ABSTRACTION

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- *One of the most powerful problem-solving tools of computer science is abstraction*. Abstraction isn't about solving a particular problem faster or with fewer resources.
- Instead, *the goal of abstraction is to allow us to arrange information more quickly and reliably in our heads and ignore irrelevant details*.
- The purpose of abstraction is largely for helping humans think, rather than helping computers work.

- We'll look at abstraction with the help of Jing, the mayor of a small city.

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- A city government is made of dozens, hundreds, or even thousands of people who have their own roles, who help the city in their own ways, and whose responsibilities may overlap with one another.

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- In a very small town, Mayor Jing might be able to keep track of a dozen or so people's overlapping job descriptions, with everyone reporting directly to the mayor.
- In a city with hundreds or thousands of employees, this approach will certainly run into problems. What could go wrong?

○ The mayor will need to spend too much time communicating with her employees.

○ It's hard to keep track of who needs to be informed about any specific issue (like tree planting).

○ Both of these.

○ Neither of these.

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- Correct answer: **Both of these.**
- All of the downsides mentioned are potential problems. With more and more employees, Mayor Jing will need to start spending an increasing portion of her time “micromanaging” her many employees.
- Overlapping responsibilities mean that every task potentially needs to be coordinated with a different set of employees. This confusion can sometimes make it hard to ensure that all the important parts of a city's business, like fixing water pipes, are ultimately covered.

- As organizations grow, it's usual to break people up into separate groups or departments, which each have their own leaders, goals, and objectives. In the image below, Mayor Jing has separated her employees into three departments: Fire, Parks and Rec, and Sanitation.

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

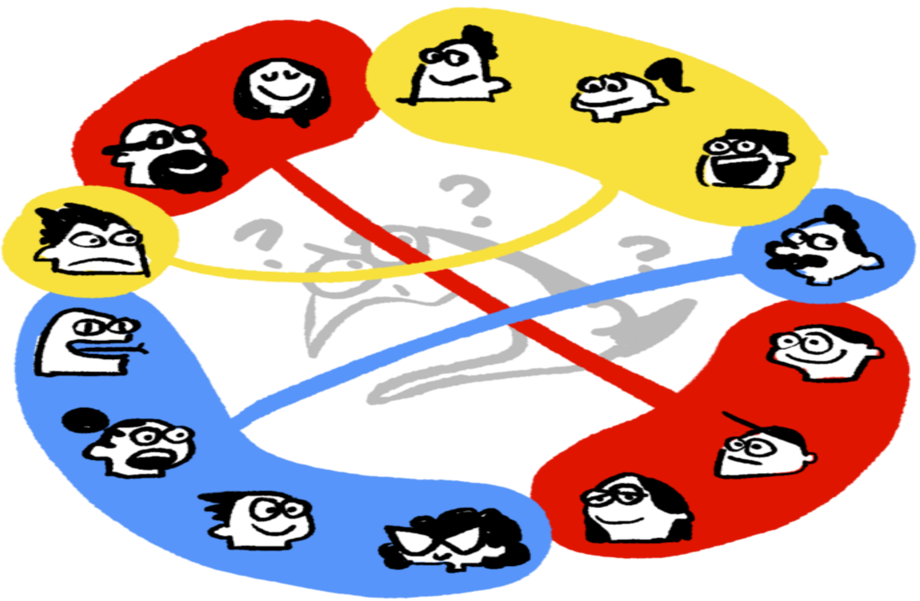
- Creating a box that everyone agrees on and labeling it “The Fire Department” is a form of *abstraction*. Mayor Jing can tell the “Fire Department” to do something—like focusing on rescuing cats in trees—without necessarily understanding all the details of how that affects the individual humans, buildings, and fire trucks.
- This kind of change is not without drawbacks. What could go wrong with the particular divisions shown here?

- It's not clear which department is supposed to be in charge of planting trees.
- Sewers might get inspected too much, or not enough.
- It's hard to keep track of all the people that need to be informed about fighting fires.
- Fewer people are responsible for fixing water pipes.
- All of the above

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- Correct answer: **Sewers might get inspected too much, or not enough.**
- In this arrangement, tree planting is the clear responsibility of the Parks and Rec, the Fire Department is in charge of fighting fires, and Sanitation is fully in charge of fixing water pipes.
- Because sewer inspection ended up partially under Water and Sanitation and partially under Parks, the most likely problem is confusion about responsibilities between these two groups.

- Splitting up a complicated system, like a city government, into pieces **involves tradeoffs**, because different splits may have different advantages and disadvantages. But it's possible to imagine that some arrangements are clearly better or worse than others, regardless of the tradeoffs.



- It's helpful to realize that many times the abstractions we encounter are the result of historical accidents, other people's personal preferences, or just because it's what everyone else does. This is true in computer science and in city governments. For example, it's easier for Mayor Jing to follow the convention of having a Parks and Rec Department and a Fire Department, just because those are common departments that other cities and future employees will be used to.
- It's important to organize complicated systems into understandable parts with more or less well-defined tasks. In both computer systems and human systems, these abstractions exist to help ***humans understand the system, despite its complexity***.

- **Both computer systems and human systems can end up with lots of levels of abstraction:** Mayor Jing communicates with the head of the Fire Department, who then communicates with the heads of her four firehouses, who then each communicate with their half-dozen or so shift leaders, who then communicate with their shift employees.

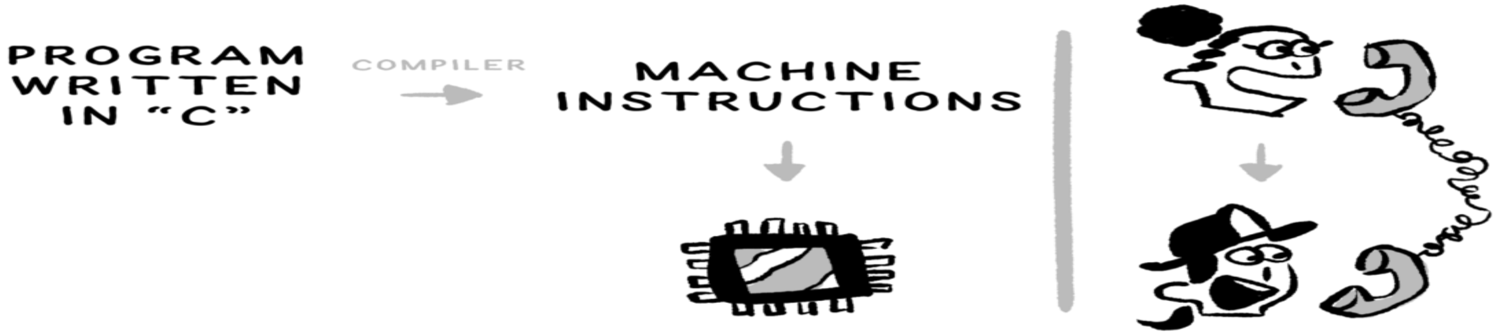
- What is a likely symptom of this proliferation of abstraction layers in Mayor Jing's government?

