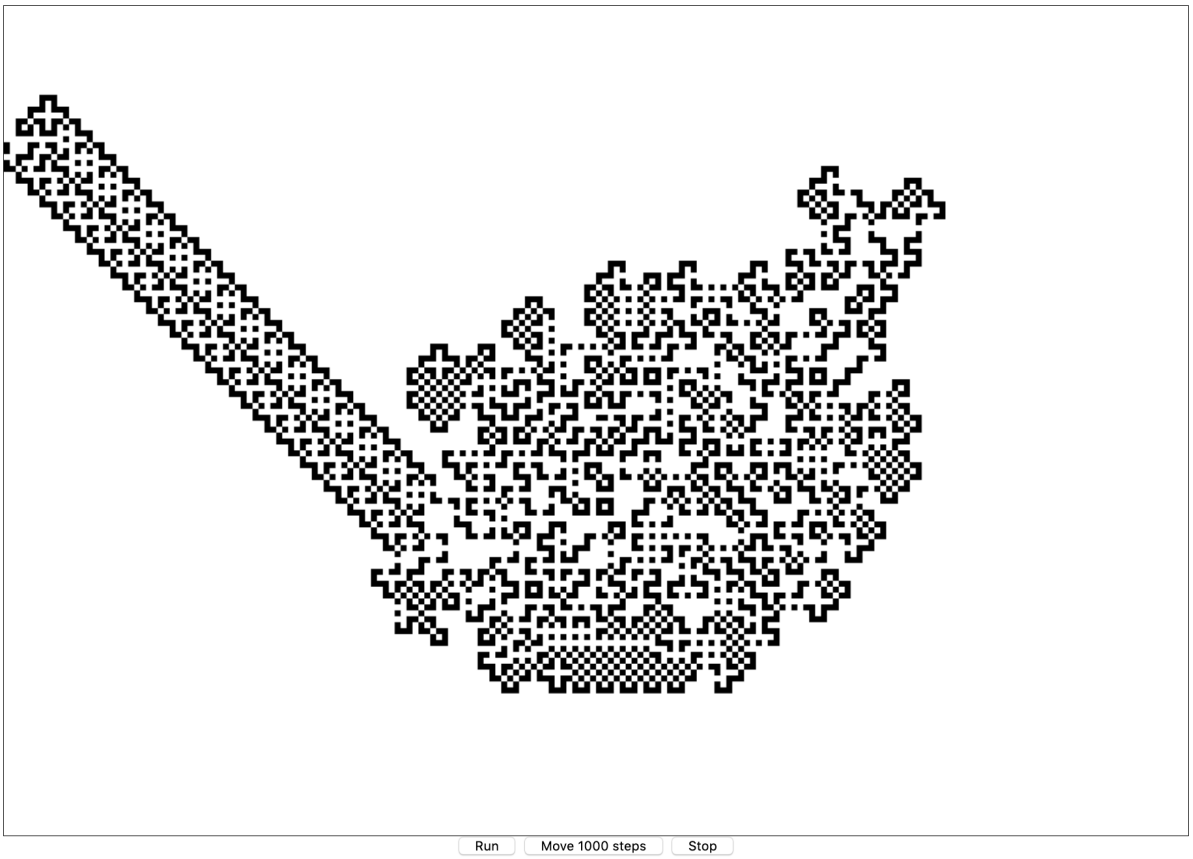
Projects

Παρακάτω σας δίνονται 5 εργασίες για την τελική εξέταση του εργαστηρίου, από τις οποίες θα επιλέξετε την μία.

Εργασία 1: Κυψελικά Αυτόματα "Turmites"



Δημιουργήσετε σελίδα HTML που θα υλοποιεί με χρήση του HTML5 Canvas ενα κυψελικό αυτόματο "Turmite" όπως φαίνεται στην εικόνα πιο πάνω. Σε αυτό το Video μπορείτε να δείτε πώς φαίνεται καθώς και τη χρήση των 3 control buttons.

