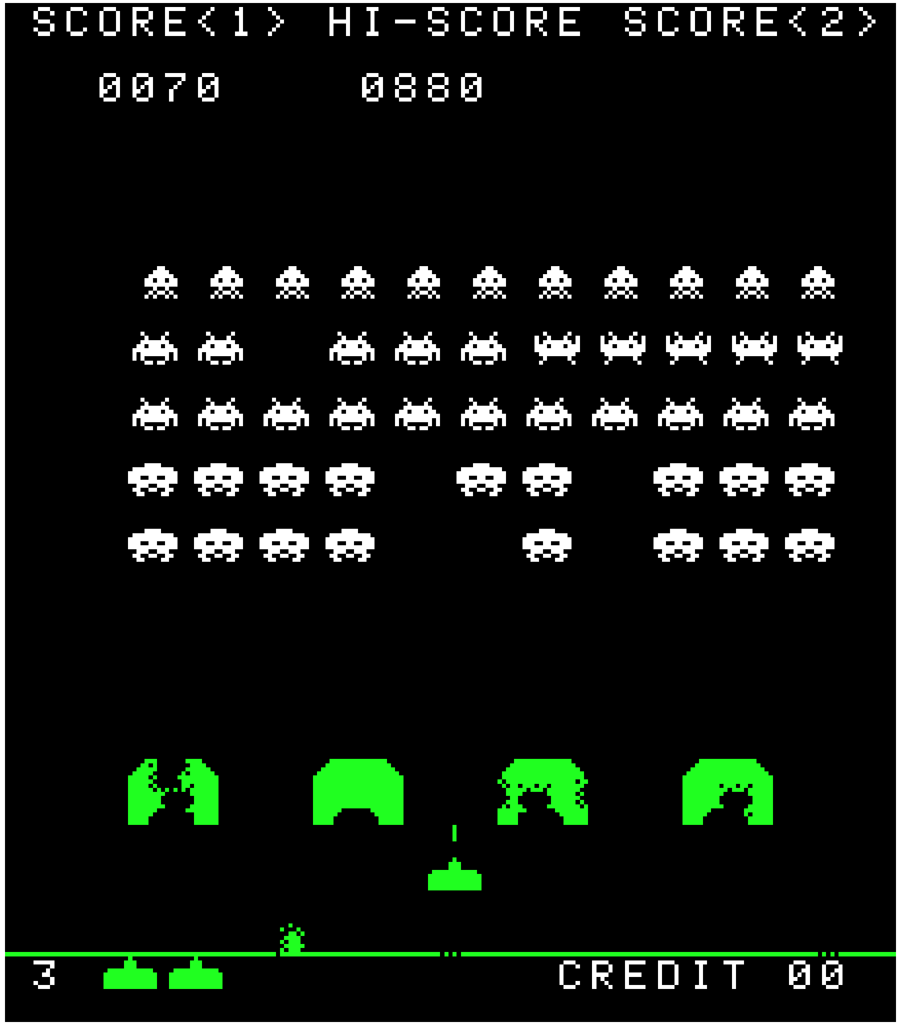
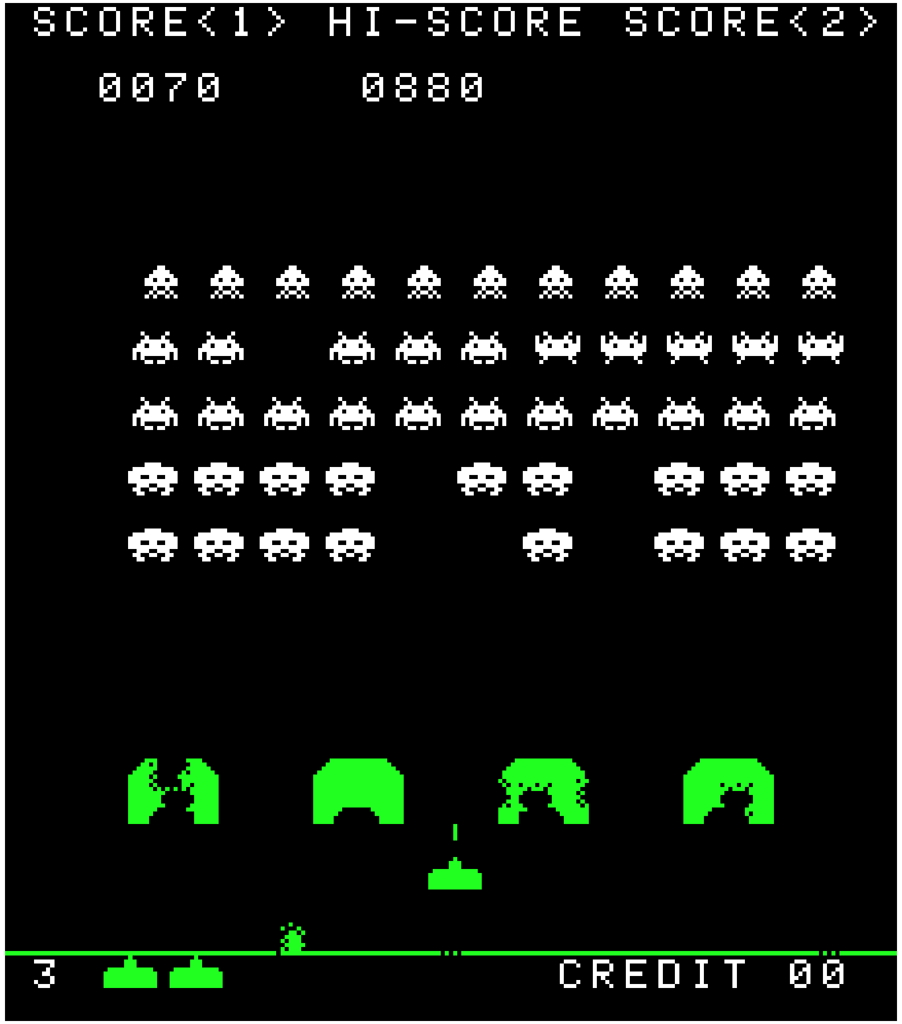
SPACE INVADERS