- City employees are unclear who they report to.
- It takes a long time to communicate with everyone.
- Mayor Jing doesn't know about all the fires that happen.
- Citizens don't know who is responsible for parks.

Correct answer: **It takes a long time to communicate with everyone.**
If Mayor Jing can't communicate directly with everyone, and if all communication always follows her chain of command, then communication will almost necessarily be slower.

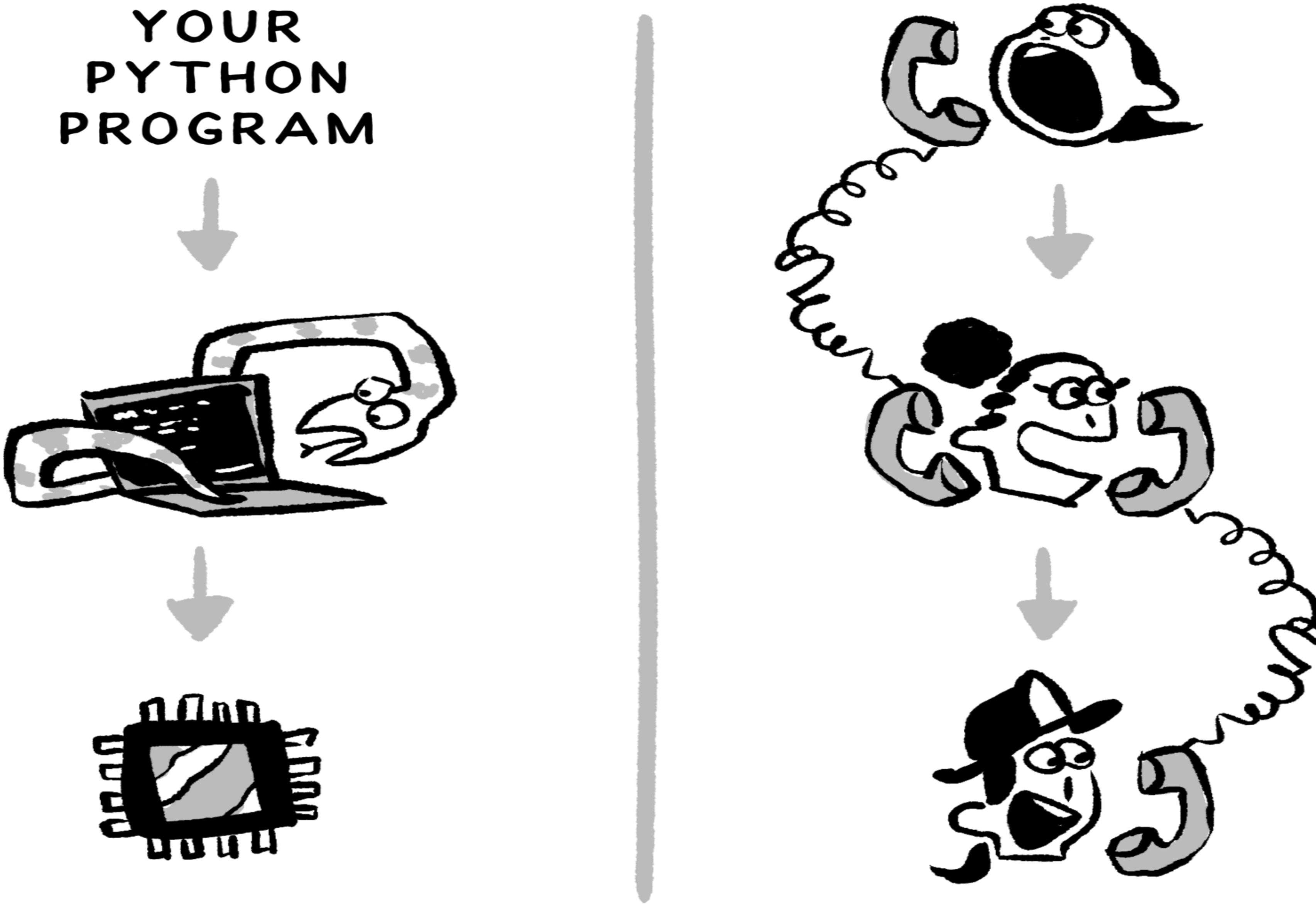
Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- The kinds of layers that Mayor Jing deals with in her organizational structure are everywhere in computer science. When a company like Intel or AMD designs the computer chip powering a computer or phone, that company also publishes an abstraction layer: a set of commands for controlling the chip. These commands are called *assembly instructions* or *machine instructions*.
- You can create a program by writing machine instructions directly, but this is way too hard. It would be like Mayor Jing calling every employee to micromanage every detail of rescuing a cat from a tree or repairing a water pipe.
- Instead of writing machine instructions, it's easier to write commands using a programming language that's easier to write and read. **A process called *compilation* transforms the code you write into machine instructions**. Those machine instructions then direct the chip, as the Fire Department's chief directs her subordinates.



Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- Computer systems and human systems do frequently have many layers. Many languages that are designed to be easier to use, like Python, don't ever turn the code you write into machine instructions. Instead, there is a second computer program, called an ***Interpreter***, which is made up of machine instructions.
- The program you write in Python is a bunch of Python instructions that the *Python interpreter* understands. **The Python interpreter is a list of machine instructions that the *machine* understands**. This is like the multi-level delegation where Mayor Jing and the firefighters don't ever talk directly but instead go through the fire chief.



Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- **There is a cost to having our computer code run on top of Python**, just as there is a cost to having Mayor Jing communicate with her firefighters through the head of the Fire Department. In both cases, **the extra layer of communication slows things down.**
- If we write a program with Python, we may worry that our program would run faster if we'd written it in some other programming language. **It's conceivable that our program would run *even faster* if we'd painstakingly written the machine instructions by hand instead of using a more human-friendly programming language**. Perhaps our program would run *faster still* if we'd specifically designed the hardware's structure to solve the problem rather than relying on machine instructions!