Στη συγκεκριμένη εργασία θα προσομοιώσετε ενα "turmite" που έχει 2 καταστάσεις και χρησιμοποιεί 2 χρώματα ως εξής:

- Φανταστείτε ότι ο καμβάς διαιρείται σε μικρές "κυψελίδες" (cells) που είναι τετράγωνα διαστάσεων 3 επί 3 pixels ή 5 x 5 pixels. Έτσι σχηματίζεται ένα πλέγμα (grid) από γραμμές και στήλες Παράδειγμα, αν ο καμβάς έχει διαστάσεις 600 pixels πλάτος και 400 pixels ύψος, τότε μπορούμε να έχουμε 120 στήλες και 80 γραμμές που σχηματίζουν 120 * 80 = 9600 cells διάστασης 5 x 5.
- Ενα μυρμήγκι διατρέχει το πλέγμα κάνοντας ενα βήμα τη φορά και αλλάζοντας το χρώμα του

cell που βρίσκεται. Το μυρμήγκι μπορεί να βρίσκεται σε μια από τις δύο καταστάσεις "0" και "1" καθε χρονική στιγμή και μπορεί να ζωγραφίσει με 2 χρώματα π.χ. "άσπρο" / "μαύρο". Η κίνηση του γίνεται με βαση καποιους κανόνες που καθορίζουν την κατευθυνση του (αν θα στρίψει δεξιά, αριστερά, ή θα συνεχίσει ευθεία) καθώς και ποιο από τα δύο χρώματα θα χρησιμοποιήσει για να γεμίσει το τρέχον cell στο οποίο βρίσκεται. Οι κανόνες για το turmite που φαίνεται στην εικόνα πιο πάνω δίνονται στον ακόλουθο πίνακα:

		Τρέχον Χρώμα cell					
		Άσπρο			**Μαύρο**		
		Γέμισμα	**Στροφή**	**Κατάσταση**	**Γέμισμα**	**Στροφή**	**Κατάσταση**
Τρέχουσα Κατάσταση	0	Μαύρο	Δεξιά	1	Άσπρο	Αριστερά	1
	1	Μαύρο	-	0	Άσπρο	-	0

Αρχικά το μυρμήγκι βρίσκεται στο κέντρο του πλέγματος στην κατάσταση "1" με κατεύθυνση προς τα δεξιά και όλα τα cells του πλέγματος είναι με χρώμα άσπρο. Βασει του πίνακα πιο πάνω, εφόσον είναι στην κατάσταση "1" ελέγχουμε τη 2η γραμμή και αφού το χρώμα του cell είναι άσπρο θα πρέπει: 1) Να το γεμίσει με Μαυρο χρώμα, 2) να μην στρίψει δηλ. να συνεχίσει ευθεία στην τρέχουσα κατ/ση (δεξιά), και 3) να αλλάξει την κατάσταση του σε "0". Την επόμενη "χρονική στιγμή" βρίσκεται σε ένα cell ξανα με άσπρο χρώμα με κατευθυνση προς τα δεξιά αλλά η κατάσταση του ειναι "0" οπότε θα πρέπει, με βαση την 1η γραμμή του πίνακα, να γεμίσει το τρέχον κελί με Μαυρο χρώμα, να στρίψει δεξιά, και να κάνει την κατάσταση του "1". Στροφή δεξιά σημαίνει ότι εφόσον είχε κατευθυνση προς τα δεξιά, τώρα θα έχει κατεύθυνση προς τα κάτω άρα θα μετακινηθεί στο "απο κάτω" κελί. **Η διαδικασία αυτή συνεχίζεται μέχρι το "μυρμήγκι" να βγει έξω από τον καμβά**.

