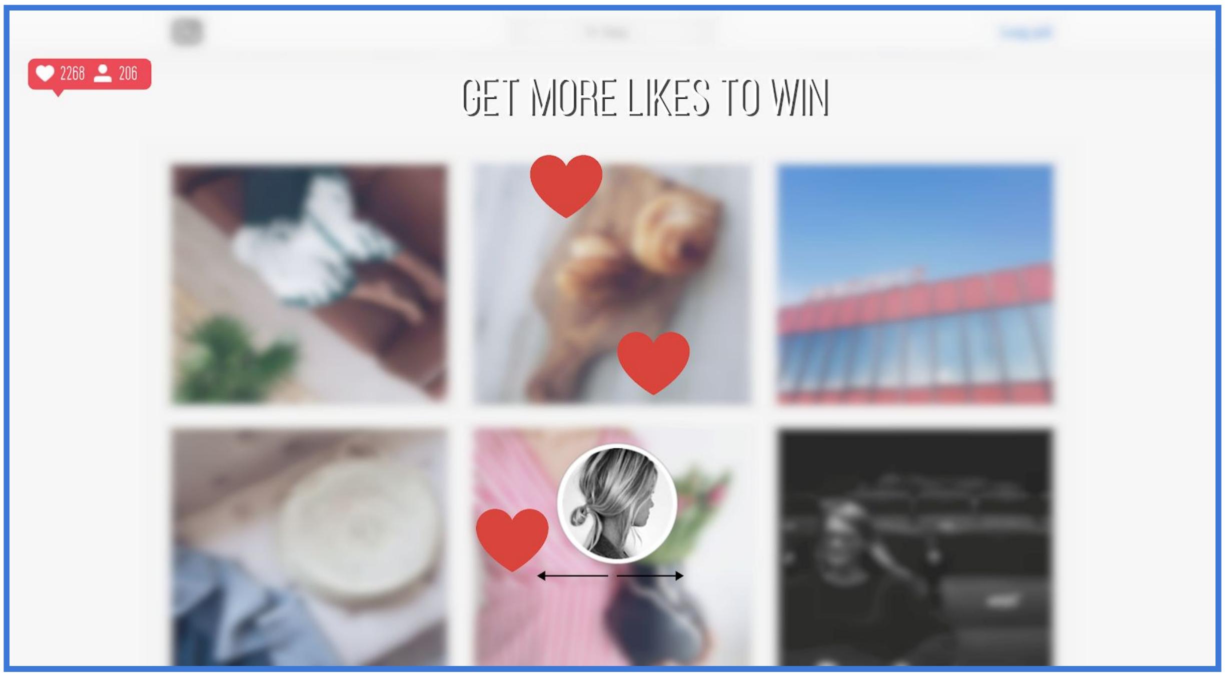
LIKE HUNTER

Link to a video of our program: [click here]



In relation to the course Aesthetic Programming, we have created a program named "Like Hunter", using p5.js. Our overall idea is to create something that relates to social media, as we find that the different social media platforms play an increasingly bigger role in all of our lives and the society in general. Because of the huge impact social media has on our culture and society, the issue calls for a great deal of attention and reflection – this, we want to express through the program "Like Hunter".