- Each of these theoretically faster and more direct approaches is also more difficult and complicated. **There's a resource tradeoff between the amount of time it takes for humans to solve a problem and the speed with which a computer can implement the human-designed solution.**

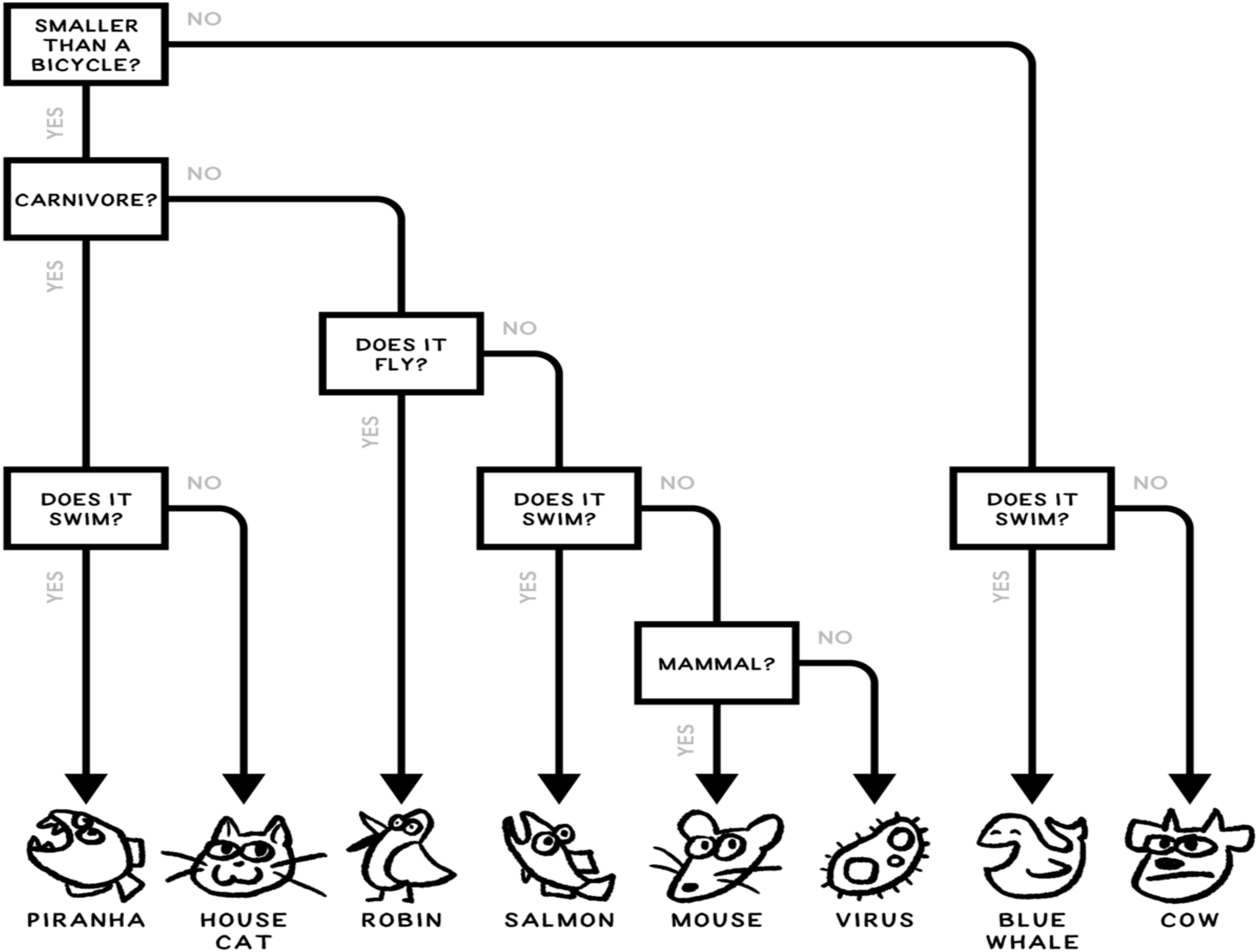
Making Decisions

Powered by Brilliant Application

Prepared by Dr Syed Khaldoon Khurshid

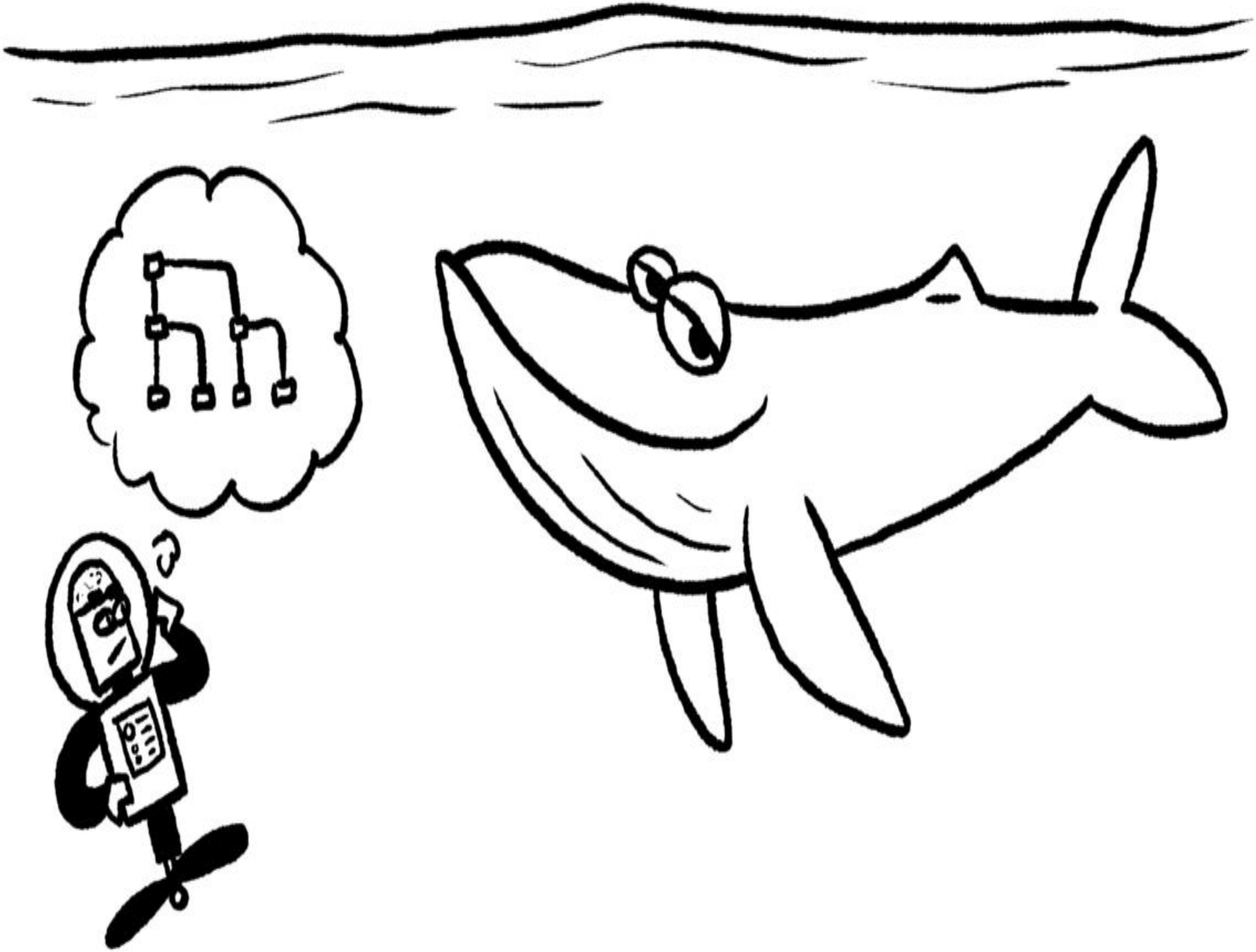
- **Computers can *make decisions*, and computers can do things *very very* fast.** Right now, a computer is deciding what the solution to a mathematical equation is. Somewhere else, a computer is deciding whether to suspend someone's credit card to protect them from fraud, and another computer is deciding whether an image represents a stop sign or a bird.
- **An important part of computer science is understanding how computers can make *the right decisions*, or at least pretty good ones.**