Τα 3 κουμπιά ελέγχου που φαίνονται στην εικόνα έχουν τις εξής λειτουργίες:

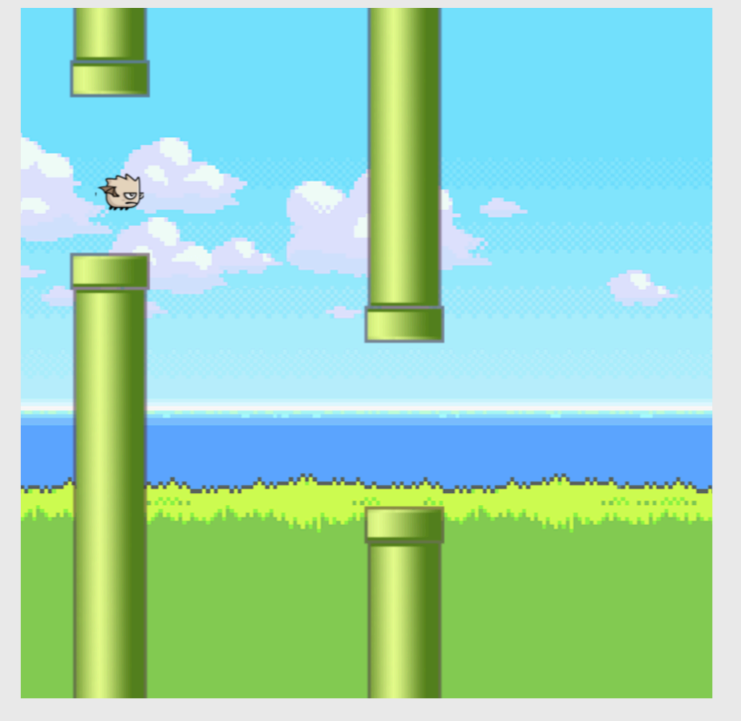
- Αρχικά ο καμβάς είναι κενός (με άσπρο χρώμα όλα τα κελλιά). Όταν ο χρήστης πατήσει το "Run" το μυρμήγκι θα ξεκινήσει από το κέντρο, αλλάζοντας τα κελλιά που επισκεπτεται από άσπρο σε μαυρο και το ανάποδο με βαση τους κανόνες του πίνακα μετάβασης που έχω πιο πάνω. **Το τρέχον κελλί του μυρμηγκιού θα φαίνεται πάντα με κόκκινο χρώμα**.
- Το κουμπί Stop θα σταματάει το animation, και αν στην συνέχεια πατηθεί το Run θα συνεχίζει από εκεί που σταμάτησε
- Το κουμπί "Move 1000 steps" θα σταματάει το animation (αν τρέχει ήδη), και στη συνέχεια θα εκτελεί 1000 φορές τους κανόνες μετάβασης δηλ. το μυρμήγκι θα κάνει 1000 βήματα με βαση τους κανόνες. Τα ενδιάμεσα βήματα δεν θα φαίνονται με animation, απλά θα εμφανίζεται η κατάσταση του καμβά μετα από τόσες κινήσεις. Στη συνέχεια θα μπορεί ο χρήστης να πατάει το Run για να συνεχίζει το animation.

Περισσότερες οδηγίες και παραδείγματα για τα "Turmites" μπορείτε να δείτε στα ακόλουθα sites:

- http://mathworld.wolfram.com/Turmite.html
- Wikipedia: https://en.wikipedia.org/wiki/Turmite
- Archived Math Games: https://web.archive.org/web/20130516052344/http://www.maa.o

rg/editorial/mathgames/mathgames_06_07_04.html

Εργασία 2: Flappy Bird!