What makes Space Invaders unique?



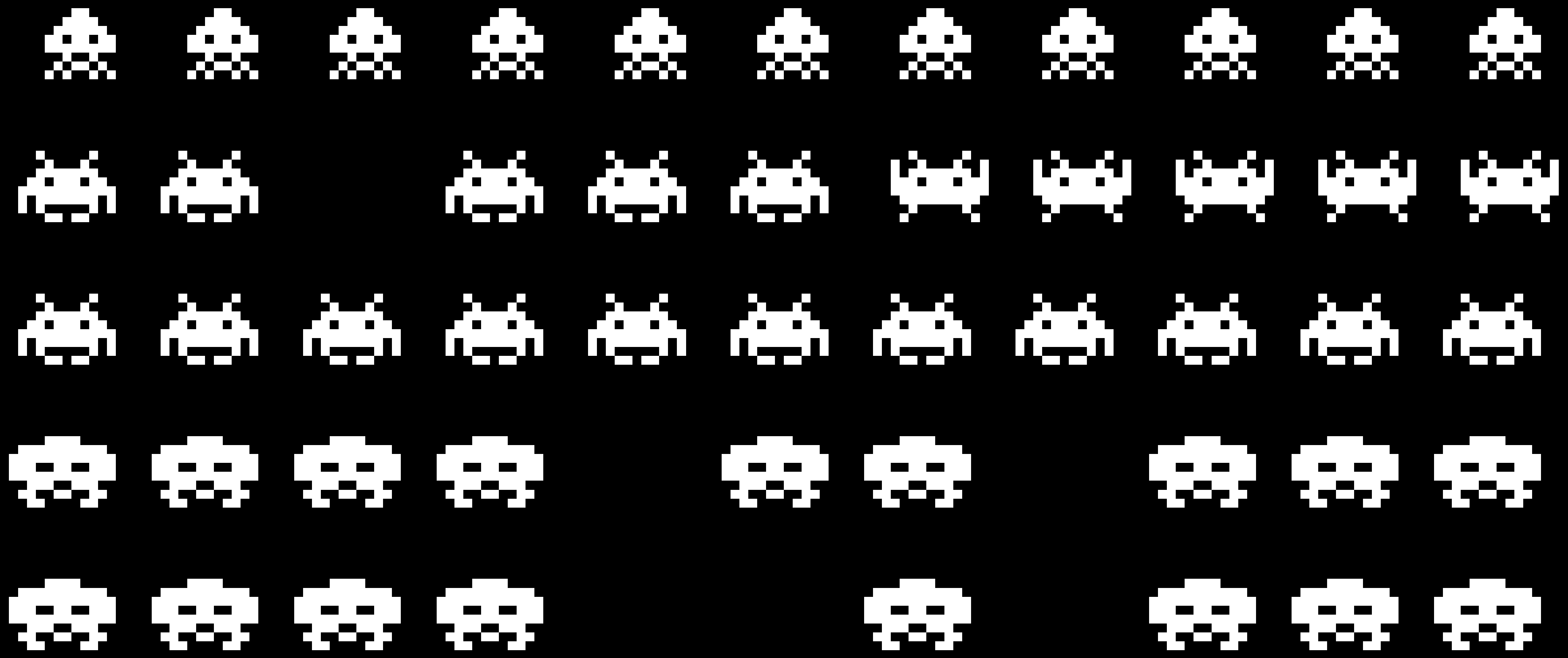
Key characteristics (requirements)

