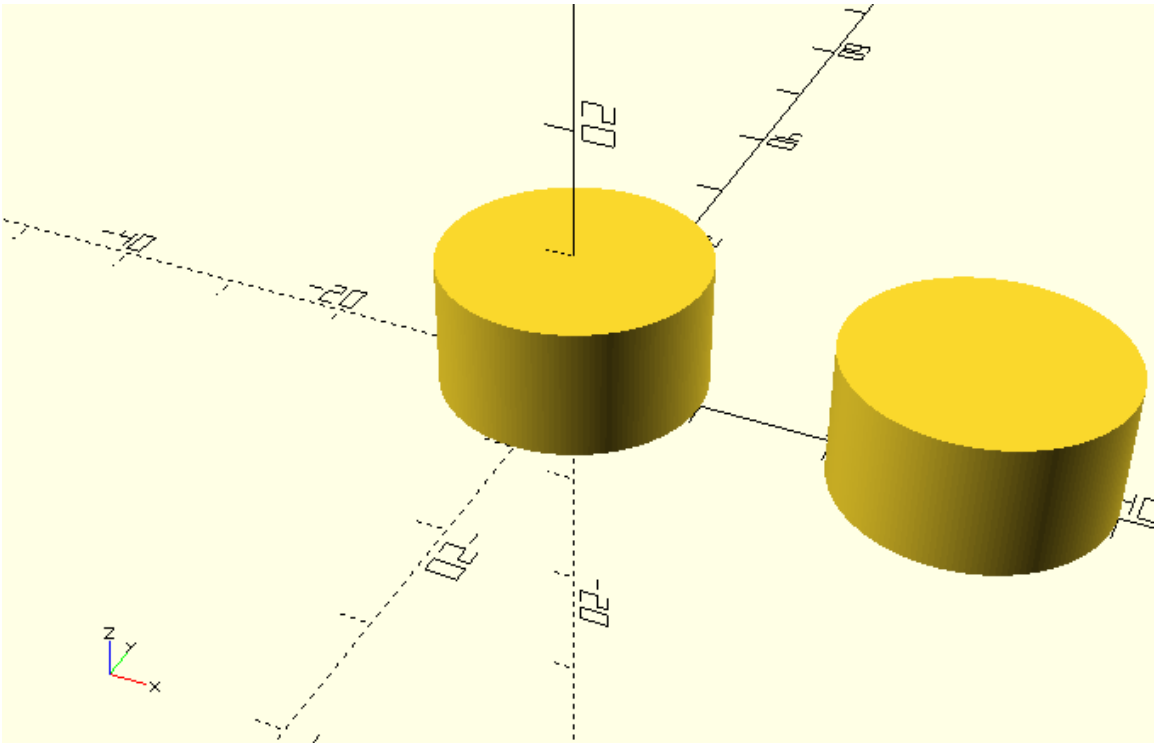
Chapter 6

OpenSCAD variables

In the previous chapters you have made use of variables to parameterize your designs and make them easily customizable. Specifically, you have been assigning them numerical values at some part of your script and then using their stored value in some other part. For example, you can set a wheel_radius variable equal to the desired wheel radius and use that variable in the corresponding statements that create the wheels of your car. This way you can easily customize the radius of your car's wheels without having to search for and change multiple values, but only by directly changing the value of the wheel_radius variable.

You also learned about an important property of OpenSCAD variables. This is that a variable can only have one specific value. If you assign one value to a variable and then assign it a different value at a later part of the script, your variable will have only the final value throughout the execution of your design. This is demonstrated on the following example.

```
$fa=1;

$fs=0.4;

height=10;

radius=5;

cylinder(h=height,r=radius);

radius=10;

translate([30,0,0])cylinder(h=height,r=radius);
```



Both cylinders have a radius of 10 units, which is the last value that is assigned to the radius variable.

When variables store numerical values, they can be used to specify dimensions of different objects or define transformation commands. Numerical values aren't the only kind of values that can be assigned to a variable. Variables can also hold boolean values (true of false) as well as characters (a, b, c, d, …). As you are going to see on the following topics, by using boolean or character variables you can further parameterize your models and modules.

Conditional variable assignment

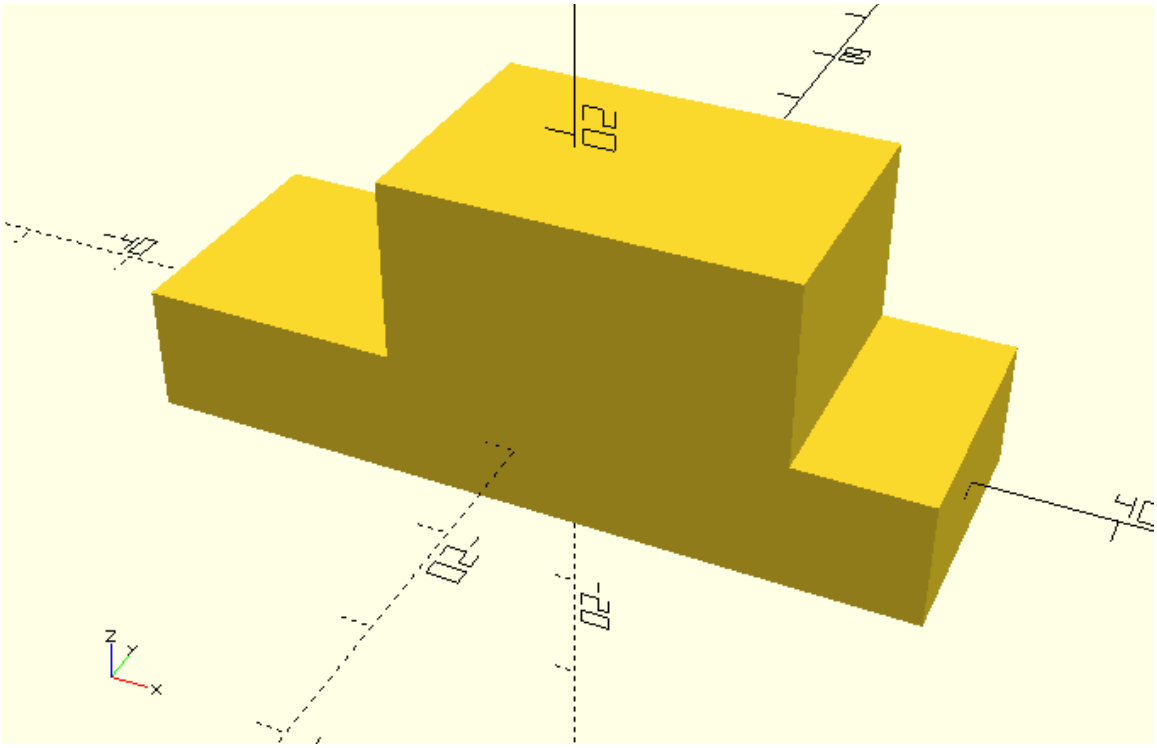
So far you have been assigning specific values to variables using appropriate assignment commands. There are case though where you would prefer the assignment itself to be parametric and dependent on some aspect of your design.

The creation of a car's body requires the definition of various parameters. These parameters can be defined when calling the body module by using corresponding variables that have been defined in your script. One example of this is the following.

```
use <vehicle_parts.scad>

$fa=1;

$fs=0.4;

base_length = 60;

top_length = 30;
```

top_offset = 5;

body(base_length=base_length, top_length=top_length, top_offset=top_offset);



The above version of the car's body will be called the short version. By choosing different values for the variables a long version can also be created.

use <vehicle_parts.scad>

$fa=1;

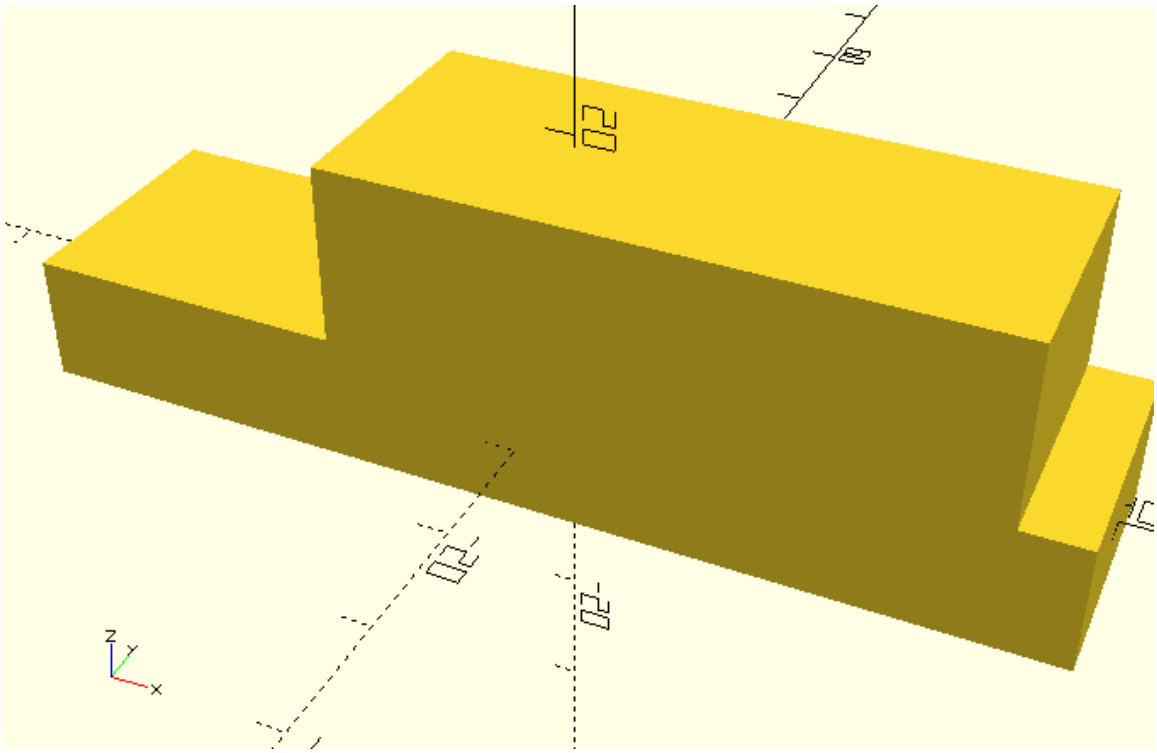
$fs=0.4;

base_length = 80;

top_length = 50;

top_offset = 10;

body(base_length=base_length, top_length=top_length, top_offset=top_offset);



What if these two versions of the car's body are the only versions that you currently interested in? Is there a way to quickly switch between these two versions without having to modify each variable separately?

