Asteroid Detection

International Coding Contest
27th April 2018

event organizer

Catalysts

proudly sponsored by

DEUTSCHE BÖRSE GROUP

CODING CONTEST.org

> Intro

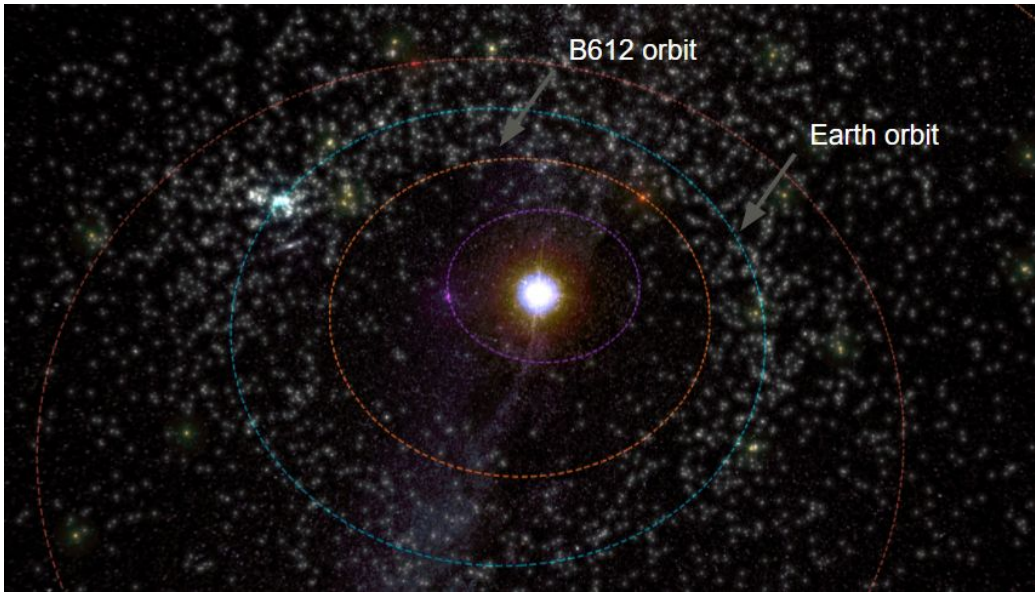
February 15th 2013
Tscheljabinks / Sibiria 3:20 UTC

The largest meteor in 100 years exploded some kilometers above the ground and injured 1.500 people.

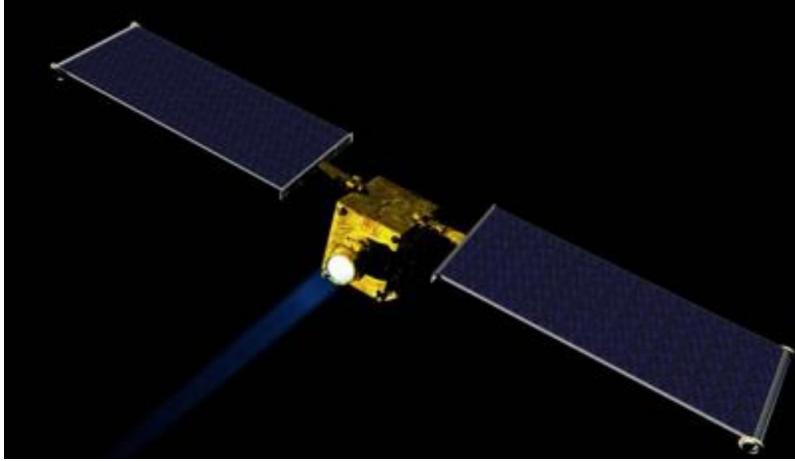
The asteroid came straight from the sun. Therefore it was undetectable for all operational asteroid surveillance missions which monitor more than 700.000 asteroids orbiting mainly beyond Earth in direction to Mars and Jupiter.

> October 2020 nearby Venus Orbit ...