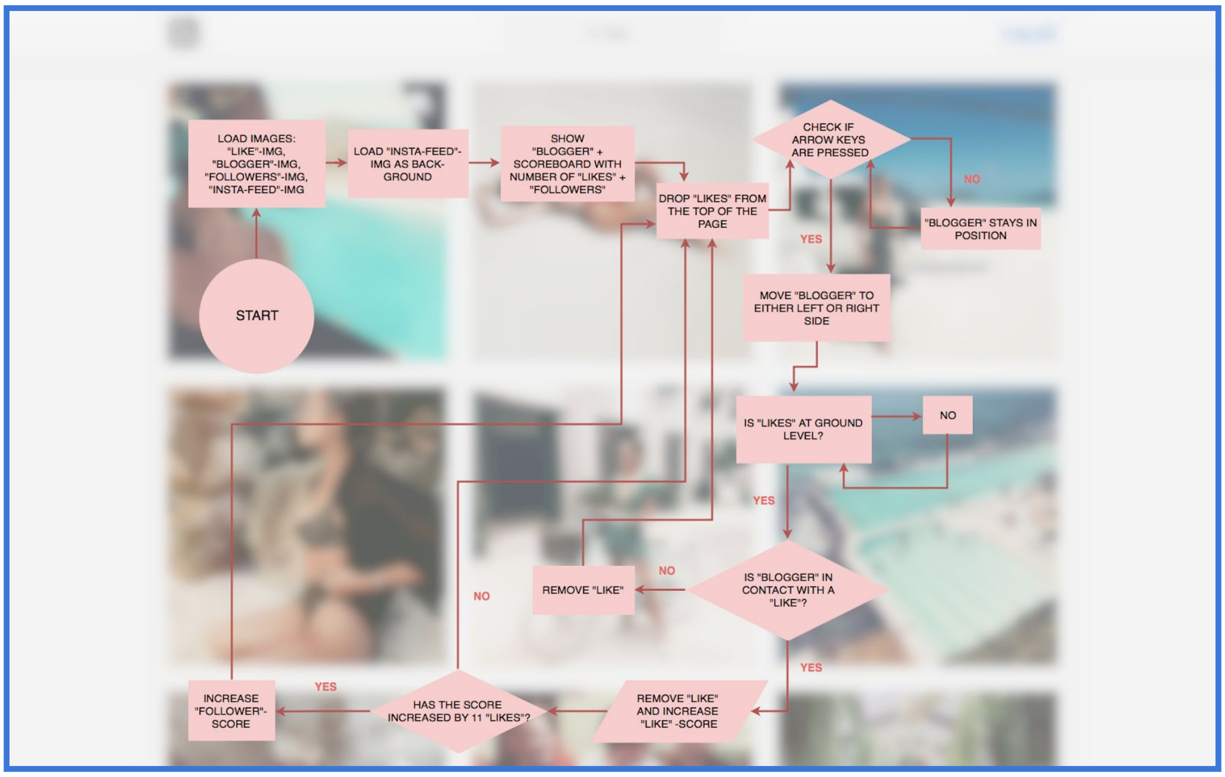
The program is a game based on the social media platform Instagram, where the ability to "like" and "follow" dominates the overall use of the software. Since the emergence of Instagram, several tendencies have arisen in the use of the platform, such as striving for likes and followers, also known as the concepts of "like4like" and "follow4follow". These concepts, especially "like4like", has inspired the term "Like Hunter". The definition of this term relates to a user on social media whose main goal is to gain as many likes as possible, sometimes regardless of the content.

Instagram is a social media platform where you can express yourself creatively by posting and editing pictures from your everyday life. However, during the past few years, Instagram has grown to become one of the most used social media platforms (Kulturstyrelsen, 2015 and Tassy, 2018), where many users have established a career through Instagram. These users are often bloggers or influencers, who has gained a large amount of followers, which makes them attractive to advertisers, who seek to promote various products to a large amount of people. Through these sponsorships, the bloggers or influencers have the ability to build a career off of followers and likes, using Instagram.

The transition from more private users to a more capitalistic approach is related to the text “What Do Metrics Want? How Quantification Prescribes Social Interaction on Facebook” by Benjamin Grosser, where the quality of the likes is not nearly as important as the quantity. This quantification has an increasing effect on people's self-esteem and self-worth. As Benjamin Grosser writes: *“If our numbers are rising, our desire is met; if not, it remains unmet. Personal worth becomes synonymous with quantity.”* (Grosser, 2014) We are not as concerned with *who* likes the posts, but rather *how many*. We are in this way, as users of Instagram or other social media platforms, reduced to simple numbers.

This reduction is one of the problems we will try to address in the game. Another aspect of this issue, is the way opinions are not individual or personal but are instead put into (often binary) boxes created by the social media platform. This relates to the dominant programming methodology which is the object-oriented paradigm. The theme of object orientation in particular highlights the connection between the technicality and the conceptual, hereby expressing the aesthetics of code.