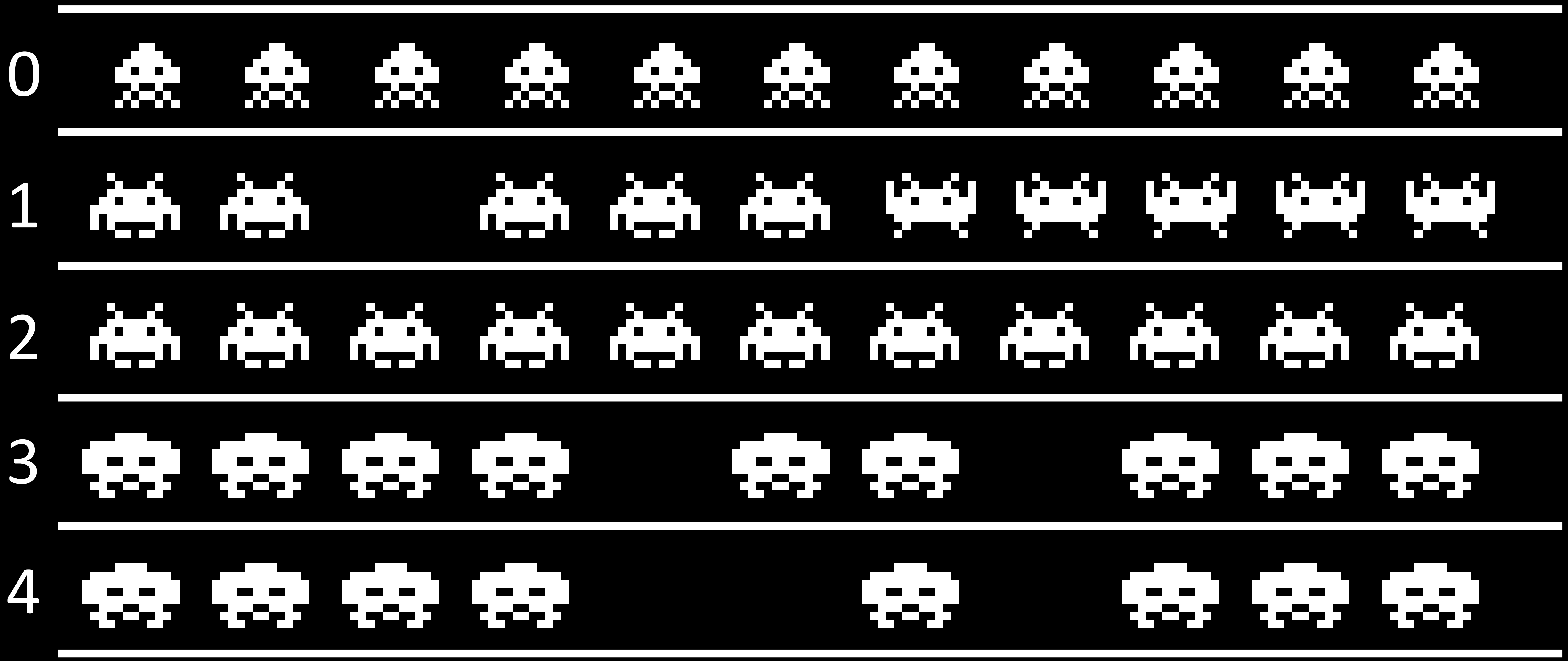
- Main shooter moves left and right
- The array of invaders that move together row by row when the left-most or right-most invader reaches the edge
- The invaders array speed of movement also increases
- The barriers that crumble as they are shot
- UFO boss that appears in intervals