You may think modifying three variables isn't much work, but the number of required variables on more complex models can easily get unmanageable. Likely there is a solution to this problem, which is the conditional assignment of variables. The conditional assignment of variables is a way to instruct OpenSCAD to assign different values to variables depending on whether some condition is true or false. In this case the condition is whether the car's body should be long or not. You can represent this condition by defining a long_body variable and setting it equal to true if you want the body to be long or equal to false if you don't want the body to be long.

The choice of a long body is represented by the following statement.

long_body = true;

Respectively the choice of a short body is represented by the following statement.

long_body = false;

The long_body variable is called a boolean variable because boolean values (true or false) are assigned to it. The next step is the definition of the conditional assignments which will assign the appropriate values to base_length, top_length and top_offset variables depending on the value of the long_body variable. These conditional assignments can be defined in the following manner.

base_length = (long_body) ? 80:60;

```
top_length = (long_body) ? 50:30;

top_offset = (long_body) ? 10:5;
```

You should notice the following points about the definition of a conditional assignment. First the name of variable is typed out followed by the equal sign. Then follows a pair of parentheses that contains the condition which will be used in the conditional assignment. The condition in this case is a boolean variable. In general, the condition can also be a combination of logical and comparison operations between multiple variables. After the closing parenthesis follow a question mark and the two corresponding variable values that are separated by a colon. If the supplied condition is true, the first value will be the one assigned to the variable. If the supplied condition is false, the second values will be the one assigned to the variable.

By incorporating the above conditional assignments in your script, you can switch between a short and a long car body just by changing the long_body variable from false to true and vice versa.

```
use <vehicle_parts.scad>

$fa=1;

$fs=0.4;


// Conditional assignment of body variables

long_body = false;

base_length = (long_body) ? 80:60;

top_length = (long_body) ? 50:30;

top_offset = (long_body) ? 10:5;

// Creation of body

body(base_length=base_length, top_length=top_length, top_offset=top_offset);


// Creation of wheels and axles

track = 30;

wheelbase = 40;

wheel_radius = 8;

wheel_width = 4;

// Front left wheel
```

translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius, wheel_width=wheel_width);

// Front right wheel

translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius, wheel_width=wheel_width);

// Rear left wheel

translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius, wheel_width=wheel_width);

// Rear right wheel

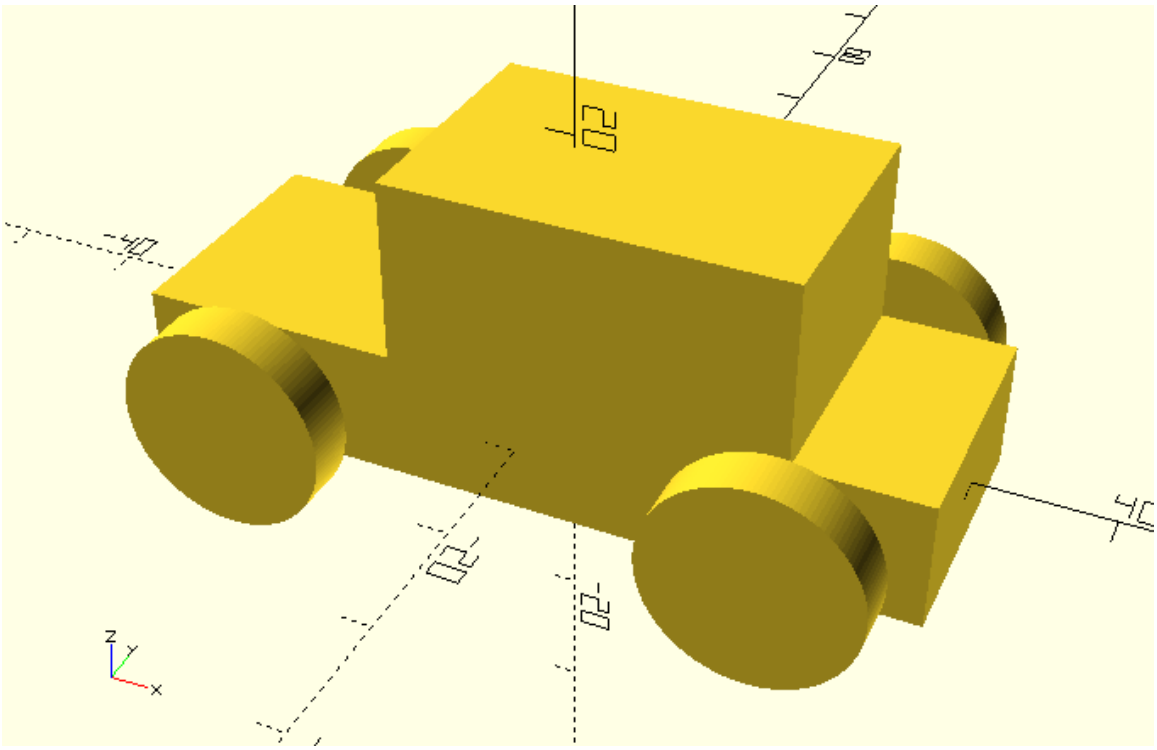
translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius, wheel_width=wheel_width);

// Front axle

translate([-wheelbase/2,0,0])axle(track=track);

// Rear axle

translate([wheelbase/2,0,0])axle(track=track);

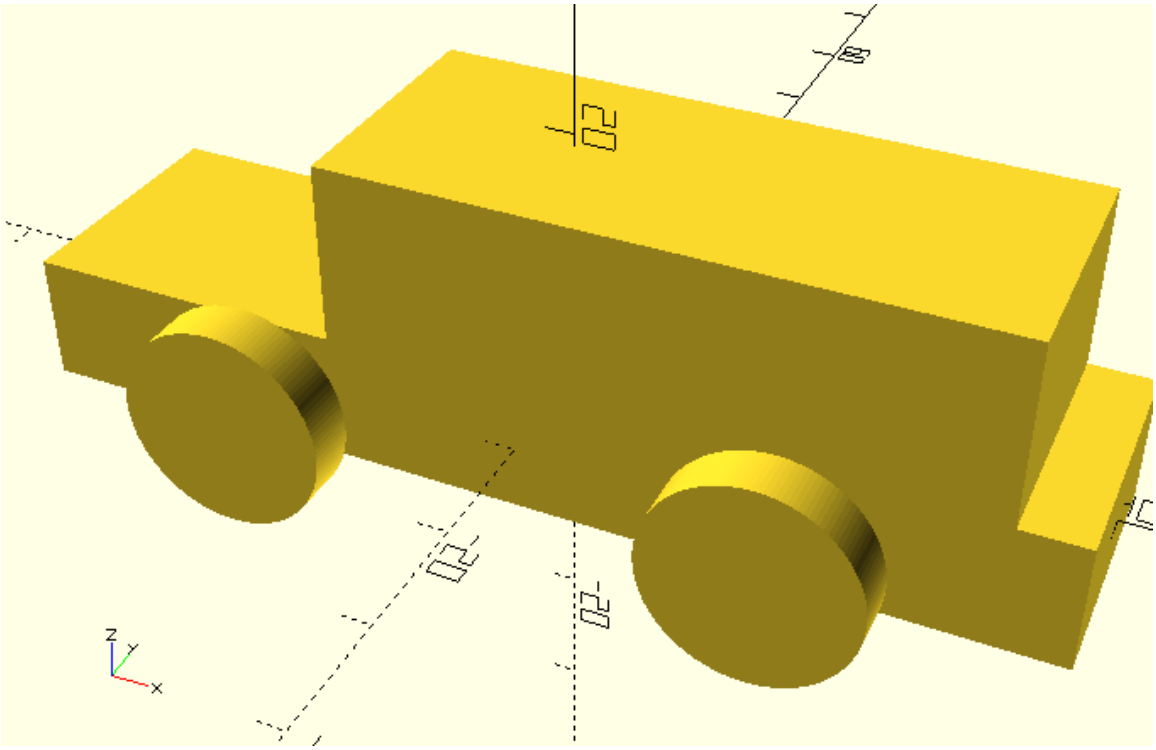


use <vehicle_parts.scad>

$fa=1;

```
$fs=0.4;


// Conditional assignment of body variables

long_body = true;

base_length = (long_body) ? 80:60;

top_length = (long_body) ? 50:30;

top_offset = (long_body) ? 10:5;

// Creation of body

body(base_length=base_length, top_length=top_length, top_offset=top_offset);


// Creation of wheels and axles

track = 30;

wheelbase = 40;

wheel_radius = 8;

wheel_width = 4;

// Front left wheel

translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

 // Front right wheel

translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear left wheel

translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear right wheel

translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Front axle

translate([-wheelbase/2,0,0])axle(track=track);

// Rear axle
```

translate([wheelbase/2,0,0])axle(track=track);