How "Like Hunter" works


START
LOAD IMAGES: "LIKE"-IMG, "BLOGGER"-IMG, "FOLLOWERS"-IMG, "INSTA-FEED"-IMG
LOAD "INSTA-FEED"-IMG AS BACK-GROUND
SHOW "BLOGGER" + SCOREBOARD WITH NUMBER OF "LIKES" + "FOLLOWERS"
DROP "LIKES" FROM THE TOP OF THE PAGE
CHECK IF ARROW KEYS ARE PRESSED
NO
"BLOGGER" STAYS IN POSITION
YES
MOVE "BLOGGER" TO EITHER LEFT OR RIGHT SIDE
IS "LIKES" AT GROUND LEVEL?
NO
YES
IS "BLOGGER" IN CONTACT WITH A "LIKE"?
NO
REMOVE "LIKE"
NO
YES
REMOVE "LIKE" AND INCREASE "LIKE" -SCORE
HAS THE SCORE INCREASED BY 11 "LIKES"?
YES
INCREASE "FOLLOWER"-SCORE

Our program is a game called "Like Hunter". The player is displayed as a round picture of a stereotypical blogger from Instagram. "Likes" in the shape of hearts (inspired by Instagram) will appear from the top of the screen and "fall" to the bottom of the page. The aim for the player is to move the blogger from side to side by pressing the arrow keys and catch the likes by getting the blogger to be in contact with the likes when they reach the bottom of the page. At the top of the page there is a counter displayed as a scoreboard, which increases in both likes and followers when the likes are caught. The visuals are again in the style of Instagram in terms of colorschemes and shapes. Throughout the game the player will be encouraged by text appearing at the top of the screen to keep hunting for likes to win, even though the game is never-ending. Therefore you will keep on hunting for likes, which is equivalent to the tendency seen on social media.

Technical explanation and Object-Oriented Programming

In order to get a deeper understanding of the critical aspects of our program, we can take a closer look at how the program addresses the topic of object orientation. This is both interesting and relevant on a technical plan as well as a more conceptual plan. Object oriented programming was introduced when the structured programming methodology that had been the basic programming methodology up until the 1970s, seemed to have a fundamental flaw in the form of oversimplification. Where the structured paradigm had separated operations and data, the object-oriented paradigm combined these two fundamentally related halves by having objects that are designed to both be able to manage data and simultaneously carry out operations. (Lee, 2013, p. 17)

Putting an object in a program requires that you have a class that contains the attributes of the object – the object is constructed from the class, and the class is in this way the object's blueprint. Apart from the different attributes and methods, the class commonly consists of a constructor and a destructor function. The destructor function is quite self-explanatorily the action of deleting a given object, and the constructor function is the action of physically placing the object into the computer's memory. In the constructor, it hereby initializes the attributes and methods of the class. Many objects can be created from one class, which means that an object is an instance of a certain class, and therefore an object can not be conceived without a predefined class. The same class can also construct different objects, depending on which methods are being used.

In our particular program, we use object oriented programming, for instance when we call the class of hearts in our code. The class consists of these red hearts or "like buttons", copied from the interface of the social media platform Instagram. The class is one of two classes in our program, and it is called "Heart" - it has the methods "show", "move" and "touched". The "Heart" class' constructor has predefined values, since we needed a lot of hearts with the same look and abilities. Therefore the class' constructor looks as the following:

```
 constructor() {
   this.positionX = random(width);
   this.positionY = 0;
```

```
    this.xSpeed = 0;
    this.ySpeed = 3;
    this.w = 100;
    this.h = 100;
    this.vis = true;
}
```

A method was needed in order to make the blogger catch the hearts falling down. Therefore we created a method in the "Heart" class, that enabled the blogger and the heart to interact with one another. The method is called "touched" and looks as the following:

```
  touched() {
  let d = dist(this.positionX-50,this.positionY-50,grammer1.x,grammer1.y);
if (d<120) {
this.vis=false
likes++;
follow=floor(likes/11)
}
}
}
```

The syntax "dist" is used, in order to make the program calculate the distance between the two objects, the blogger and the heart. We created a variable called "d", that has the same value as the distance between the blogger and the heart. If this distance is less than 120, the variable "this.vis" changes to false, which makes the heart disappear.