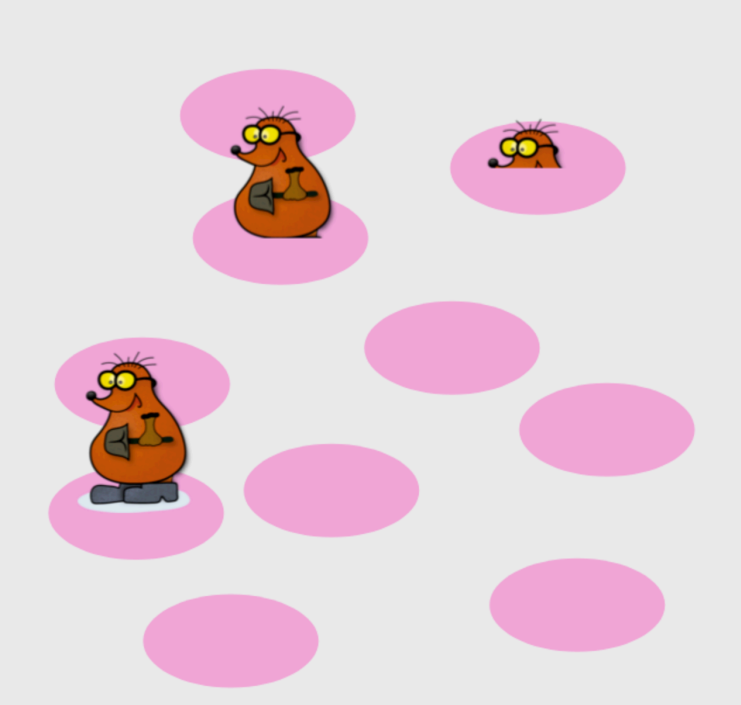
Ζητείται να υλοποιήσετε ενα παιχνίδι όπως αυτό που φαίνεται στο video εδώ. Ειναι το Flappy Bird όπου ο χρήστης με τα βελάκια πάνω-κατω του πληκτρολογίου καθοδηγεί το πουλί να περάσει ενδιάμεσα από τα κενά που αφήνουν οι σωλήνες.

Σας δίνεται ενα "spite sheet" που μπορείτε να κατεβάσετε από εδώ και έχει τη φιγούρα σε 8 διαφορετικές φάσεις. Καθε στιγμιότυπο σε αυτό το sprite sheet έχει πλάτος 49 pixels και ύψος 50 pixels. Επίσης, εδώ σας δίνεται μια εικόνα για τη σωλήνα που μπορείτε να χρησιμοποιήσετε, η οποία έχει πλάτος 81 pixels και ύψος 500 pixels.

Τα κενα θα πρέπει να κατασκευάζονται με τυχαίο τρόπο, δηλ. έχουν ένα ύψος που ποικίλλει (π.χ. είναι μεταξύ 150 και 200 pixels) και ομοίως ποικίλλει και η απόσταση μεταξύ των σωλήνων που προσδιορίζουν τα κενα.

Ως υπόβραθρο (background) έχω χρησιμοποιήσει την εικόνα που βρίσκεται στο eclass εδώ αλλά μπορείτε να βαλετε όποια άλλη θέλετε.

Εργασία 3: "Χτυπα τον τυφλοπόντικα"