- One of the ways computers (and sometimes humans) make decisions is with a structure called a ***decision tree***. Decision trees encode a series of simple yes-or-no questions that you can follow in order to answer a **more complex question**. Here is a silly decision tree that helps you decide which of eight different creatures you're dealing with:


SMALLER THAN A BICYCLE?
NO
YES
CARNIVORE?
NO
YES
DOES IT FLY?
NO
YES
DOES IT SWIM?
NO
YES
DOES IT SWIM?
NO
YES
MAMMAL?
NO
YES
DOES IT SWIM?
NO
YES
PIRANHA
HOUSE CAT
ROBIN
SALMON
MOUSE
VIRUS
BLUE WHALE
COW

- If a computer were using this diagram while looking at a creature that happened to be an adult blue whale, the computer would start at the top box, the ***root*** of the decision tree. The computer would ask whether the creature was smaller than a bicycle. The answer is definitely *no*, a blue whale is not smaller than a bicycle.

- Next, the computer would follow the "no" path, the arrow that goes to the right. The next box contains the question "Does it swim?" The answer to that is definitely "yes," blue whales swim. So the computer would continue down the "yes" path, which is the arrow that goes straight down. The computer then correctly concludes that it's looking at a blue whale.



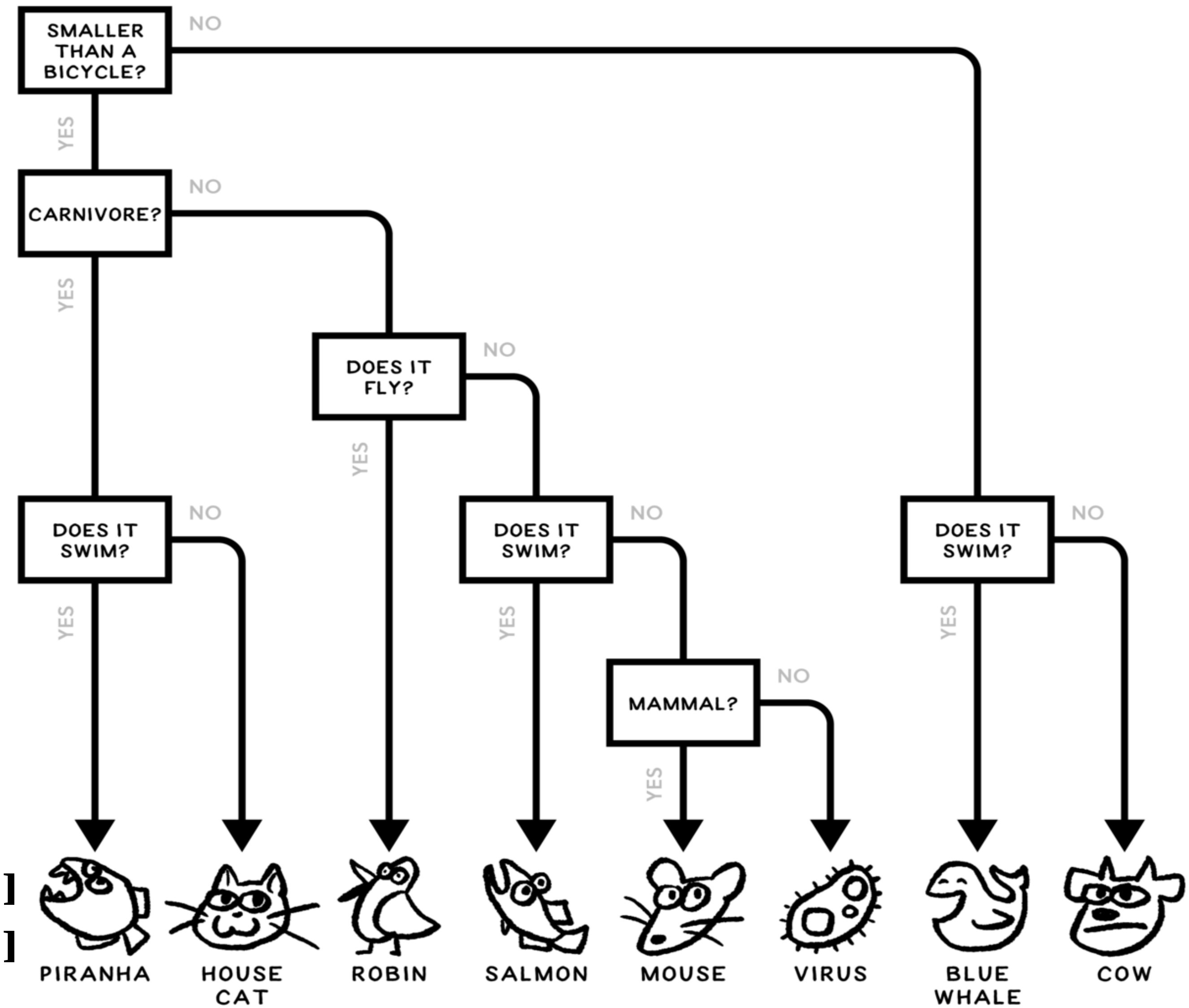
- A computer is using this decision tree. Which of the listed creatures will require the computer to ask the ***most* questions**?