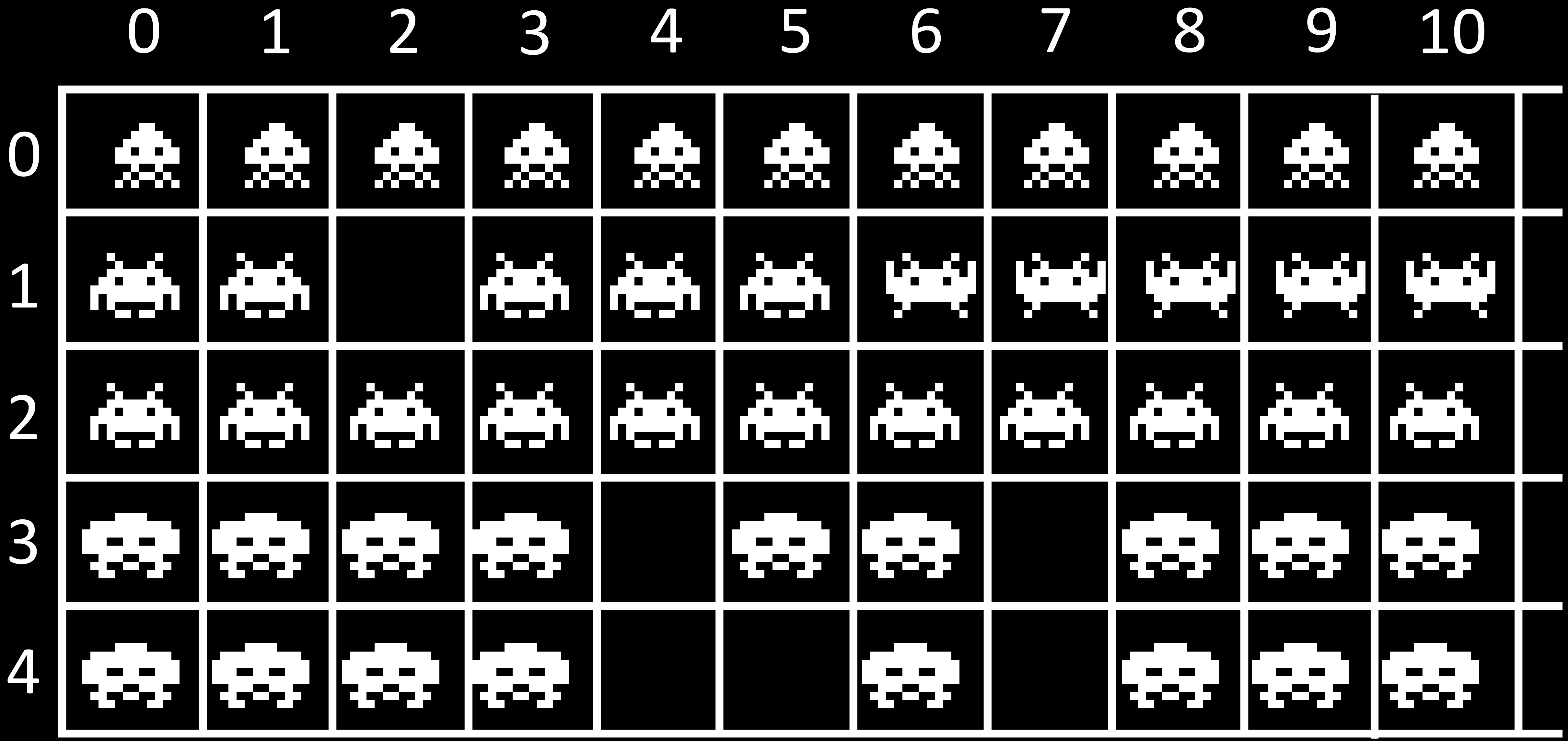
How do we design a solution to this?