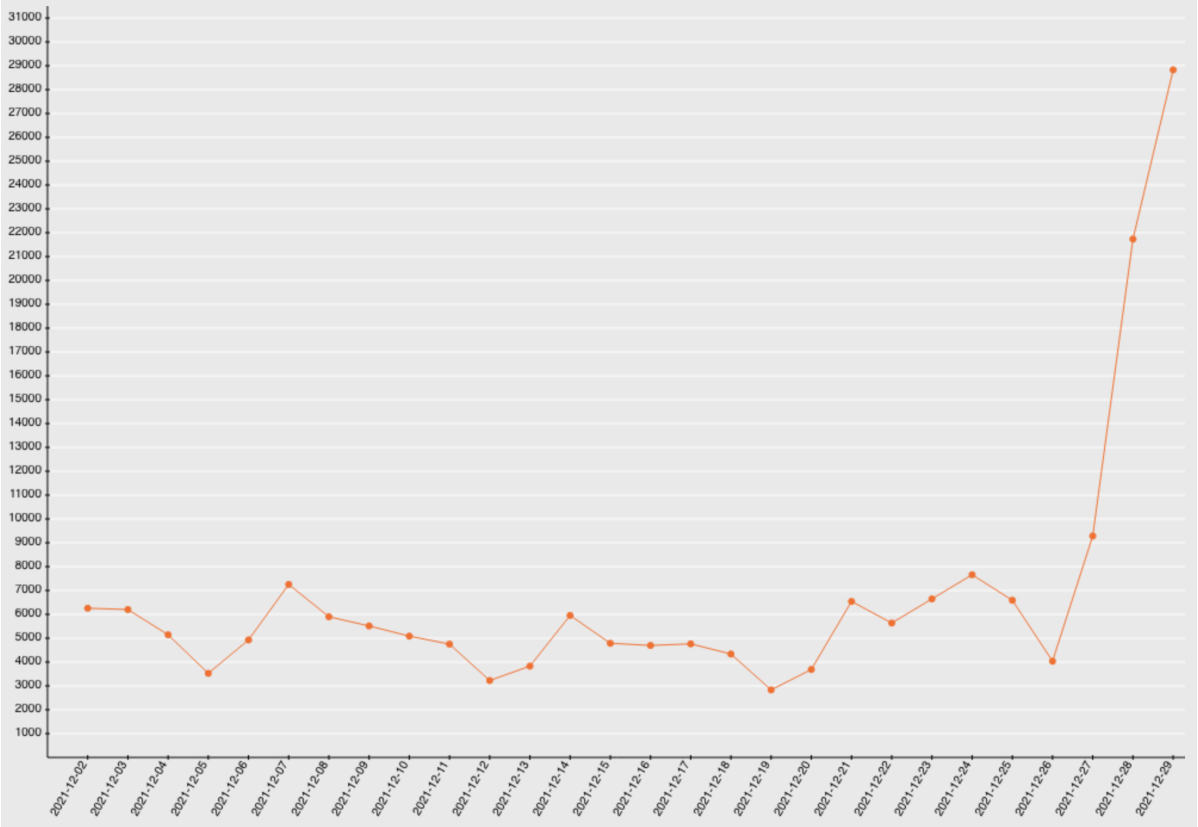


Ζητείται να κατασκευάσετε σε canvas το κλασσικό παιχνίδι "χτυπα τον τυτφλοπόντικα" (whac-a-mole) Το τελικό παιχνίδι πρέπει να ειναι όπως φαινεται εδώ.

Οταν φορτώνεται η σελίδα επιλέγονται με τυχαίο τρόπο οι θέσεις που θα βρίσκονται οι τρυπες απο τις οποίες βγαίνουν οι τυφλοπόντικες. Καθε περίπου ενα δευτερόλεπτο θα επιλέγεται μια "κενή" τρυπα και απο εκεί θα ξεκινά να εμφανίζεται ενας τυφλοπόντικας. Οι τυφλοπόντικες εμφανίζονται στην πλήρη τους μορφή για 1 δευτερόλεπτο πριν να ξεκινήσουν τη κάθοδο τους μεσα στην τρύπα. Ο σκοπός του χρήστη ειναι να προλάβει να κανει κλικ πανω σε εναν τυφλοπόντικα πριν εξαφανιστεί στην τρύπα του, και το παιχνίδι μετράει πόσους τυφλοπόντικες έχει πετυχει ο χρήστης.

Για την εικόνα του τυφλοπόντικα μπορείτε να χρησιμοποιήσετε αυτήν εδώ που έχει διαστάσεις 100 pixels πλάτος και 138 pixels ύψος. Για τις τρύπες μπορείτε να τις ζωγραφίσετε στο canvas, ή να βρειτε σχετικές εικόνες πχ. στο opengameart.

Εργασία 4: Γραφική παρασταση εξέλιξης κρουσμάτων Κορωνοϊού



Θελουμε να κανουμε μια γραφική παράσταση όπως φαινεται πιο πάνω που να δειχνει τον αριθμό των κρουσμάτων του νεου κορωνοϊού (COVID-19) ανα ημερομηνία όπως φαινεται στην εικόνα παραπάνω. Πρόκειται για ενα "line chart" (ή line plot) μιας και τα γειτονικά σημεία στο διάγραμμα ενώνονται με γραμμες για να φανει πιο καθαρα η εξέλιξη (τάση) των κρουσμάτων. Ο οριζόντιος άξονας ("των x") εχει τις ημερομηνίες και ο κάθετος ("των y") αντιστοιχεί στον αριθμό των κρουσμάτων.

