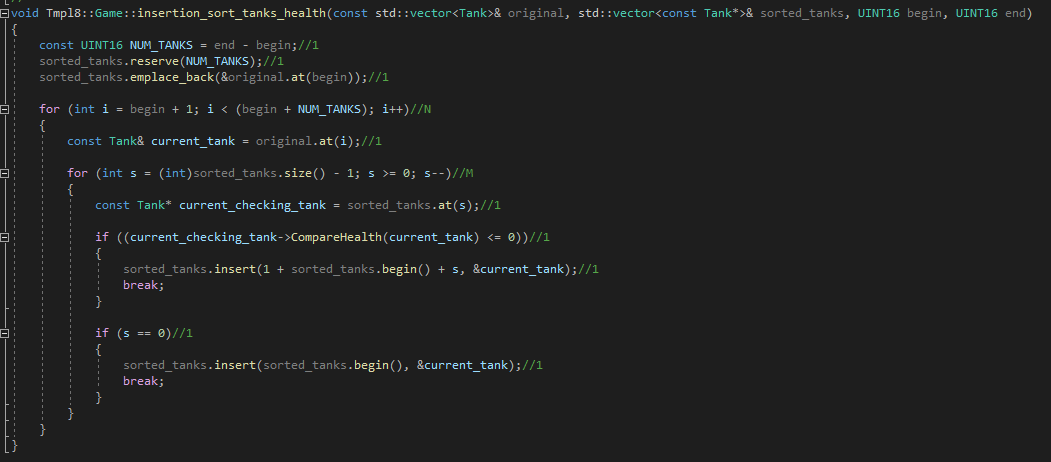
#### Update rockets:

O(N6M), O(NM)

#### Update Particle\_Beams:

O(N3M), O(NM)

#### InsertionSortTanksHealth:

O(N5N), oftewel O(N^2)

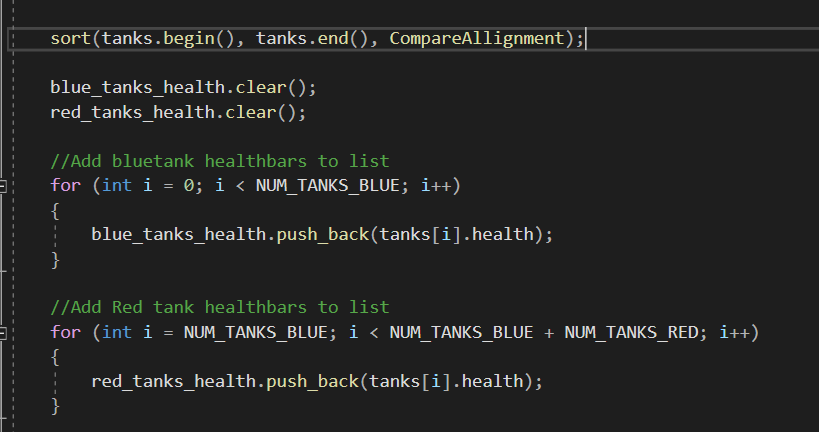
### Toegepaste verbeteringen

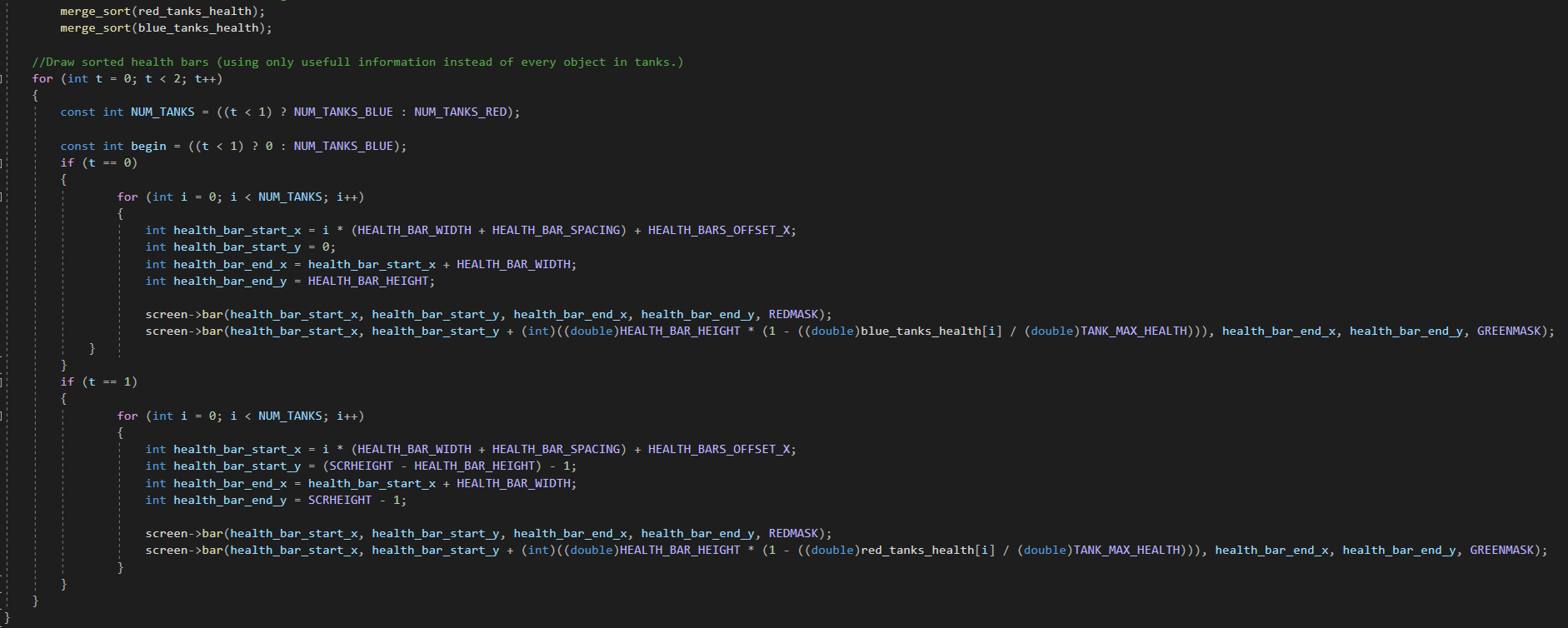
#### De tank collision detection:

