Question 1

In lecture, we saw that in order to run a C program (e.g., `hello.c`), we first need to run the command `make hello`, and then run the command `./hello`.

a) What does running `make hello` do?
b) What does running `./hello` do?
c) What might happen if you were to run `./hello` without first running `make hello`?

Answers

a) TODO
b) TODO
c) TODO

Question 2

In the first version of the game Civilization, each character had an "aggression" rating, stored as an 8-bit integer. The programmers of the game gave one of the characters, Gandhi, an initial aggression rating of 1, the lowest of any character in the game. The game also had logic such that, once a character "adopted democracy," the character's aggression rating would decrease by 2.

However, the game had a bug whereby Gandhi would become extremely aggressive, with an aggression rating of 255, if he adopted democracy.

What might explain why Gandhi suddenly became so aggressive in the game?

Answer

TODO

Question 3

Recall that, in lecture, we saw the following two blocks of code, both of which print the same output.

Version 1

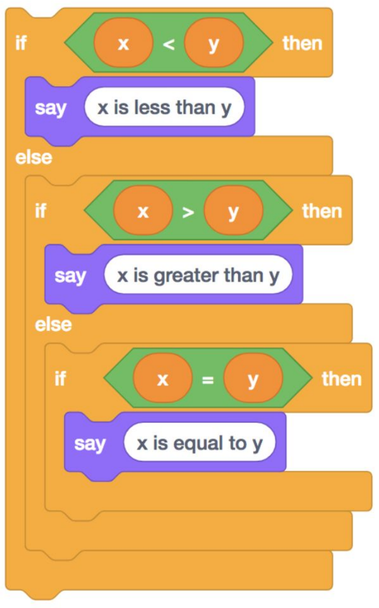
```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else if (x == y)
{
    printf("x is equal to y\n");
}
```

Version 2

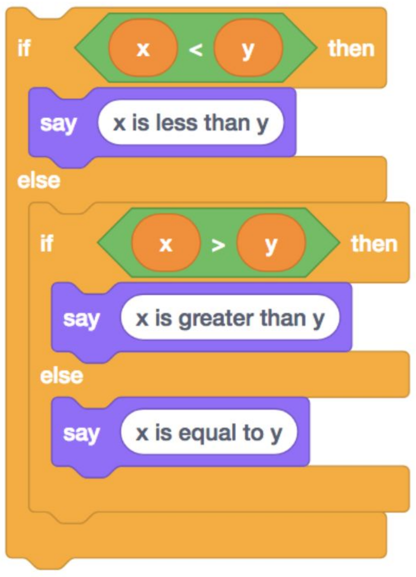
```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
    printf("x is equal to y\n");
}
```

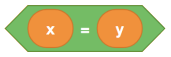
These are really just the C equivalents of the following two blocks of Scratch code.

Version 1



Version 2



a) Why, in C, do we use two equals signs (`==`) when we write `else if (x == y)`, whereas in Scratch we use just a single equals sign (`=`) in 

?

b) Why is Version 2 of the code, whether implemented in Scratch or in C, marginally more efficient than Version 1?

Answers

a) TODO

b) TODO