◯ House Cat

◯ Salmon

◯ Mouse

◯ Cow

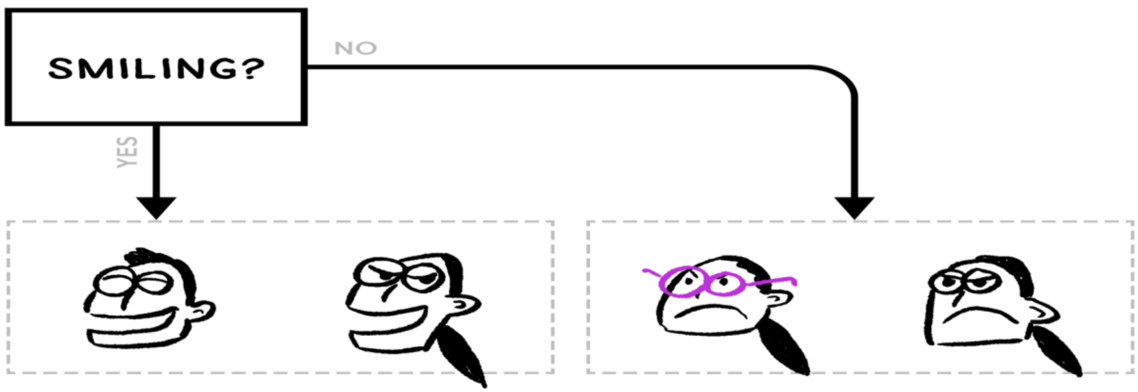


- A computer scientist can use this decision tree to write a simple face recognizer. The computer scientist first writes three simpler tests to detect glasses, long hair, and smiling. The decision tree organizes these simple tests, allowing a computer to distinguish between all the faces.
- **The first question at the very top of the decision tree is an especially important one.**

- Imagine that you want to make a decision tree to distinguish the four faces above. If you want the root of the decision tree to split the faces into two groups of two, which of these questions should you ask?

- ○ Do they have long hair?
- ○ Are they smiling?
- ○ Do they have blue hair?
- ○ Are they wearing glasses?
- ○ None of these will work.

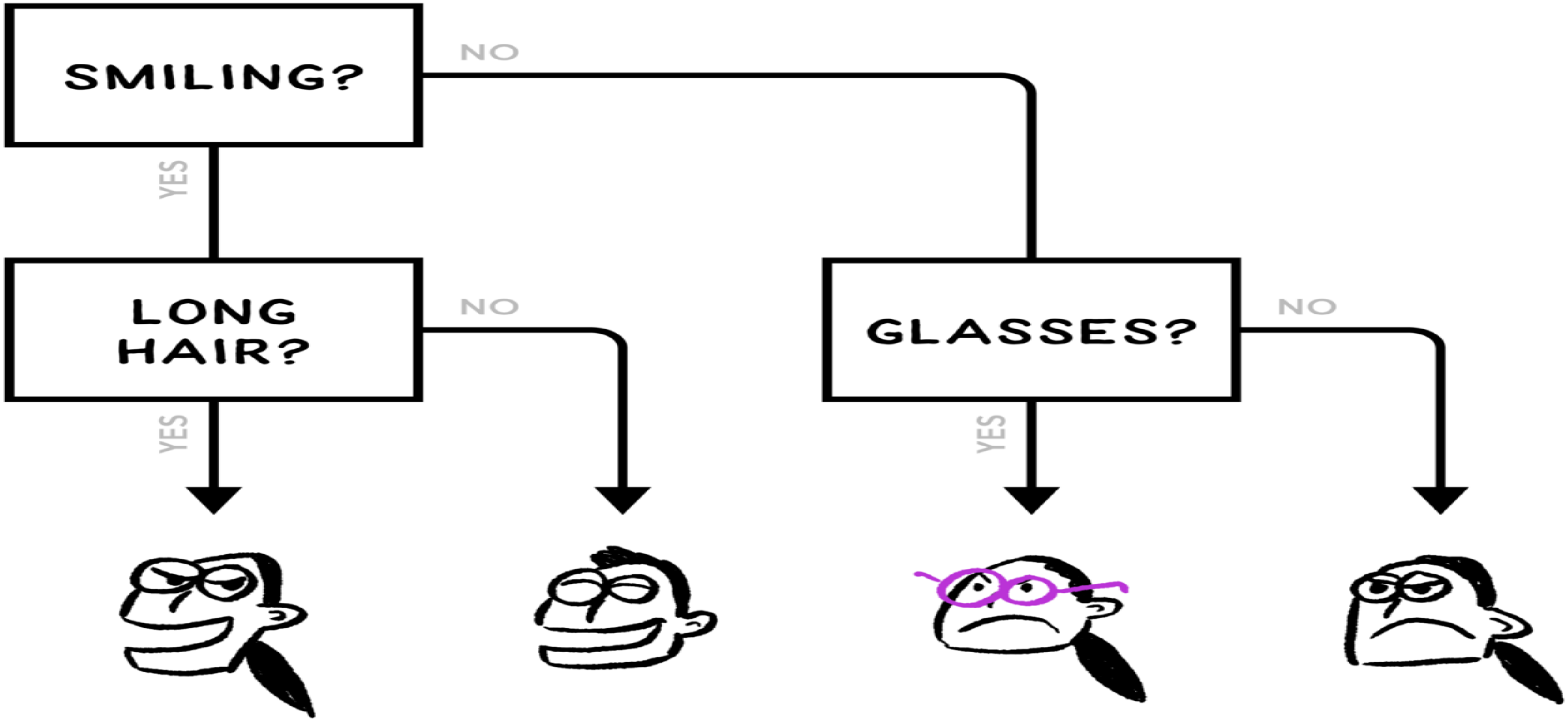
- The first question in your decision tree will split the faces into two distinct groups. What is a single question that you could ask of ***both*** groups of two in order to identify any single face?