Θεωρείστε οτι έχετε τα δεδομένα ενός μήνα στον κώδικα σας σε ενα Javascript array όπως ειναι το παρακάτω. Καθε στοιχείο του array ειναι ενα άλλο array με 2 στοιχεία: το 1ο ειναι η ημερομηνία και το 2ο ειναι ο αριθμός κρουσμάτων εκείνης της ημερομηνίας. Θεωρείστε ακόμα οτι τα δεδομένα ειναι ταξινομημένα βασει ημερομηνίας σε αύξουσα σειρα:

```
let data = [
    ['2021-12-01', 6192],
    ["2021-12-02", 6256],
    ["2021-12-03", 6201],
    ["2021-12-04", 5143],
    ["2021-12-05", 3526],
```

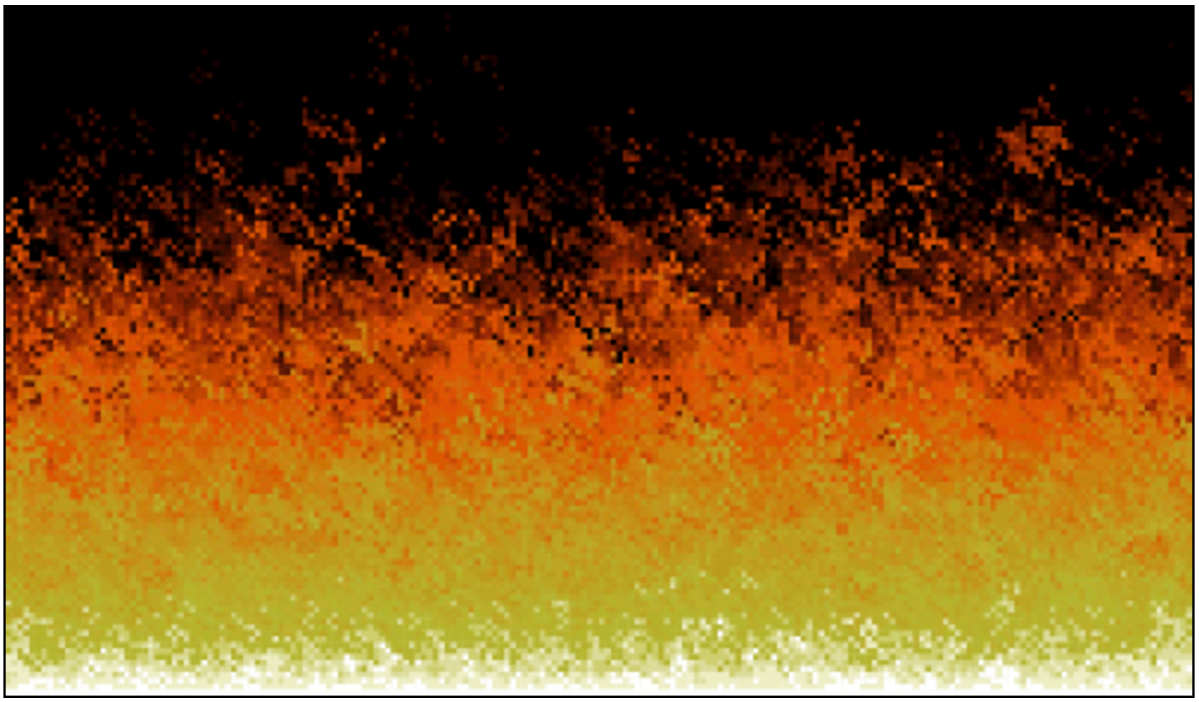
```
    ["2021-12-06", 4927],
    ["2021-12-07", 7254],
    ["2021-12-08", 5899],
    ["2021-12-09", 5513],
    ["2021-12-10", 5087],
    ["2021-12-11", 4751],
    ["2021-12-12", 3225],
    ["2021-12-13", 3829],
    ["2021-12-14", 5953],
    ["2021-12-15", 4786],
    ["2021-12-16", 4696],
    ["2021-12-17", 4761],
    ["2021-12-18", 4337],
    ["2021-12-19", 2831],
    ["2021-12-20", 3689],
    ["2021-12-21", 6542],
    ["2021-12-22", 5635],
    ["2021-12-23", 6647],
    ["2021-12-24", 7660],
    ["2021-12-25", 6590],
    ["2021-12-26", 4036],
    ["2021-12-27", 9284],
    ["2021-12-28", 21732],
    ["2021-12-29", 28828]
];
```