Meanwhile, when the variable “d” is less than 120, the amount of likes in the top left corner is increasing. Furthermore, the number of followers in the top left corner is increasing, every time the number of likes increases by more than 11. The use of object oriented programming in our code, helped us create this game, where two objects were able to interact with one another.

There is also other practical affordances in using object oriented programming, since using classes saves you from having to write a lot of the same code, to create different objects. For example when creating rectangles in different sizes, shapes, colors or placement. This can all be done more easily when using classes.

Object orientation and the real world

Object orientation is a widespread approach, and the use of objects allow software engineers to create models of the real world that had been previously thought impossible (Lee, 2013, p. 18) – in regard to this, a lot of ethical questions can be raised in relation to how real-world situations are translated to artificial objects in computer programming. However, a problem arises as real-world situations and human action that is in itself situated and ambiguous cannot be mirrored in programming. When using objects in software, there is a need to do an abstraction. Cecilie Crutzen and Erna Kotkamps quotes Hoare in the article “Object Orientation” from the lexicon “Software studies” edited by Matthew Fuller:

> *“Hoare suggests: “Abstraction arises from recognition of similarities between certain objects, situations or processes in the real world and the decision to concentrate upon these similarities and to ignore for the time being the differences.””* (Crutzen & Kotkamp, 2008, p. 203).

They argue that abstractions are simplified descriptions with a limited number of accepted properties, and as a result of this, a lot of other aspects of the world will have to be suppressed.

Returning to the way we use object oriented programming in our program, it is relevant to look at what the hearts from our class represent. Even though the heart-object has the exact same visual as the like-button of the Instagram interface, it is a different object in the sense that it has different attributes in the game “Like Hunter”. What is interesting here, is not the size of the ‘likes’, nor the speed or the color, but instead how many different opinions they represent. This is also where the critical aspect of our program is represented. When using objects in a program, the designer makes the abstraction and chooses what to focus on and what not to focus on. Hereby reducing, for instance on a platform like Instagram, the abilities of the user. Obviously, they are able to comment (almost) on whatever they want, save, send or share an image, but in the case of the like button, the user is left with a binary choice; she can click the white heart and make it red, or she can choose not to. There are no other options for the user to express herself and her opinion. The interesting thing that we think relates to our program is the way in which lots of different opinions are simply reduced to the abstraction of an object, and then, in a software like Instagram, take shape as simple numbers. These numbers come to represent only one thing, even though a various amount of intentions could lie behind different people clicking the like button for the same picture.

Buttons & Quantification

With emphasis on the social media platform Instagram, the game “Like Hunter” seeks to address different issues regarding the users and use of Instagram. Both in relation to the choosing of hearts as “likes”, and the hunt for these as the main activity in the game, but also the goal of the game itself – to hunt for as many likes and followers as possible, and how these specific numbers are represented in the game.

As a function on Instagram, the heart represents the ability to “like” a picture, and when the heart is pressed, a feedback is received on the liked picture, where the heart changes from a white colour with a thin black frame, to a filled red colour. In relation to this feedback, the text “Buttons” by Søren Pold argues that buttons in software should be aesthetically pleasing with a satisfying response, such as the feedback on Instagram when a picture is “liked” (Pold, 2008). With reference to this statement, Pold highlights this feedback in the following quote: *“They should evoke confidence by returning a smooth response, not plastickey or cheap, even though it might have nothing to do with the functionality.”* (Pold, 2008, p. 34) The “like”-button on the Instagram platform is an example of this. Though it is a button on a touchscreen, the aforementioned feedback of “liking” a picture provides a clear message of the “like”-button itself, both in its shape’s relation to the action of liking something, but especially in the other added feedback of liking a picture – the increasing number of likes a picture receives.