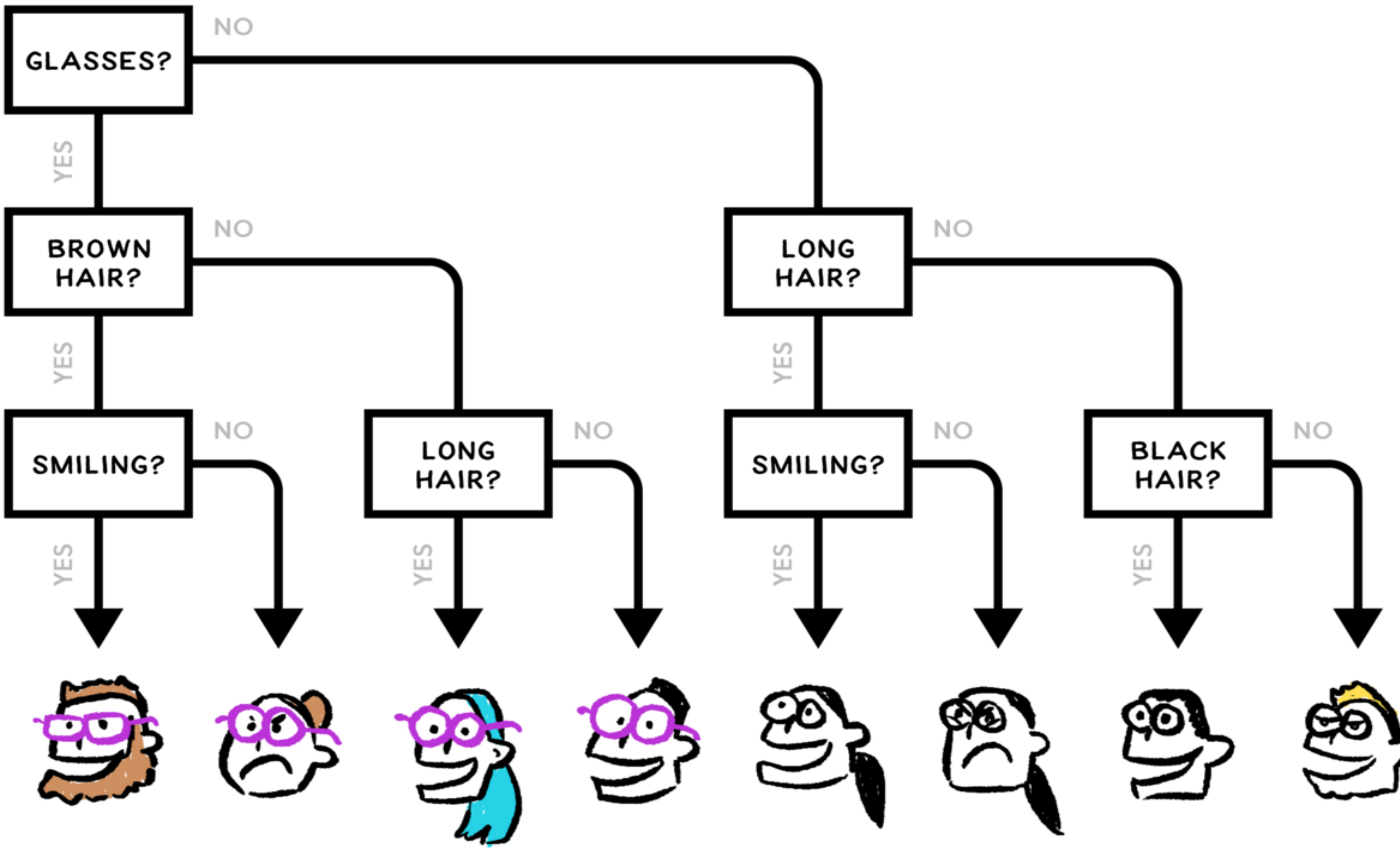
- Do they have long hair?
- Do they have blue hair?
- Are they wearing glasses?
- None of these will work.

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

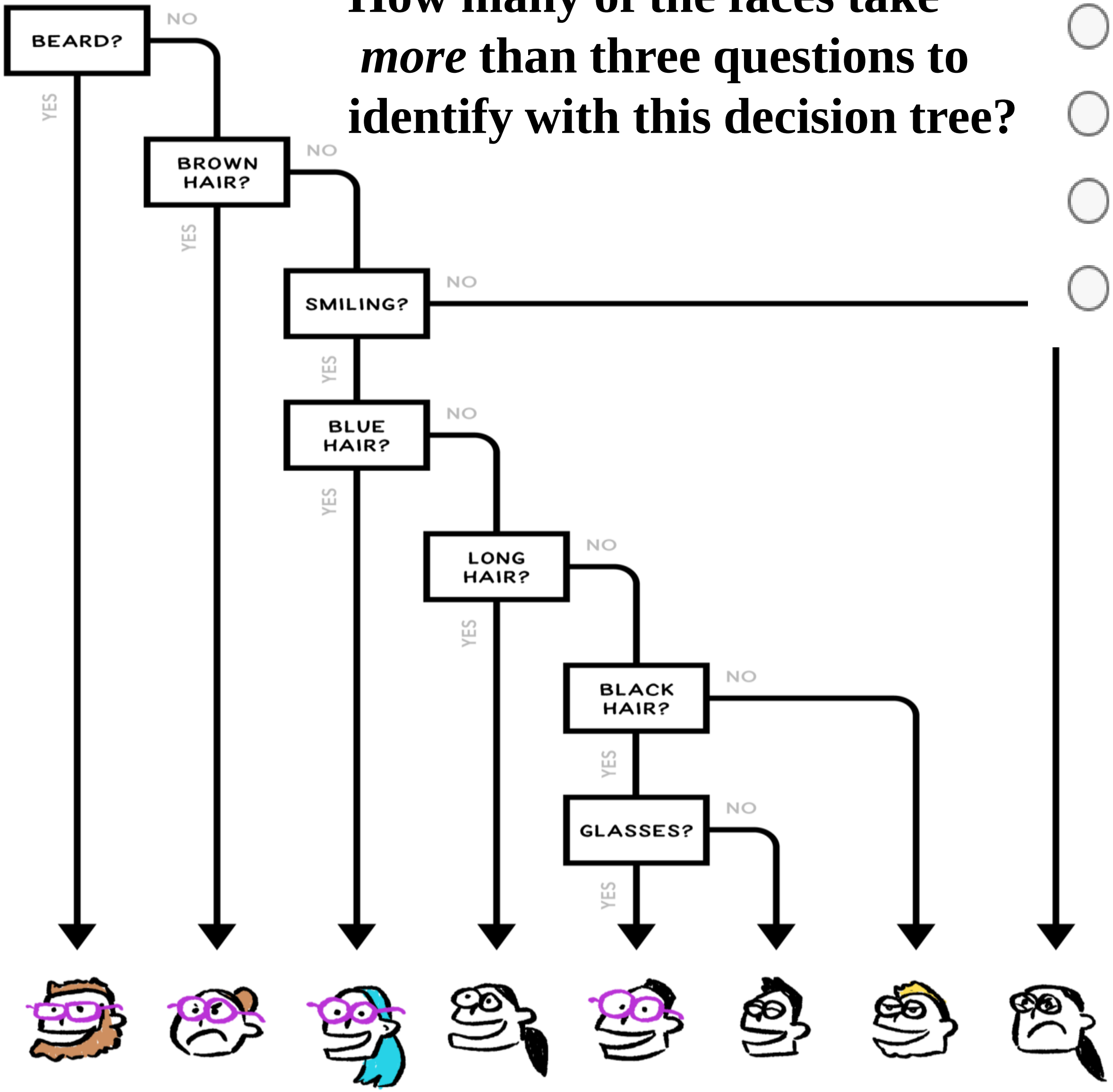
- None of the choices you were given were able to further split up both of the groups of two faces. You can nevertheless make a decision tree that distinguishes all four faces, because the tree can ask a ***different*** follow-up question based on the answer to the first question. This flexibility makes decision trees quite powerful.



- The ***shape*** of a decision tree can make a big difference in what happens when you use it. If your computer program is using this decision tree, how many questions, on average, will the computer program need to ask and answer in order to distinguish a random face?



The decision tree from the previous page distinguishes between eight faces with exactly three questions.
This decision tree Distinguishes between the exact same set of eight faces; however, it takes an average of 4 3/8 questions to identify a random face with this decision tree.