- Main shooter moves left and right
  - Move coordinates (x) when respond to left and right key press
- The array of invaders
  - 2D array of graphics?
- How do we maintain the left and right-most invaders?
  - An x coordinate to keep a track of this?
- Edge detection – both for shooting and left/right edge
  - Compare x/y coordinates?
- Increasing speed of movement – variable?
  - The movement of x/y coordinates is multiplied by a speed variable?





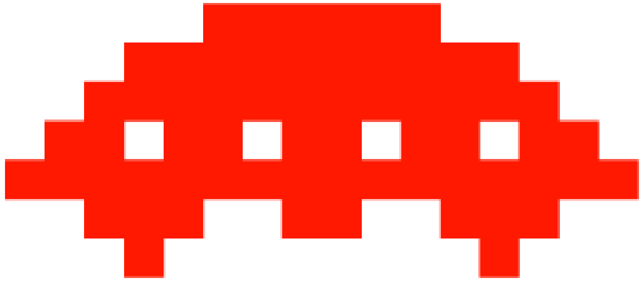
0
1
2
3
4



0
1
2
3
4
5
6
7
8
9
10
0
1
2
3
4

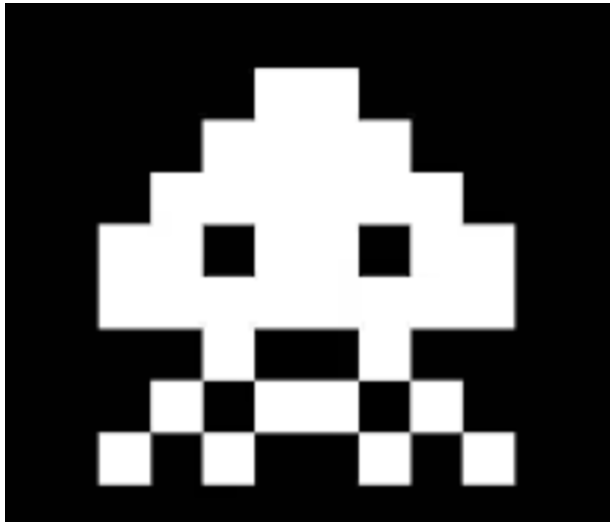
What about OOP?

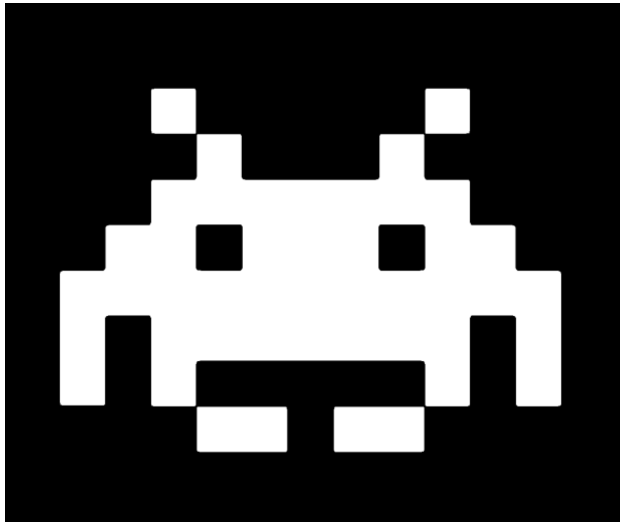
- Should we build some of these entities as classes? Is it worth it if they only have one object – like the main shooter?



At least three specialised types of invader

Do they share anything in common?
Could we use a well-known OOP concept here?