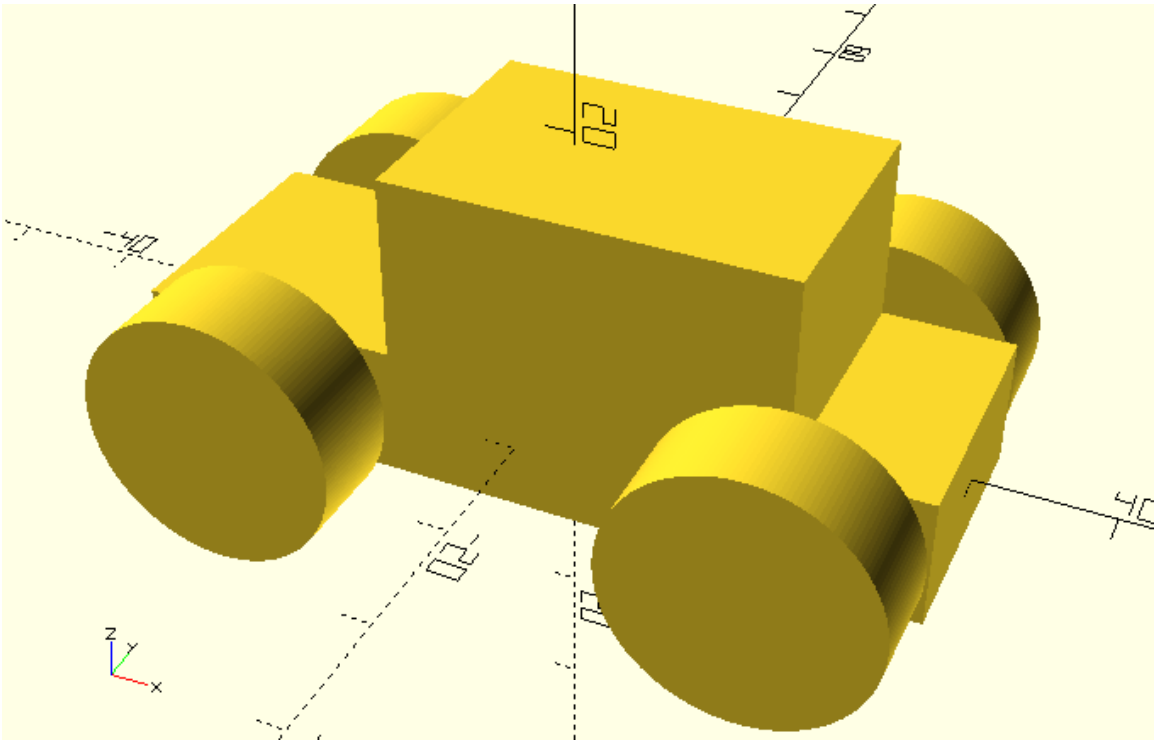


Add a large_wheels variable to the previous example. The variable should only take boolean values. Add two conditional assignments that assign different values to the wheel_radius and wheel_width variables. The large_wheels variable should be used as the condition for both assignments. If the large_wheels variable is false, the wheel_radius and wheel_width variables should be set equal to 8 and 4 units respectively. If the large_wheels variable is true, the wheel_radius and wheel_width variables should be set equal to 10 and 8 units respectively. Set appropriate values to the long_body and large_wheels variables to create the following versions of the car: short body – large wheels, short body – small wheels, long body – large wheels, long body – small wheels.

*short body – large wheels

```
use <vehicle_parts.scad>

$fa=1;

$fs=0.4;


// Conditional assignment of body variables

long_body = false;

base_length = (long_body) ? 80:60;

top_length = (long_body) ? 50:30;
```

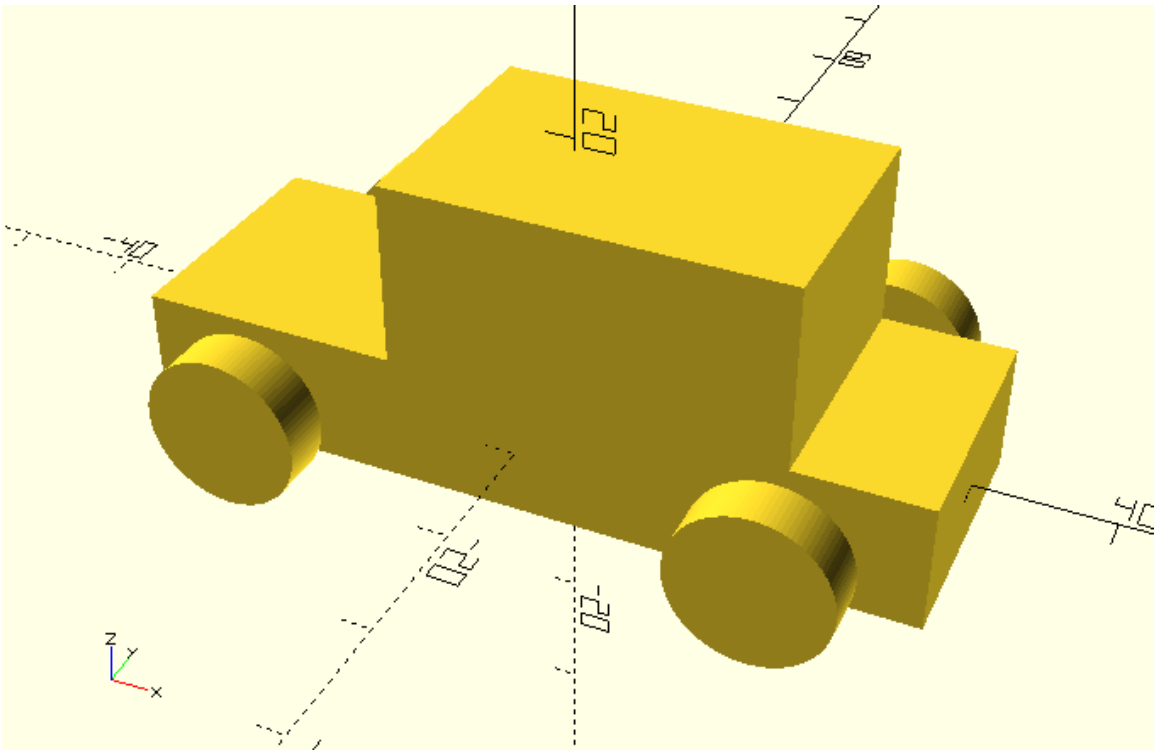
```
top_offset = (long_body) ? 10:5;

// Creation of body

body(base_length=base_length, top_length=top_length, top_offset=top_offset);


// Creation of wheels and axles

large_wheels = true;

wheel_radius = (large_wheels) ? 10:6;

wheel_width = (large_wheels) ? 8:4;

track = 30;

wheelbase = 40;

// Front left wheel

translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

 // Front right wheel

translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear left wheel

translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear right wheel

translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Front axle

translate([-wheelbase/2,0,0])axle(track=track);

// Rear axle

translate([wheelbase/2,0,0])axle(track=track);
```



*short body – small wheels

long_body = false;

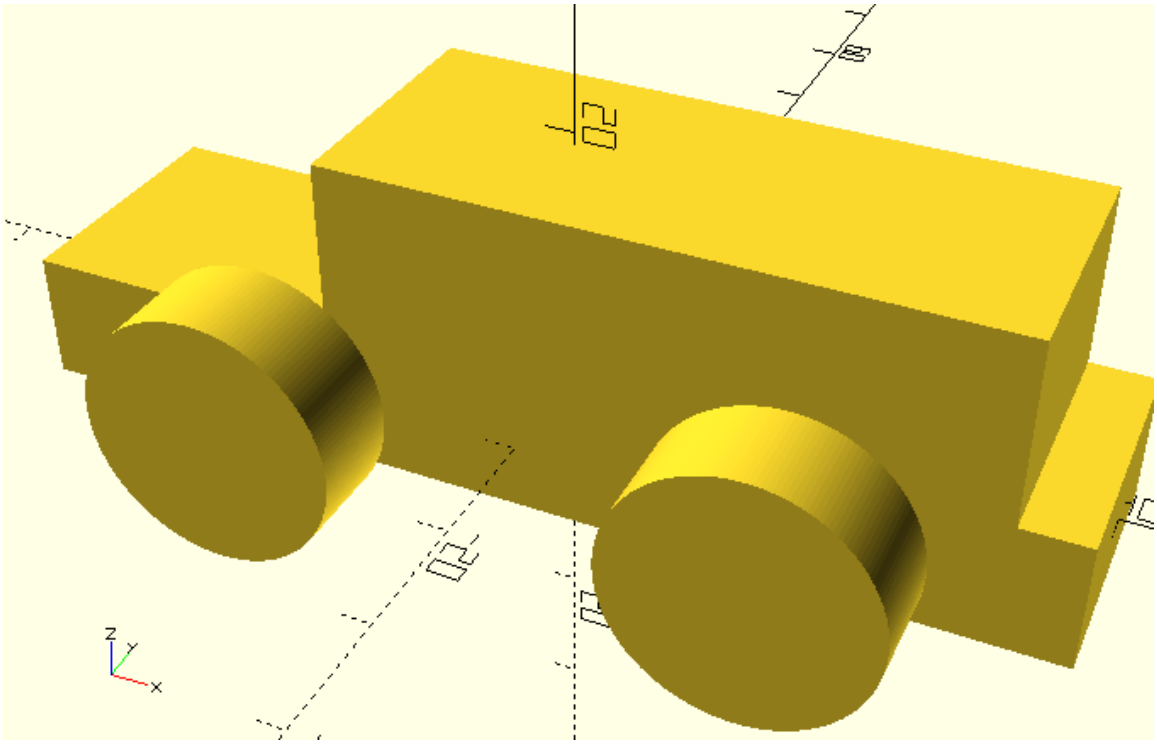
large_wheels = false;



*long body – large wheels

long_body = true;