Υποδείξεις:

- Στον άξονα y εχουμε τιμές ανα 1000 "κρούσματα" και μπορείτε πχ να το υλοποιήσετε έτσι ώστε να έχουν 50 pixels απόσταση η μια τιμή απο την άλλη. Αυτό σημαίνει οτι αυξηση του y κατα 1 pixel θα αντιστοιχεί σε 20 "κρούσματα" και επομενως, για παράδειγμα, η τιμή 6192 θα βρισκεται σε απόσταση $6192/20 = 309.6$ pixels απο την αρχη των αξόνων του γραφήματος.
- Κατω απο τον άξονα των x φαινονται οι ημερομηνίες σε ίση απόσταση μεταξύ τους. Θεωρείστε ότι τα δεδομένα μας είναι για εναν μήνα, άρα θα πρέπει να "χωράνε" μεχρι 31 ημερομηνίες. Επίσης θα πρέπει να υπάρχει ενα περιθώριο 60 - 70 pixels μεταξυ του οριζόντιου άξονα και της κάτω πλευράς του καμβα για να φαινονται σε κείμενο οι ημερομηνίες. Το ίδιο περιθώριο ομοίως πρέπει να υπάρχει αριστερα απο τον άξονα y για να φαίνονται (σε χιλιάδες) ενδεικτικά τα κρούσματα.
- Μπορείτε να χρησιμοποιήσετε ό,τι χρωμα θελετε. Στην εικόνα που εχω πιο πανω, εχουμε το `#ececec` ως χρώμα φόντου για όλο το γραφημα και το `#ef7538` για τις γραμμές.

Εργασία 5: Doom Fire!

Σε αυτή την εργασία θα υλοποιήσετε σε canvas μια "προσομοίωση" "φωτιάς" όπως φαίνεται στην ακόλουθη εικόνα.



Πρόκειται για ενα εφε που χρησιμοποιηθηκε στην εκδοση του παιχνιδιού Doom για το Playstation 1. Μπορείτε να δειτε το αποτέλεσμα που επιθυμούμε στο ακόλουθο site: https://doom-fire.com/

Σε αυτή την εργασία θα υλοποιήσετε το Doom fire effect με τη χρήση του HTML5 canvas. Η υλοποίηση σας θα βασιστεί στις εκτενείς οδηγίες που υπάρχουν στην ιστοσελίδα: https://fabiensanglard.net/doom_fire_psx/

Παρακάτω σας δινονται τα 36 χρώματα-στάδια της φωτιάς που υπάρχουν και στο παραπάνω site και μπορείτε να ενσωματώσετε στον κώδικα σας:

```
    const colors = [
            "#070707",
            "#1f0707",
            "#2f0f07",
            "#470f07",
            "#571707",
            "#671f07",
            "#771f07",
            "#8f2707",
            "#9f2f07",
```

```
            "#af3f07",
            "#bf4707",
            "#c74707",
            "#DF4F07",
            "#DF5707",
            "#DF5707",
            "#D75F07",
            "#D7670F",
            "#cf6f0f",
            "#cf770f",
            "#cf7f0f",
            "#CF8717",
            "#C78717",
            "#C78F17",
            "#C7971F",
            "#BF9F1F",
            "#BF9F1F",
            "#BFA727",
            "#BFA727",
            "#BFAF2F",
            "#B7AF2F",
            "#B7B72F",
            "#B7B737",
            "#CFCF6F",
            "#DFDF9F",
            "#EFEFC7",
            "#FFFFFF",
        ]
```