This specific function on Instagram that counts the number of “likes” on a picture, is also an important part of the game “Like Hunter”. In relation to the feedback the number of “likes” is displayed with every picture on Instagram and is always updated. In other words, “liking” on Instagram provides two kinds of feedback. As previously stated, Instagram, as a social media platform, has developed in recent years, in its shifting purpose from a supernumerary number of private users, to many users with public accounts basing their careers off of the use of Instagram using different types of business approaches such as showcasing sponsored content on their pictures. This is highlighted in the game in both in the action of “hunting likes” and in the more aesthetical choices of the game’s appearance such as the faded background of a profile on Instagram and in the choice of picture for the controlled object, the “Like Hunter” itself.

In addition to these attributes of the game, this “hunt for likes” is the issue the game actively addresses in the increasing numbers of “likes” and “followers” as the player hunts the “likes” in shapes of hearts in the game. As previously mentioned, this is an issue highlighted in the text by Benjamin Grosser, “What Do Metrics Want? How Quantification Prescribes Social Interaction on Facebook”. In this text, Grosser describes this number of “likes” or “followers” as metrics and question the effects of metrics from different angles (Grosser, 2014). One angle in Grosser's text is the before mentioned quantification, that is described as a reduction to numbers, thereby the metrics of "likes" and "followers". This is also stressed by the following quote: *"Thus, within our system of capital, quantification becomes the way we evaluate whether our desire for more is being fulfilled."* (Ibid.) In other words, this increasing number of "likes" of a picture on Instagram can become the main way users estimate their value, by basing their self-worth off of the number of likes they receive on a picture. This is one of the main issues that is addressed by the game "Like Hunter", and Grosser also highlights this in the text, by questioning this "desire for more" (Ibid.). He raises numerous questions in relation to this: *"When faced with a number, why do we want that number to go higher? Why is more–more friends, more "likes," more shares–better than less?"* (Ibid.) Grosser suggests that the answers to these questions lie in the constant comparison to others, and this everlasting desire for more is represented in "Like Hunter" in the inability to actually "win" the game, as "hunting for likes" appears eternal.

Critical thinking

Addressing the aforementioned issues, our program can be seen as a critical work in itself. It touches upon the issues of how object orientation can contribute to quantification in the sense that the opinions of a various amount of people are reduced to numbers - the quantification comes to play a major role in our use of social media. Digging into the code, we see how there is a connection between the technical aspect of using object oriented programming to what influence it has on digital culture. The same as with the abstractions related to object oriented programming, and how these abstractions might have an influence on how we perceive and interact with social media, as with the likes and followers. The program can also be seen as critical simply in the way it turns a social media platform into a game. The gamification shows how social media in general has come to revolve around gaining more and more likes and followers as if you were trying to increase your score in a game. On a final note, there are obviously a lot of different perspectives that can be drawn in relation to the critical aspect of our program -

looking at it from a broader perspective, it would also have been interesting to include subjects of data sharing, data capturing and security.

References

- Crutzen, Cecile and Kotkamp, Erna. Object Orientation. Software Studies\a lexicon. Eds Matthew F. MIT Press, 2008. 200-207
- Grosser, Benjamin ”What Do Metrics Want? How Quantification Prescribes Social Interaction on Facebook” in *Computational Culture* no. 4 (2014). http://computationalculture.net/article/what-do-metrics-want
- Kulturstyrelsen. (2015). *Mediernes udvikling i Danmark 2015/Sociale medier – Brug, interesseområder og debatlyst.*Copenhagen: Kulturstyrelsen.
- Lee, Roger Y. *Software Engineering: A Hands-On Approach*. Springer, 2013. 17-24, 35-37.
- Pold, Søren. "Button." *Software Studies\ a lexicon*. Eds. Matthew Fuller. MIT Press, 2008. 31-36.
- Tassy, A. (2018). *IT-anvendelse i befolkningen 2017.*Copenhagen: Danmarks Statistik.