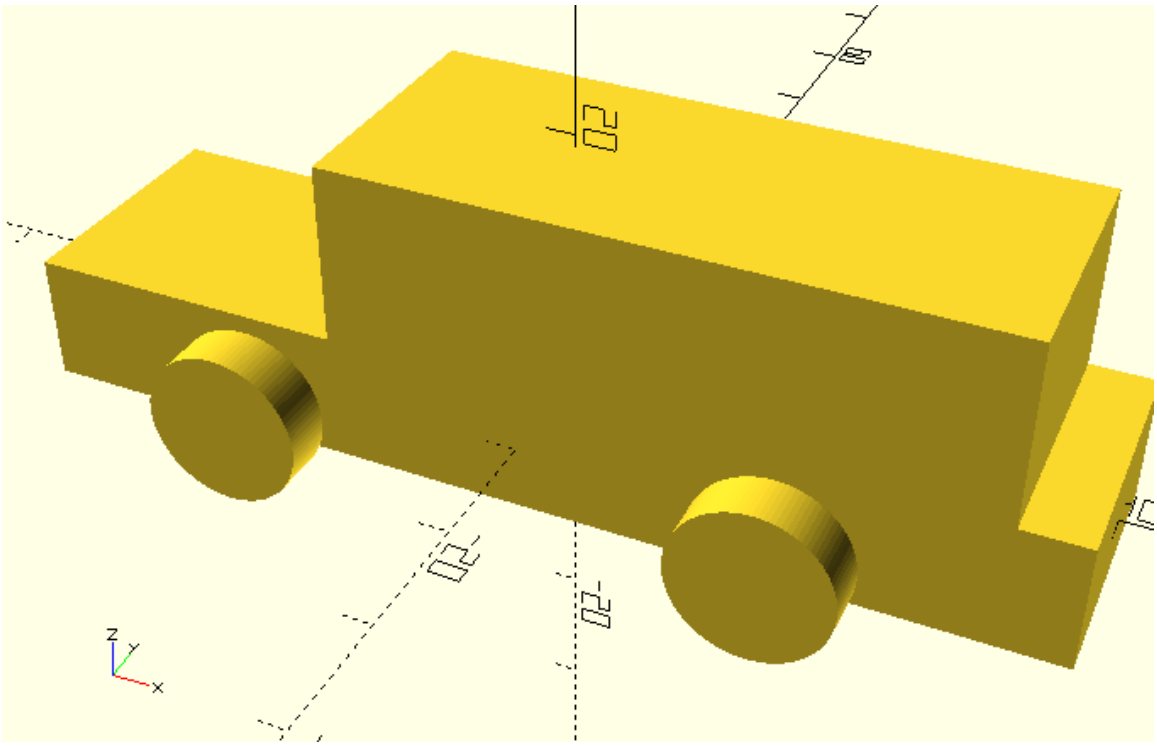
large_wheels = true;



*long body – small wheels

long_body = true;

large_wheels = false;



More conditional variable assignments

The conditional assignment of variables can also be used with a properly adjusted syntax when there are more than two cases between which you would like to choose. In the previous example there were only two options for the body (short or long) and a boolean variable (long_body) was used to choose between those two options.

What if you want to be able to choose between four versions of the body (short, long, rectangular and normal)? A boolean variable can't be used to represent your choice of body version since it can only have two values (true or false). For this reason, you are going to use a character to represent your choice of body.

The choice of a short body will be represented by the character s.

body_version = "s";

The choice of a long body will be represented by the character l.

body_version = "l";

The choice of a rectangular body will be represented by the character r.

body_version = "r";

The choice of a normal body will be represented by the character n.

body_version = "n";

The conditional assignments when there are more than two options should take the following form.

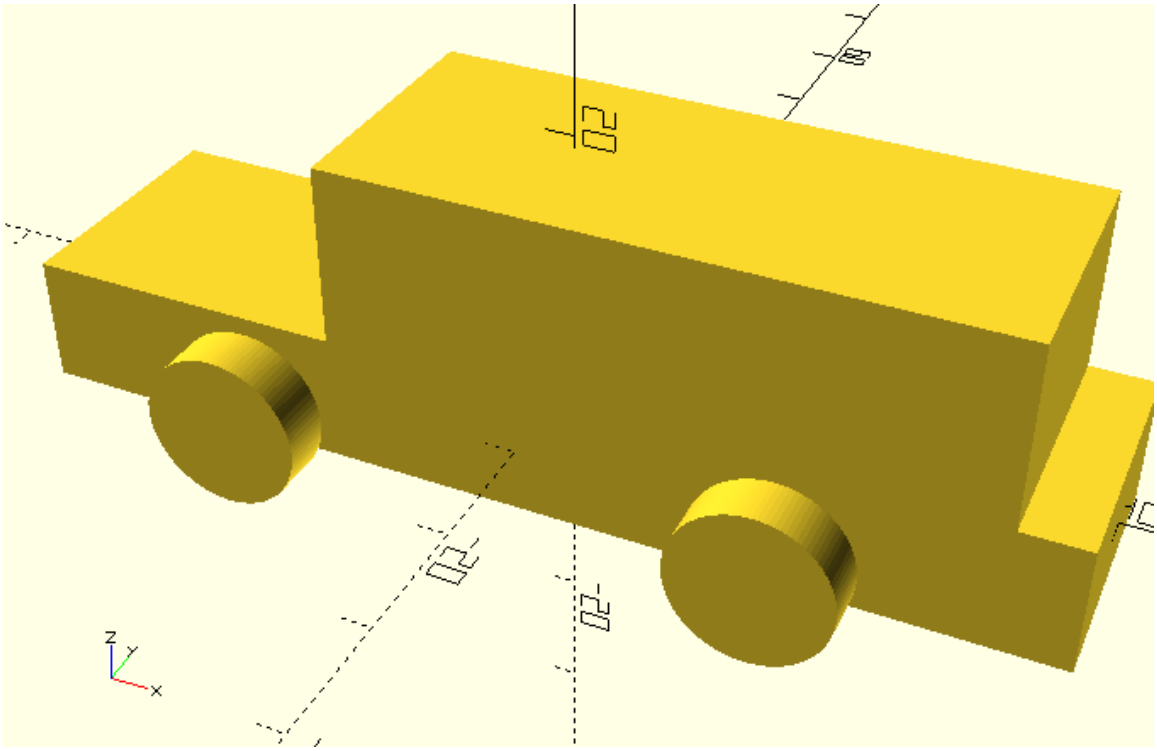
```
// base_length

base_length =

(body_version == "l") ? 80:

(body_version == "s") ? 60:

(body_version == "r") ? 65:70;


// top_length

top_length =

(body_version == "l") ? 50:

(body_version == "s") ? 30:

(body_version == "r") ? 65:40;


// top_offset

top_offset =

(body_version == "l") ? 10:

(body_version == "s") ? 5:

(body_version == "r") ? 0:7.5;
```

You should notice the following points about the definition of a conditional assignment when there are more than two options. First the name of the variable is typed out followed by the equal sign. Then follows a pair of parentheses that contains a condition, then a question mark, then the value to be assigned if the condition is true and then a colon. The previous sequence is repeated as required depending on the number of different available body versions. The last sequence is slightly different as it has an additional value which will be used as the default value when none of the conditions are true. In this case the default value corresponds to the normal version of the body. This is the reason why the character n that corresponds to the normal version of the body doesn't participate in any condition. Another thing you should notice is that the conditions are now comparison operations, specifically equality comparisons. If the value of the body_version variable is equal to the character that follows the double equal sign, then the condition is true and the corresponding value that follows the condition will be assigned to the variable.

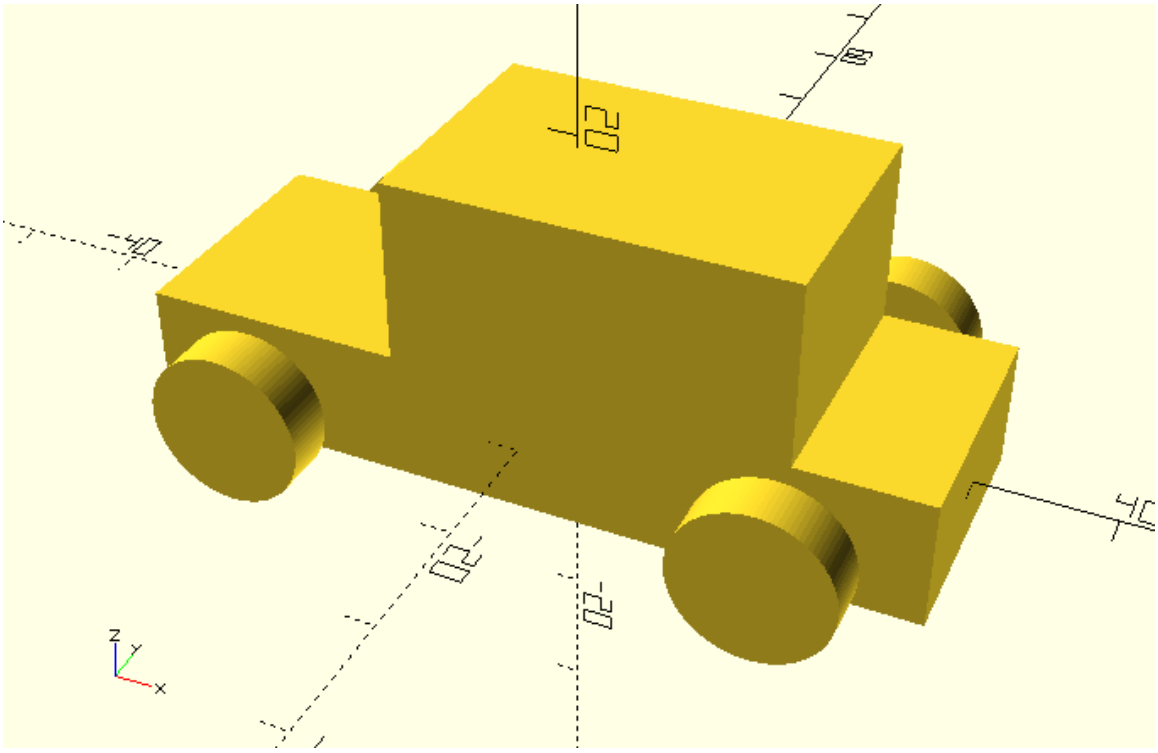
By incorporating the above conditional assignments in your script, you can switch between a short, a long, a rectangular and a normal car body just by setting the body_version equal to the character s, l, r or n respectively.

```
use <vehicle_parts.scad>

$fa=1;

$fs=0.4;


// Conditional assignment of body variables

body_version = "l";


// base_length

base_length =

(body_version == "l") ? 80:

(body_version == "s") ? 60:

(body_version == "r") ? 65:70;


// top_length

top_length =

(body_version == "l") ? 50:

(body_version == "s") ? 30:

(body_version == "r") ? 65:40;


// top_offset

top_offset =

(body_version == "l") ? 10:

(body_version == "s") ? 5:

(body_version == "r") ? 0:7.5;


// Creation of body
```

