Не выполняя код, ответить, что выведет браузер и почему:

a.

if​ (!(​"a"​ ​in​ ​window​)) { ​var​ a = ​1​; } alert(a);

Код считывается таким образом, что вначале мы задаем var a, а только потом проверяем, нет ли его у нас. Таким образом условие не выполняется, и alert(a) выдаст undefined

b.

var​ b = ​function​ ​a​(x) { x && a(--x); }; alert(a);

выдаст значение а, которое у нас тут не задано, поэтому ничего не выдаст

c.

function​ ​a​(x) { ​return​ x \* ​2​; } var​ a; alert(a);

объявления функций имеют преимущество перед объявлениями переменных, если нет инициализации. Здесь нет инициализации, поэтому предупреждение отображает исходный код функции.

d.

function​ ​b​(x, y, a) { ​arguments​[​2​] = ​10​; alert(a); } b(​1​, ​2​, ​3​);

10. внутри функции присваевается значение 3-му аргументу, после чего он выводится

e. \*

function​ ​a​() { alert(​this​); } a.call(​null​)

когда null передается call(), он по умолчанию глобальный объект, который является window. Учитывая это, пример кода может быть переписан более понятным способом, как:

function a() {

alert(this);

}

a.call(window);

Этот код делает очевидным, что предупреждение будет отображать строковый эквивалент window объекта.