is aangepast door dit uit te voeren op een grid.

Micky mag de rest invullen + plaatjes met Big O waarden

#### Draw Sorted Healthbars:



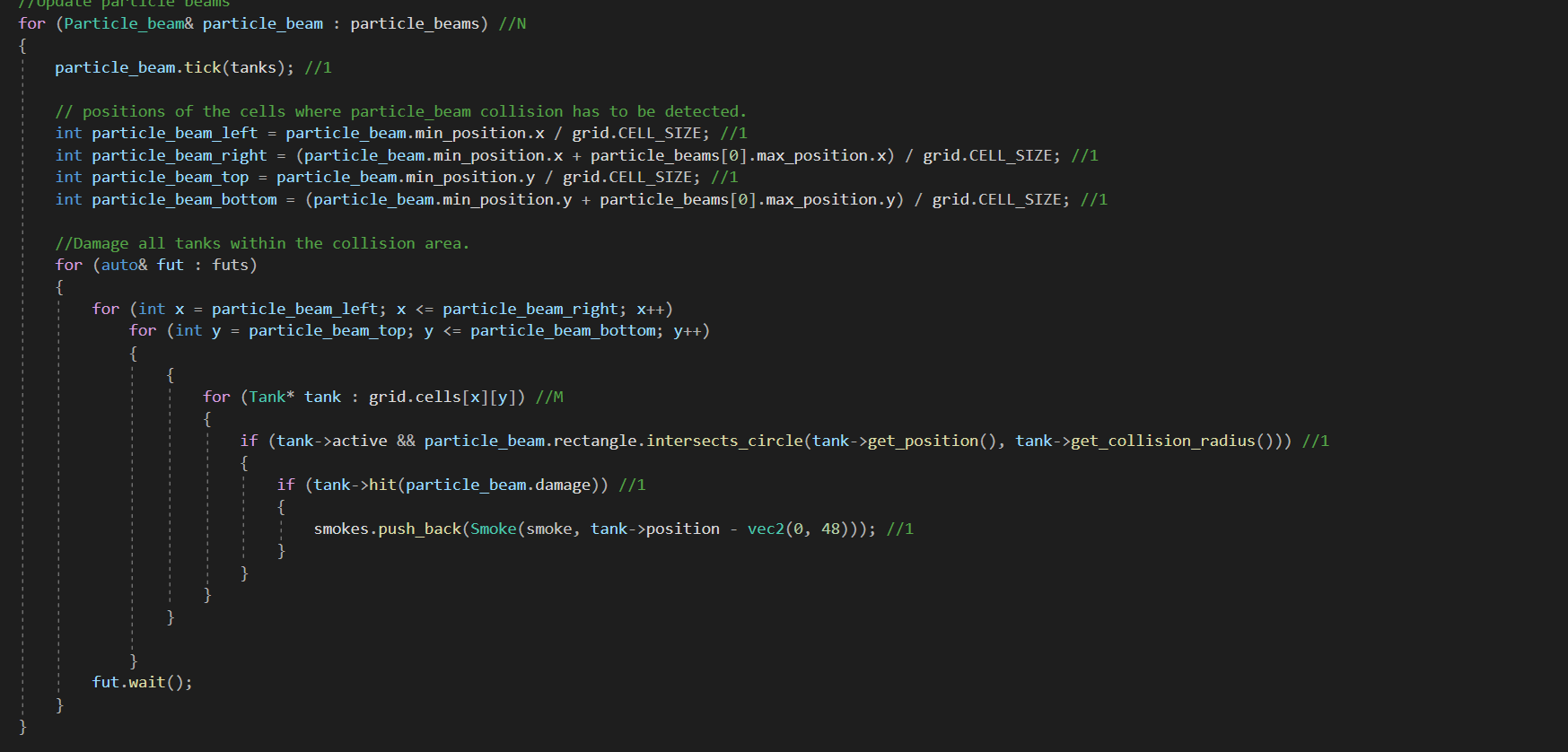


Elke update wordt tanks gesorteerd op allignement. Zo staan alle blauwe tanks in de eerste helft van de vector en de rode tanks op de tweede helft. Deze vectors worden gevuld met int waarden, bestaande uit de health waarden van de blauwe tanks en de health waarden van de rode tanks.