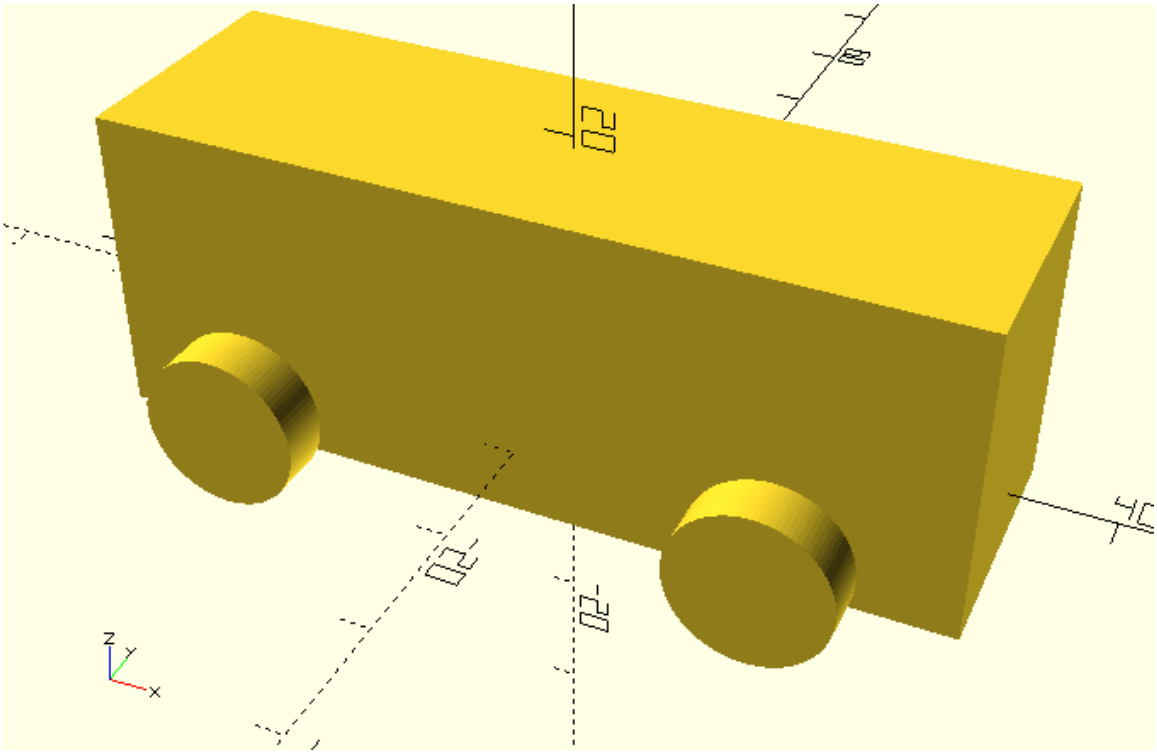
```
body(base_length=base_length, top_length=top_length, top_offset=top_offset);


// Creation of wheels and axles

large_wheels = false;

wheel_radius = (large_wheels) ? 10:6;

wheel_width = (large_wheels) ? 8:4;

track = 30;

wheelbase = 40;

// Front left wheel

translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

 // Front right wheel

translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear left wheel

translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear right wheel

translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Front axle

translate([-wheelbase/2,0,0])axle(track=track);

// Rear axle

translate([wheelbase/2,0,0])axle(track=track);
```



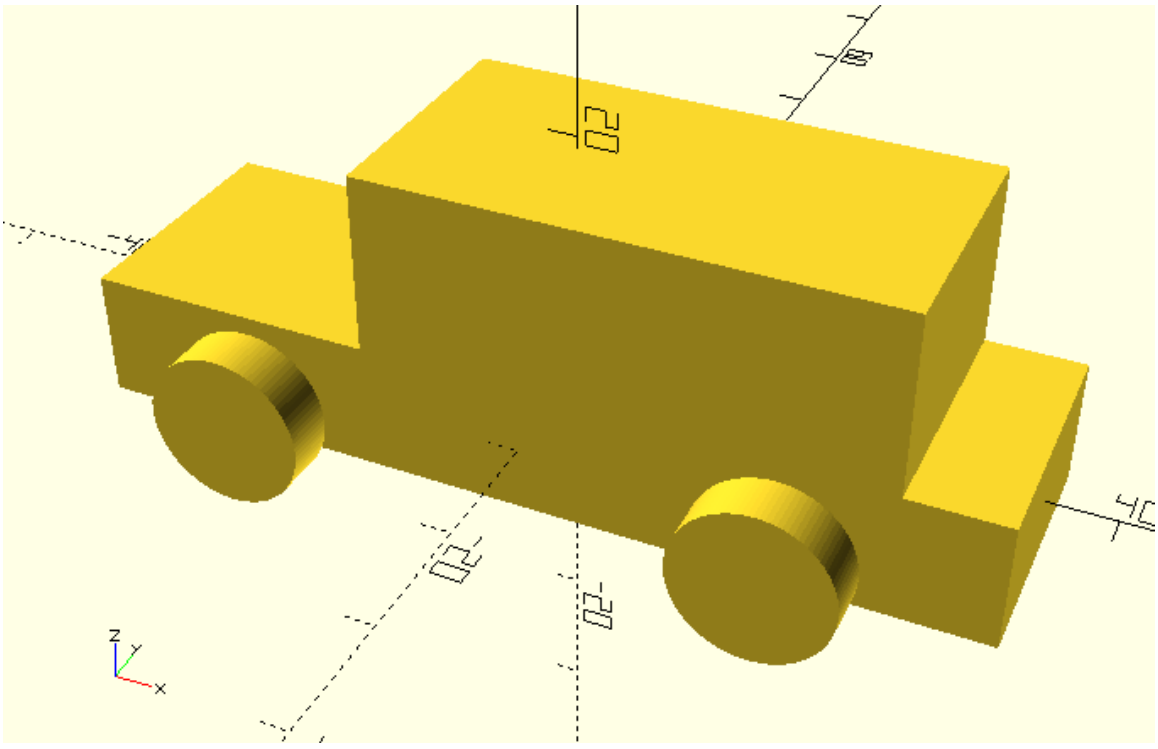
body_version = "s";



body_version = "r";



body_version = "n";



Add a wheels_version variable to the previous example. The variable should only take character values. Add appropriate conditional assignments that assign different values to the wheel_radius and wheel_width variables. The wheels_version variable should be used as the

condition for both assignments. If the value of the wheels_version variable is the character s (small), the wheel_radius and wheel_width variables should be set equal to 8 and 4 units respectively. If the value of the wheels_version variable is the character m (medium), the wheel_radius and wheel_width variables should be set equal to 9 and 6 units respectively. If the value of the wheels_version variable is the character l (large), the wheel_radius and wheel_width variables should be set equal to 10 and 8 units respectively. The case of the small version of the wheels should be used as the default case of the conditional assignments. Set appropriate values to the body_version and wheels_version variables to create the following versions of the car: short body – medium wheels, rectangular body – large wheels, normal body – small wheels.

*short body – medium wheels

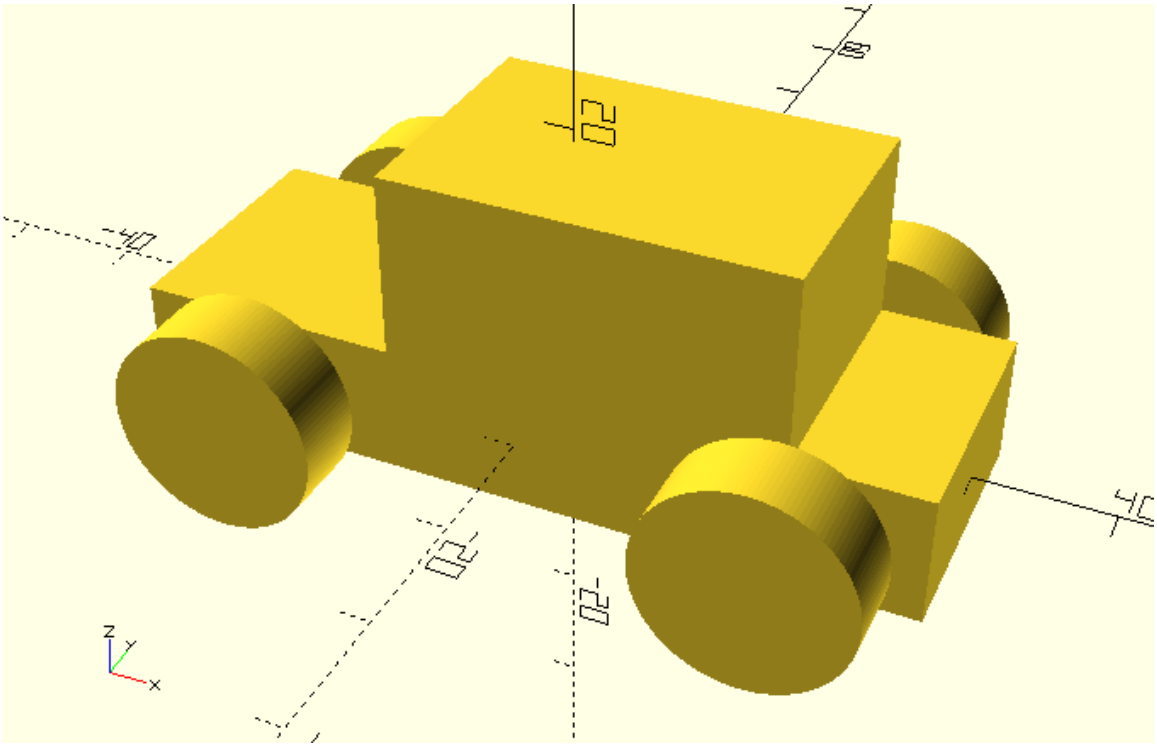
…

```
body_version = "s"; //s-short, n-normal, l-large, r-rectangular
```

…

```
wheels_version = "m"; //s-small, m-medium, l-large
wheel_radius =
(wheels_version == "l") ? 10:
(wheels_version == "m") ? 8:6;
wheel_width =
(wheels_version == "l") ? 8:
(wheels_version == "m") ? 6:4;
```