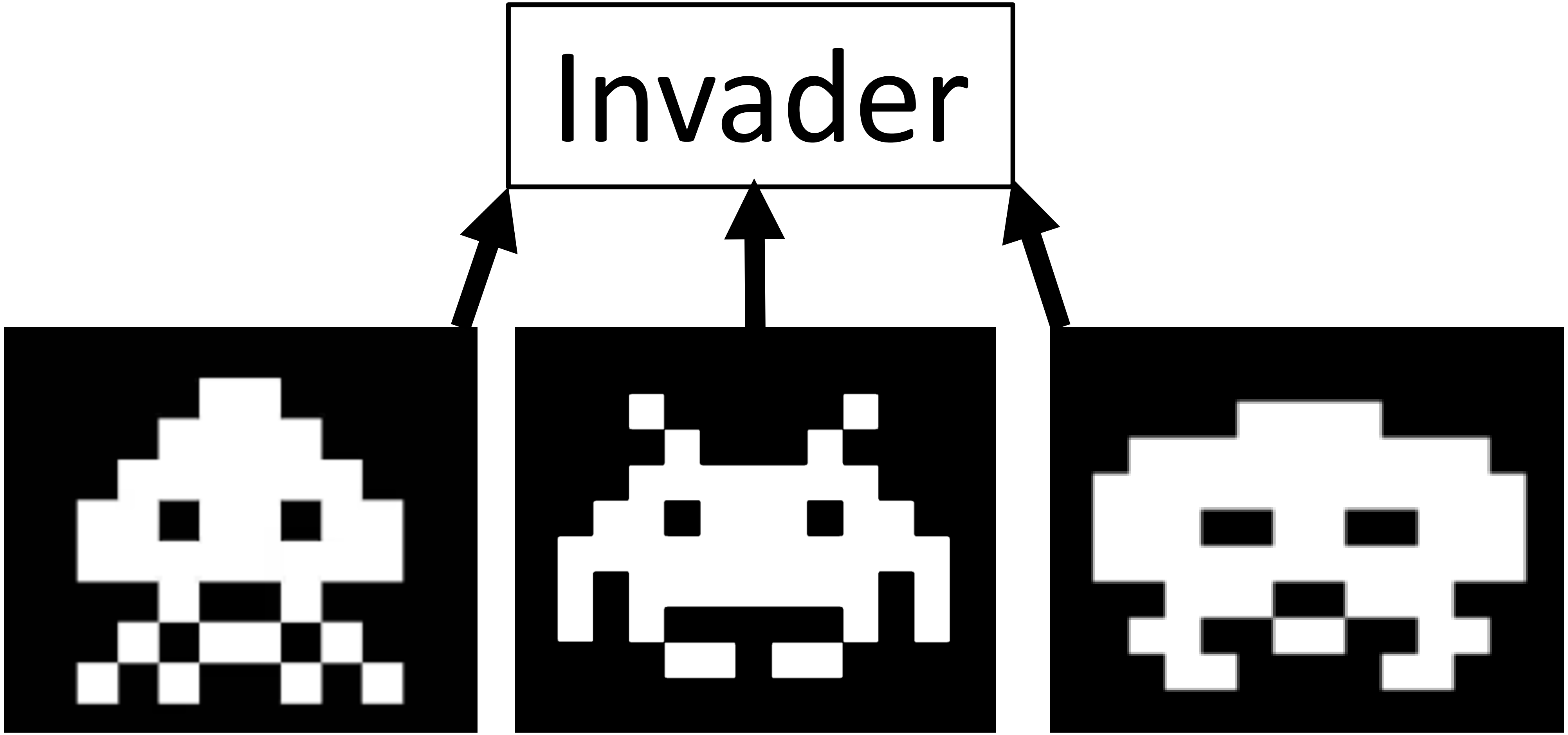




Inheritance?

Parent:

Children:



Invader

Possible plan:

- 1.0 Player basics
  - 1.1 Draw on the screen
  - 1.2 Move left and right by keypress
  - 1.3 Shoot
- 2.0 Invader basics
  - 2.1 Draw one invader on screen
  - 2.2 Draw a row of invaders
  - 2.3 Draw alternating rows of invaders
  - 2.4 Move across the screen left then down, then right and down
- 3.0 Edge detection
  - 3.1 When player shoots invader (and vice-versa)
  - 3.2 When invaders collide with left and right edge of screen

Implementation

Summary

- Establish key requirements – what makes Space Invaders unique?
- THINK about how possible solutions (design) BEFORE coding
- When you implement – build in stages – you could focus on building and testing one requirement/key feature at a time.