In de functie draw worden deze vectors gesorteerd en dit wordt gebruikt om de health bars te maken.

Hierbij is geen verandering van big O. Wel is dit efficiënter dan het sorteren en gebruiken van vectors gevuld met Tank objecten.

#### Update Particle\_Beams:



Particlebeams controleren alleen op collision op de tanks binnen de opgegeven area.

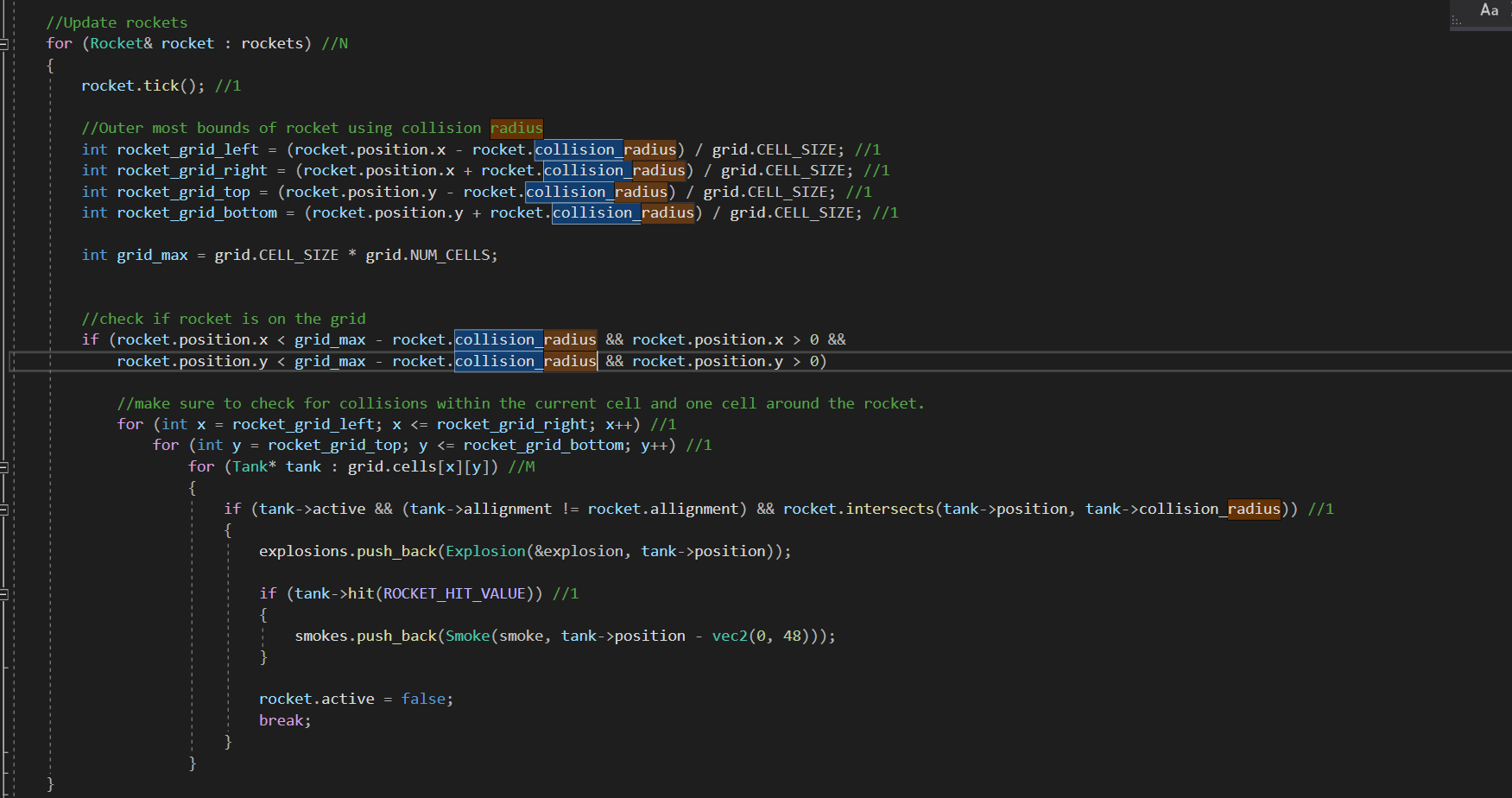
Ook hier is geen verandering van de Big O. Wel is dit efficiënter omdat er eerst gecontroleerd werd op collision voor elke tank op het veld.

#### Insertion Sort Tank Health:

Micky invullen+plaatjes met Big O Waarden <3

De insertion sort is gewijzigd tot een merge sort.

#### *Update Rockets:*



## Concurrency