…



*rectangular body – large wheels

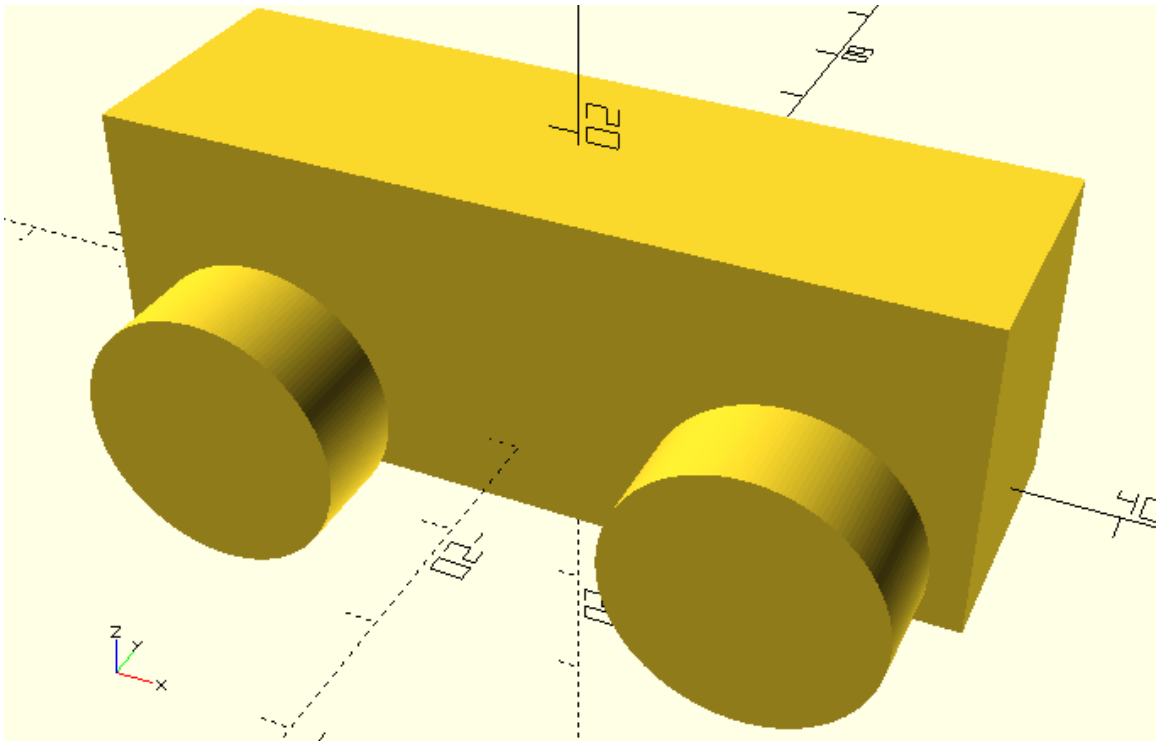
…

body_version = "r"; //s-short, n-normal, l-large, r-rectangular

…

wheels_version = "l"; //s-small, m-medium, l-large

…



*normal body – small wheels

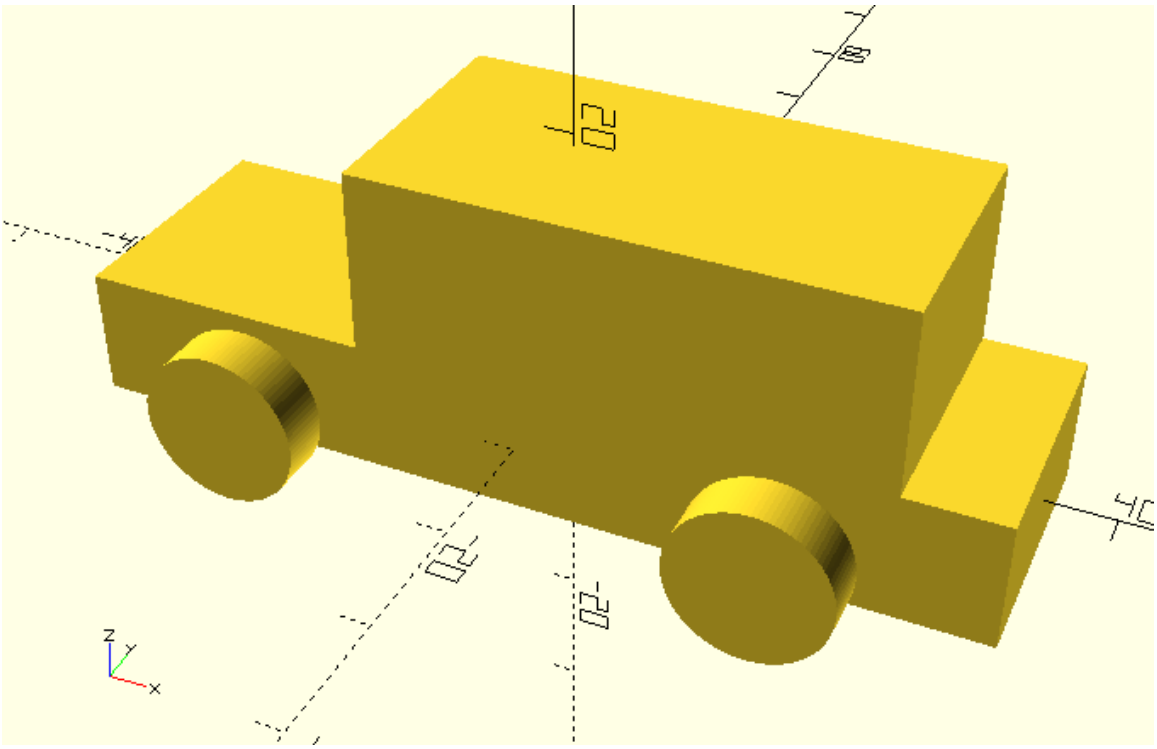
…

body_version = "n"; //s-short, n-normal, l-large, r-rectangular

…

wheels_version = "s"; //s-small, m-medium, l-large

…



Conditional creation of objects – If statement

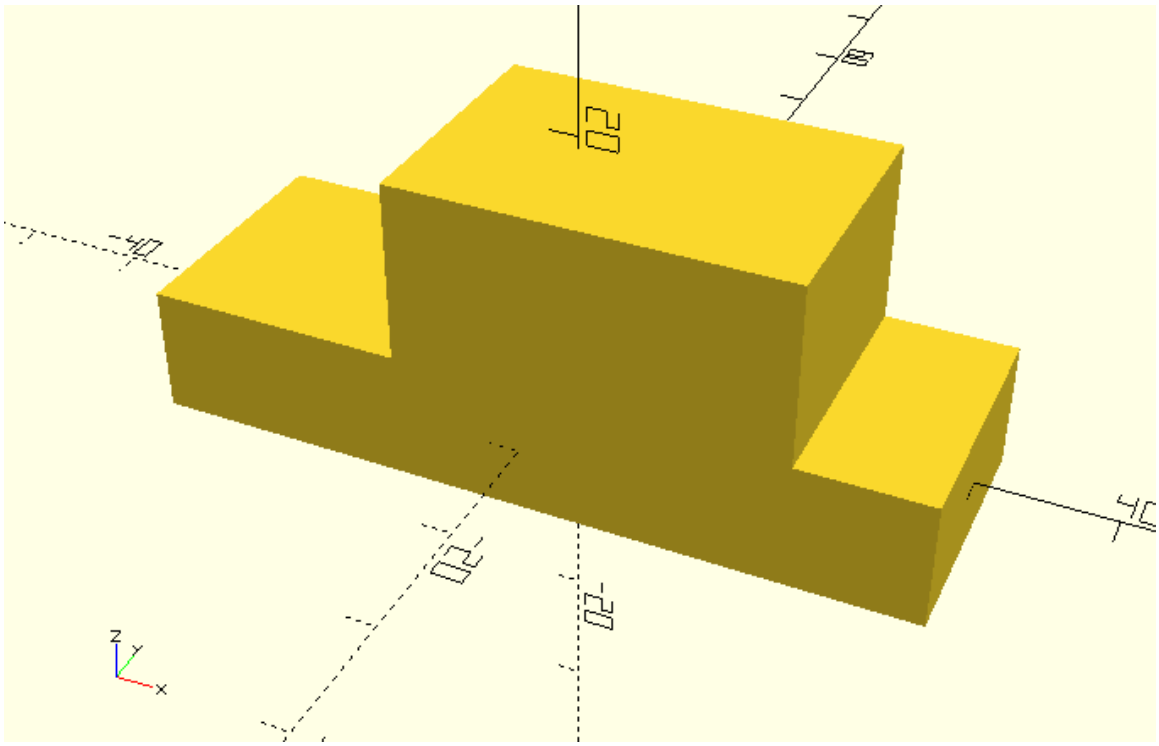
Conditional assignment of variables is a great tool to easily navigate between different but specific version of your model. Using conditional assignments, you were able to define different body and wheel sizes for your car and effortlessly choose between them without having to manually provide the values for all involved variables every single time.

What if you wanted to have the same control over the type of wheel (ex. simple, round, complex) or body (ex. square, round)? What would this require? In order to achieve this, you would need to have conditional creation of objects, which can be achieved with the use of the if statement.

Before you go into customizing the type of wheel and body, you can get familiar with if statements with some shorter examples. Bring into your mind for a moment the car body module that you created on a previous chapter. The module has some input parameters which are used to create two cubes, one cube for the body's base and one for the body's top.