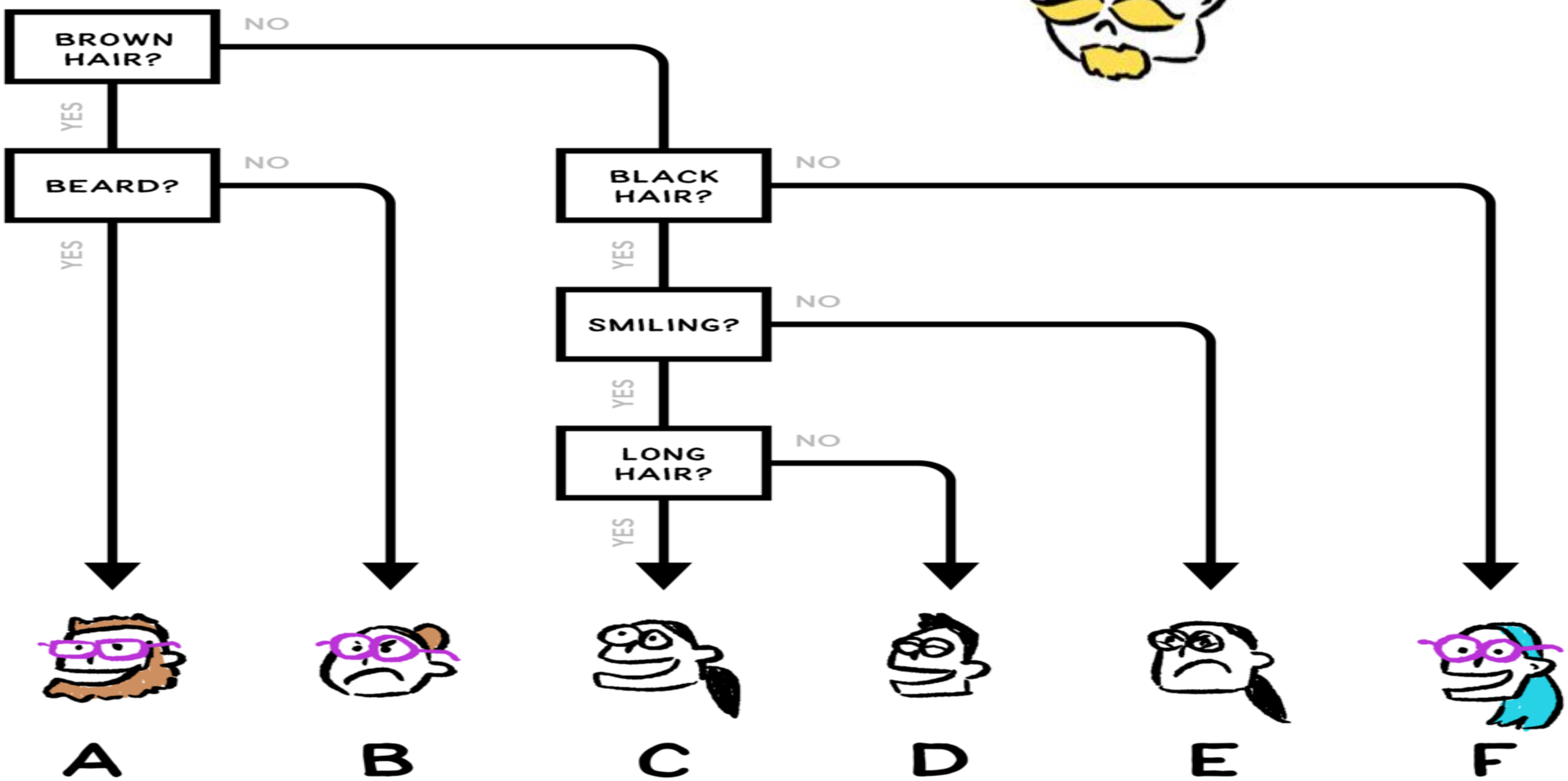


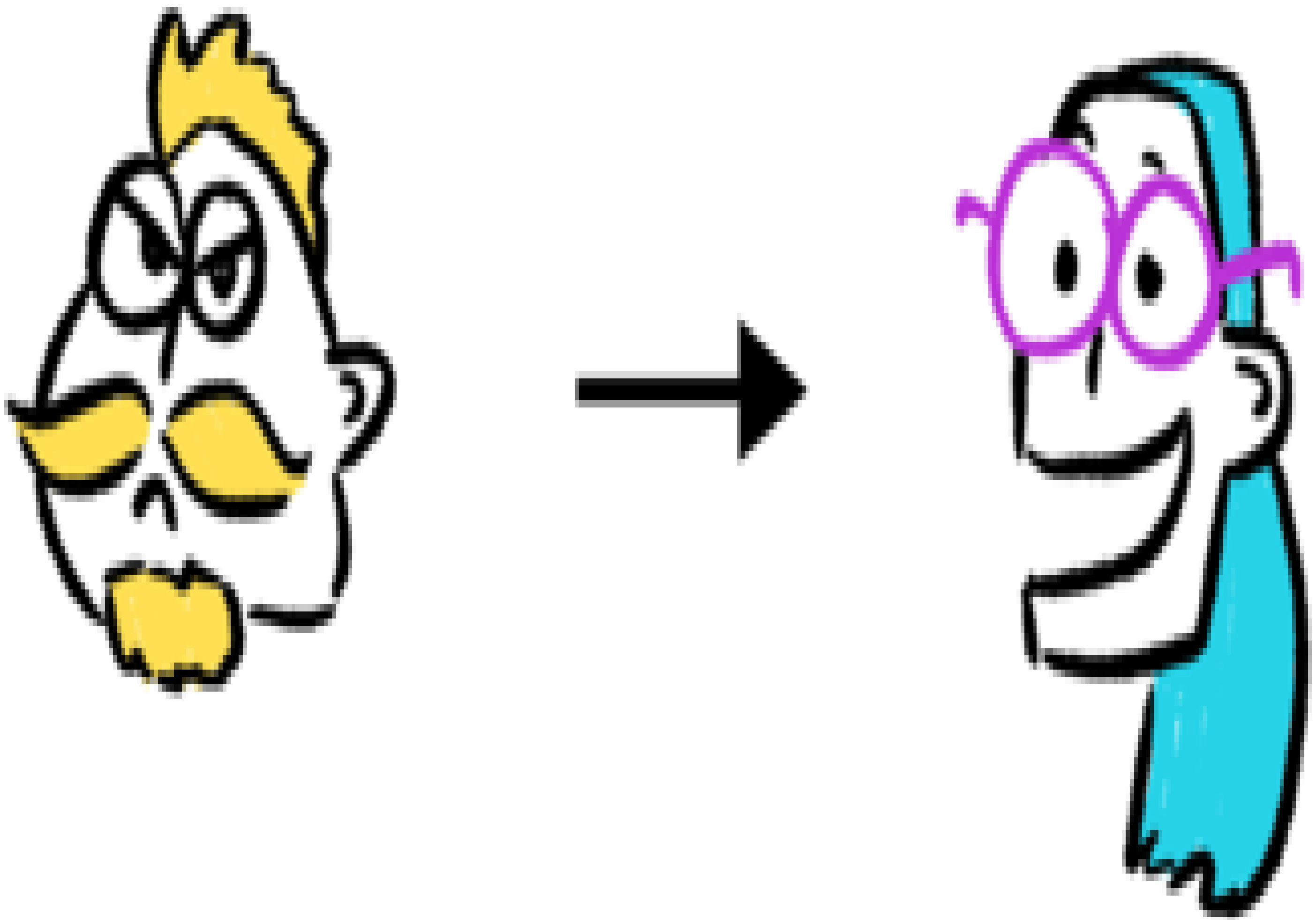
How many of the faces take *more* than three questions to identify with this decision tree?