B612 - A NASA and ESA funded mission - enters an orbit close to the sun and monitors the space between Venus and Earth with its infrared sensors. Till 2030 it is expected to detect more than a million additional asteroids close to the Earth's orbit



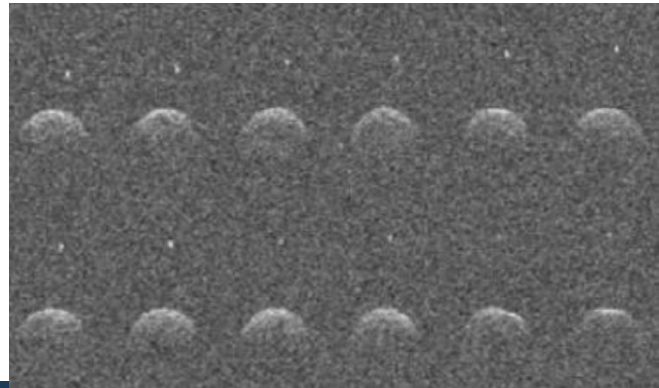
April 2031 Earth



NASA and **ESA** hands out all **B612** mission data to you.

You should run a full mission reprocessing with the goal of counting and classifying all asteroids you find in the data in order to:

- Identify threats to life on Earth and enable preventive deflective missions
- Use them for asteroid mining



Level 1

> Level 1

The **B612** sensor is continuously taking images.

Task for Level 1:

Capture the timestamps of asteroid occurrences and output them in order.

> Level 1

If not stated otherwise you may assume the following throughout all levels:

- You are given time-stamped images. A timestamp is just a positive integer denoting a specific moment in time.
- An image is a 2D array, containing pixels with intensity values (positive or zero).
- Connected regions of one or more pixels with positive intensity values indicate the presence of an asteroid.
- An image shows at most one asteroid.
- There are no asteroids in occultation. That is, there will never be two asteroids in the same line of sight, in the same image.

> Level 1

Input format. You will receive a file that contains:

```
› input ::= start end imagecount NL image*
› image ::= timestamp rowcount colcount NL row*
› row ::= intensity* NL
```

(`NL` is newline, `element*` are repeated instances of `element`.)

Timestamps are unique and all within the observation period.

The images are ordered by `timestamp`.

Example: Two images, first one with timestamp 100 and size 3 by 3, the second one with timestamp 200 and size 2 by 3:

```
10 500 2
100 3 3
622 593 231
0 442 0
0 0 0
200 2 3
0 0 0
0 0 0
```

Input format: The meaning of the input parameters:

name	type	description
`start`	integer	start timestamp of observation period
`end`	integer	end timestamp of observation period
`imagecount`	integer	number of following `images`
`timestamp`	integer	timestamp of the `image`
`rowcount`	integer	number of following image `rows`
`colcount`	integer	number of `intensity` values in each `row`
`intensity`	integer	pixel `intensity` (zero or positive)
`NL`		new-line

> Level 1

Output format. You must upload a file that contains:

- `output ::= resultline*`
- `resultline ::= timestamp NL`

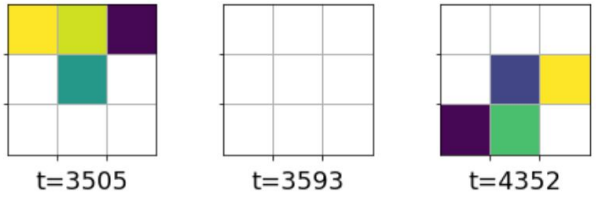
The `resultlines` must be ordered by `timestamp`.

name	type	description
`resultline`		describes the occurrence of an asteriod
`timestamp`	integer	timestamp of the occurrence
NL		new-line

> Level 1

Sample input:

```
1000 9999 3
3505 3 3
622 593 231
0 442 0
0 0 0
3593 3 3
0 0 0
0 0 0
0 0 0
4352 3 3
0 0 0
0 298 708
191 557 0
```



Sample output:

```
3505
4352
```