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20, top_offset=5) {
    // Car body base
    cube([base_length,width,base_height],center=true);
    // Car body top
```

```
translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],center=true);

}

$fa = 1;

$fs = 0.4;

body();
```



Using an if statement you are going to see how the creation of the body's top can be parameterized. First you need to define an additional input parameter for the module. This parameter will be named top and will hold boolean values. If this parameter is false, the module should create only the base of the body. If it's true, it should also create the top of the body. This can be achieved by using an if statement in the following way.

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20, top_offset=5, top) {

  // Car body base

  cube([base_length,width,base_height],center=true);

  // Car body top

  if (top) {
```

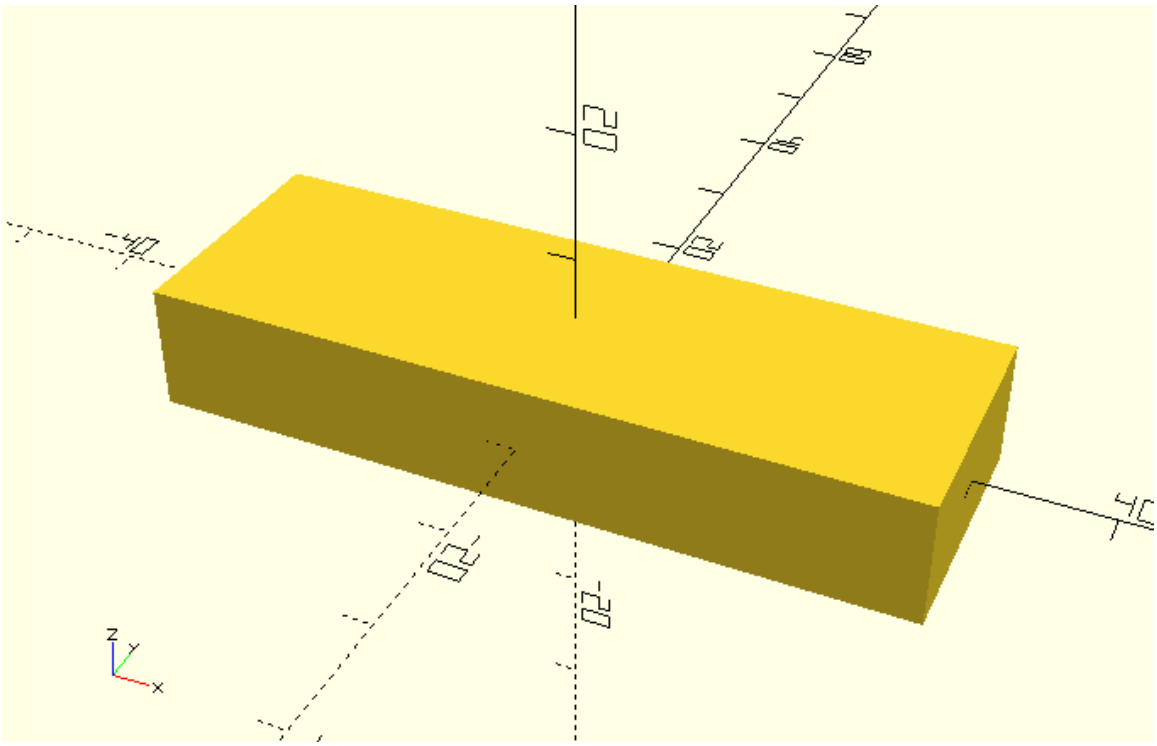
```
translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],center=true);

   }

}

$fa = 1;

$fs = 0.4;
```

You should notice the following points regarding the definition of the if statement. First the if keyword is typed out and then follows a pair of parentheses. Inside of the parentheses the condition that will dictate whether the if statement will be executed is defined. Lastly, there is a pair of curly brackets inside of which exist all statements that will be executed if the supplied condition is true. In this case the supplied condition is the boolean variable top, which represents your choice to create a car body that does or doesn't have a top part. The statement that is placed inside the curly brackets is the statement that creates the top part of the car's body.

The if statement at its current form is called simple if statement. This means that if the condition is true then the corresponding commands are executed, otherwise nothing happens. There are also two other forms of the if statement that you will later discover, but first check out how the new body module works in practice.

When the input parameter top is set to false, only the base of the body is created.

```
…

body(top=false);

…
```

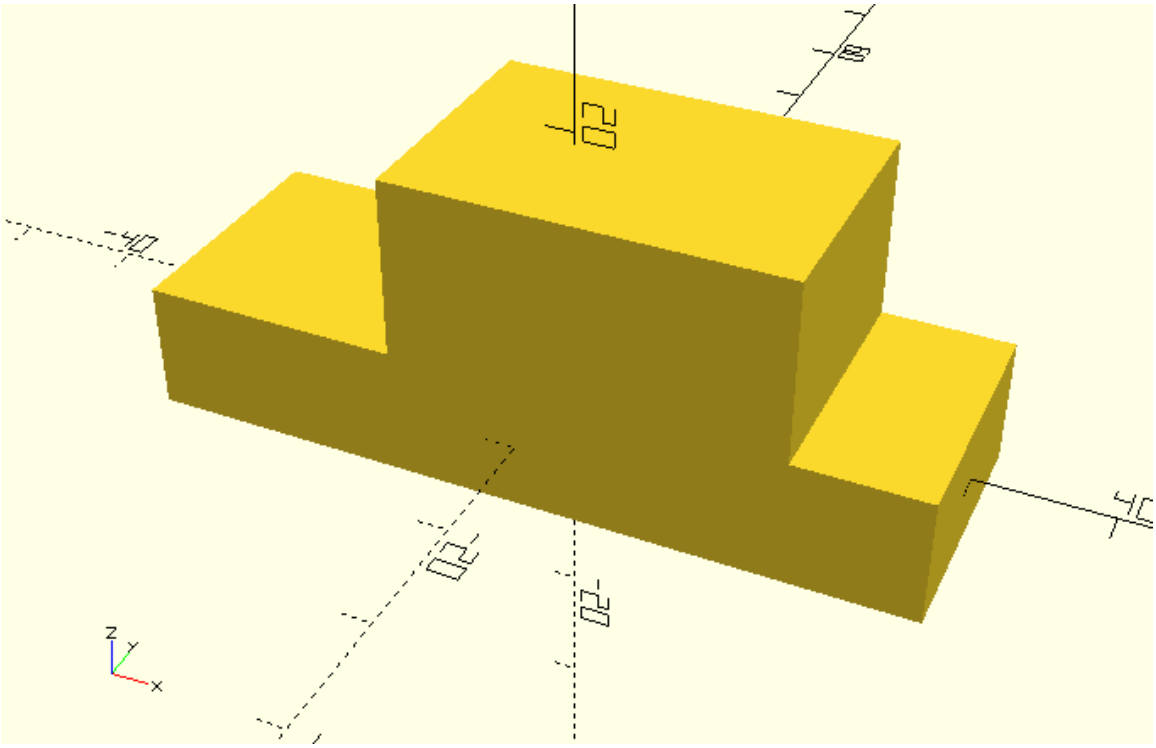


When it’s set to true, both the base and the top of the body are created.

…

body(top=true);

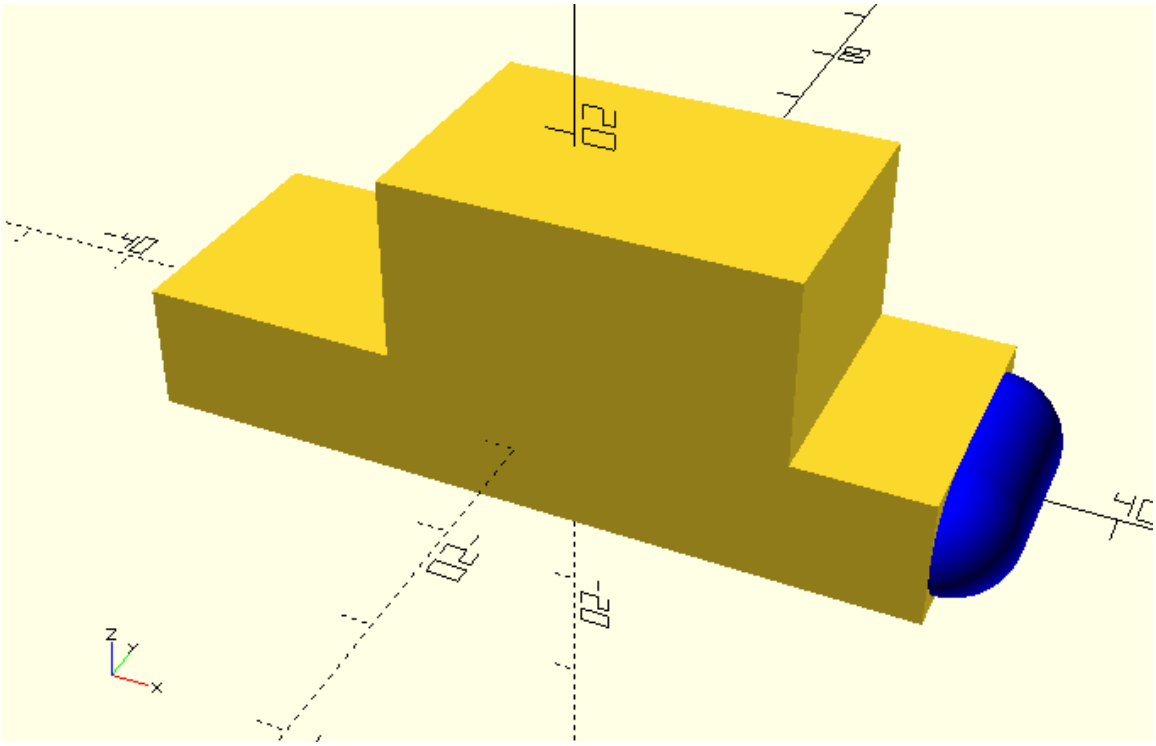
…



Take a look at the following car body module. You will notice that the module has been modified to include the creation of a front bumper. The color command that has been applied on the bumper has only a visual effect during preview. The color command is just used to draw your attention on the newly added part and shouldn't bother you any further than that.