- 3
- 4
- 5
- 6

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid

- This decision tree distinguishes six faces. Which face will be chosen if the decision tree is used to identify this different face:



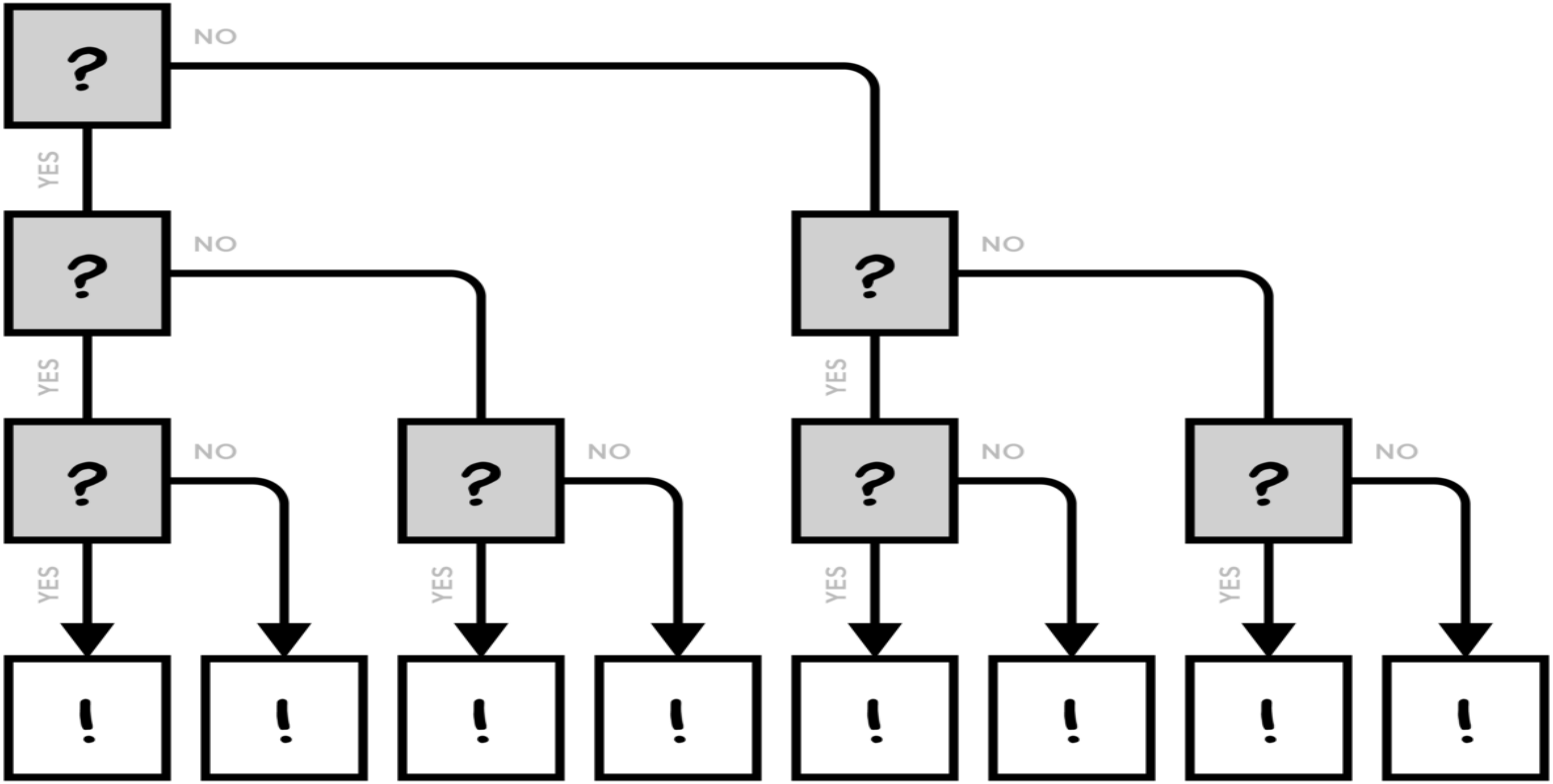


- The decision tree on the previous page did not do a very good job classifying this face. These faces are almost as different as they can be! One reason that the decision tree failed so badly was that the first few questions focused on hair color and the decision tree contained no faces with blond hair. ***The lack of blond-haired faces in the decision tree made it easier to badly misclassify a blond-haired face.***
- This was a simple example, but if you pay attention to the news, you'll see many real-world examples of exactly the same kind of failure.

- It can be funny when computers make mistakes because they were designed with limited information. For example, computer programs designed to identify pictures sometimes "hallucinate" sheep in every field. This is because the computer programs are designed based on a bunch of pictures taken by people on vacation. The pictures of fields taken by people on vacation mostly also contain sheep.
- **Other cases of computers failing to make good decisions are more worrying. Many computer programs that do real-life facial recognition don't work well for people with darker skin. This can happen when the facial recognition program is designed around a bunch of pictures the designers had easy access to, and those pictures mostly contained white people.**

- Decision trees are useful tools for computer scientists. They turn simple yes-or-no decisions into more complex decisions involving many different options. Computers are often better at answering simple yes-or-no questions, so decision trees help computers manage more complexity.
- **Some computer scientists use decision trees designed by human experts to help computers make smarter choices. Other computer scientists use computer programs that do machine learning in order to create the best decision trees for solving a new problem. When you're using decision trees, the order of questions can make a big difference in the number of questions you have to ask.**

- Decision trees *aren't* the right tool for every problem. For example, how would you use a decision tree to sort a **deck of cards**?



Google Classroom:

- https://classroom.google.com/u/0/c/MjExOTgyMDc0NjY5
- Code for Joining: kaxlulx

Powered by Brilliant Application
Prepared by Dr Syed Khaldoon Khurshid