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20,
top_offset=5, top) {
   // Car body base
   cube([base_length,width,base_height],center=true);
   // Car body top
   if (top) {

translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],cente
r=true);
   }
   // Rear bumper
   color("blue") {
      translate([base_length/2,0,0])rotate([90,0,0]) {
         cylinder(h=width - base_height,r=base_height/2,center=true);
```

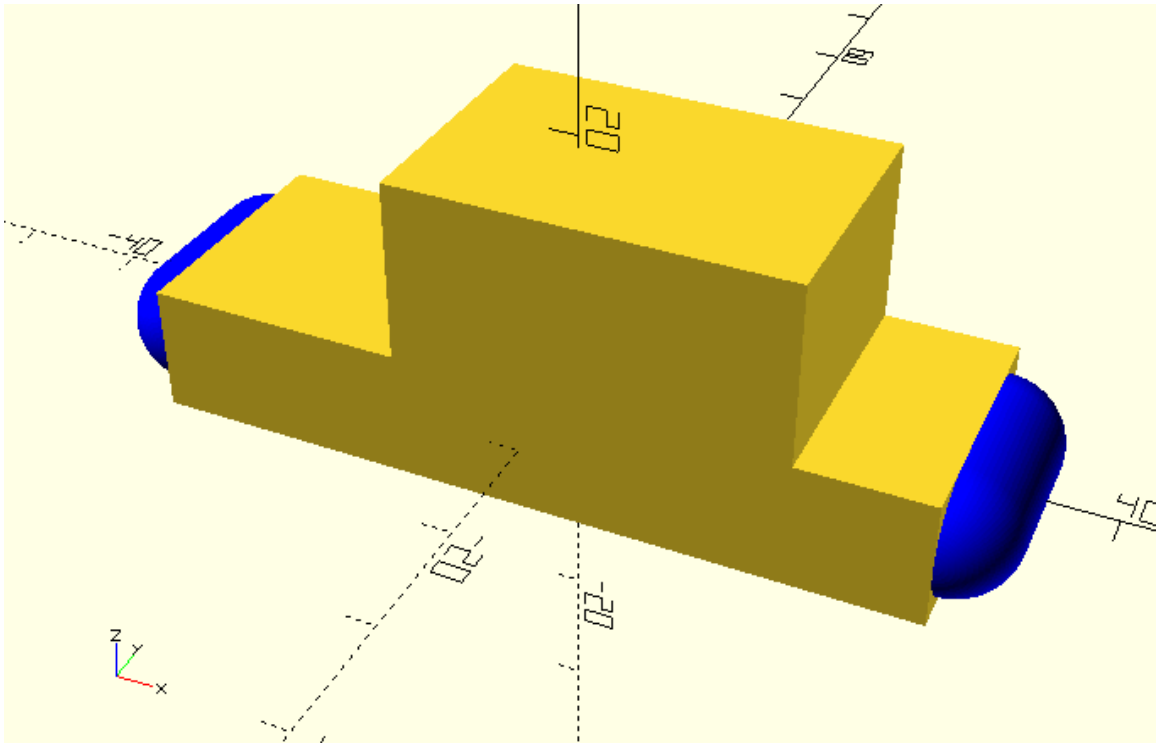
```
        translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
        translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
      }
   }
}
$fa = 1;
$fs = 0.4;
body(top=true);
```



Copy and paste inside the body module the statements that create the rear bumper. Modify accordingly the appropriate translation statement to create the front bumper too.

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20,
top_offset=5, top) {
   // Car body base
   cube([base_length,width,base_height],center=true);
   // Car body top
   if (top) {
```

```
translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],center=true);
    }
    // Front bumper
    color("blue") {
        translate([-base_length/2,0,0])rotate([90,0,0]) {
            cylinder(h=width - base_height,r=base_height/2,center=true);
            translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
            translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
        }
    }
    // Rear bumper
    color("blue") {
        translate([base_length/2,0,0])rotate([90,0,0]) {
            cylinder(h=width - base_height,r=base_height/2,center=true);
            translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
            translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
        }
    }
}
$fa = 1;
$fs = 0.4;
body(top=true);
```

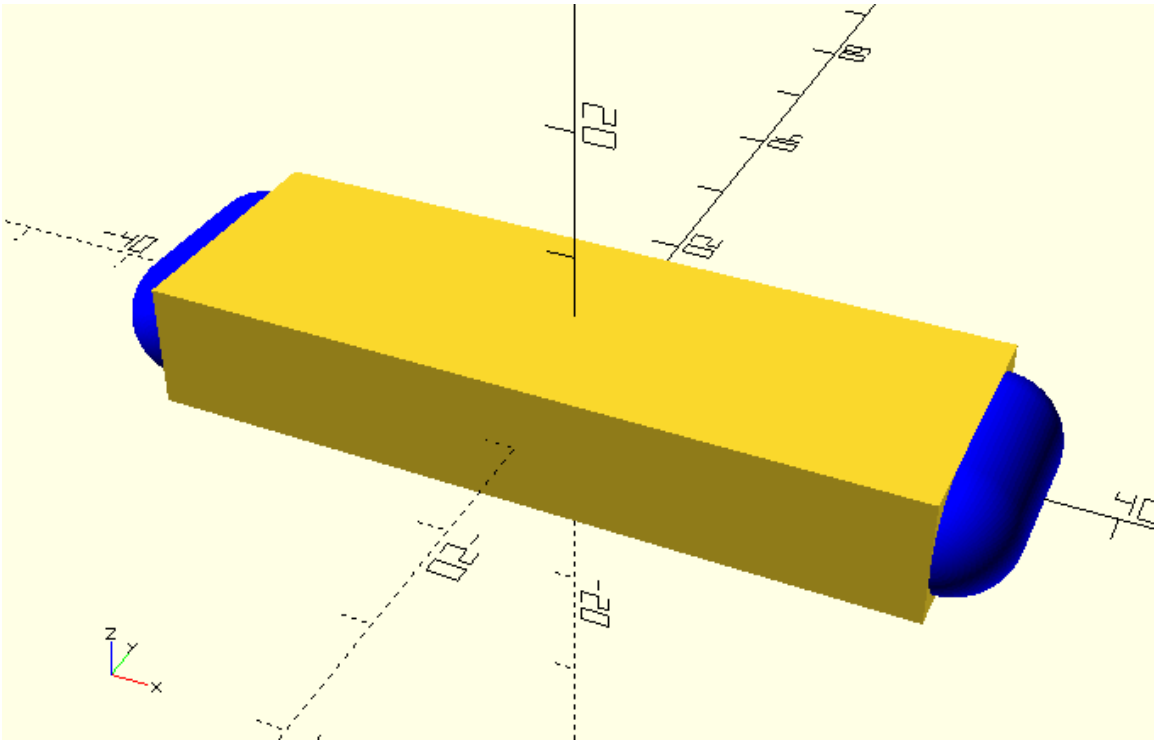


Define two additional input parameters for the body module. One named front_bumper and one named rear_bumper. Keeping in mind that these parameters should take boolean values define two if statements that conditionally create the front and rear bumpers. The front bumper should be created only if the front_bumper input parameter is true, the second bumper accordingly. Use the body module to create the following car bodies: base only – front and rear bumper, base and top – front bumper, base and top – front and rear bumper.

*base only – front and rear bumper

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20, top_offset=5, top, front_bumper, rear_bumper) {
   // Car body base
   cube([base_length,width,base_height],center=true);
   // Car body top
   if (top) {

translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],center=true);
   }
   // Front bumper
```

```
    if (front_bumper) {
        color("blue") {
            translate([-base_length/2,0,0])rotate([90,0,0]) {
                cylinder(h=width - base_height,r=base_height/2,center=true);
                translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
                translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
            }
        }
    }
    // Rear bumper
    if (rear_bumper) {
        color("blue") {
            translate([base_length/2,0,0])rotate([90,0,0]) {
                cylinder(h=width - base_height,r=base_height/2,center=true);
                translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
                translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
            }
        }
    }
}
$fa = 1;
$fs = 0.4;
body(top=false,front_bumper=true,rear_bumper=true);
```

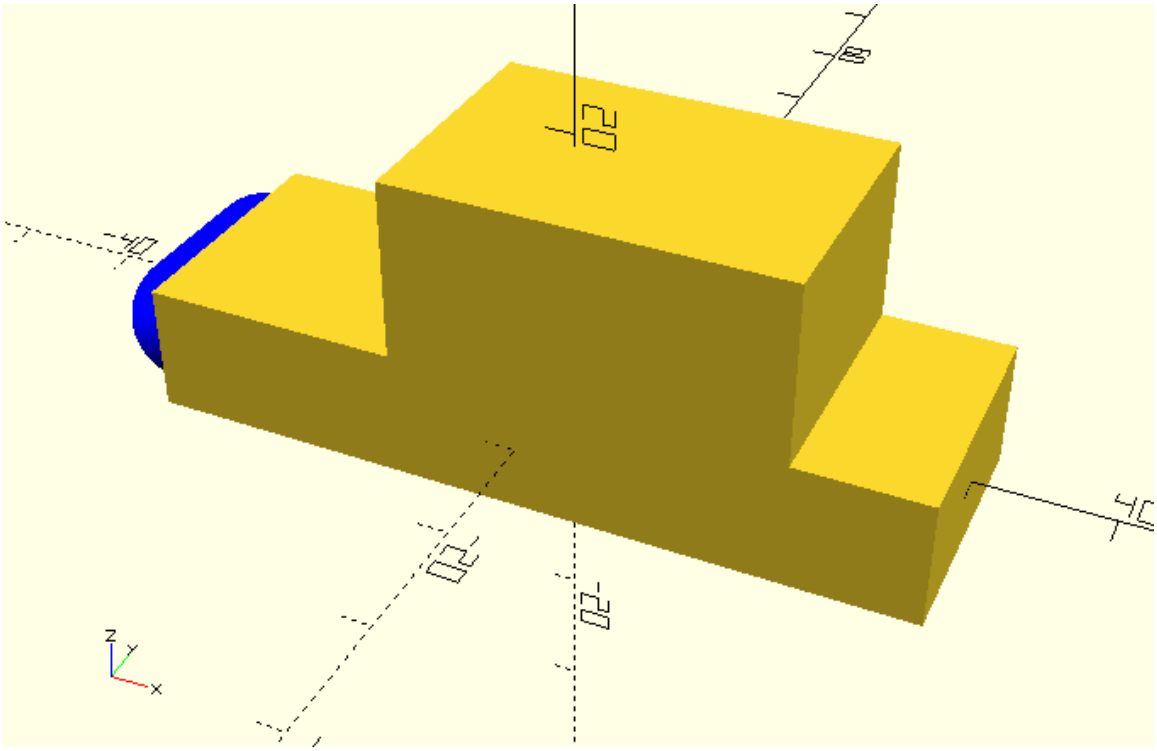


*base and top – front bumper

…

body(top=true,front_bumper=true,rear_bumper=false);

…

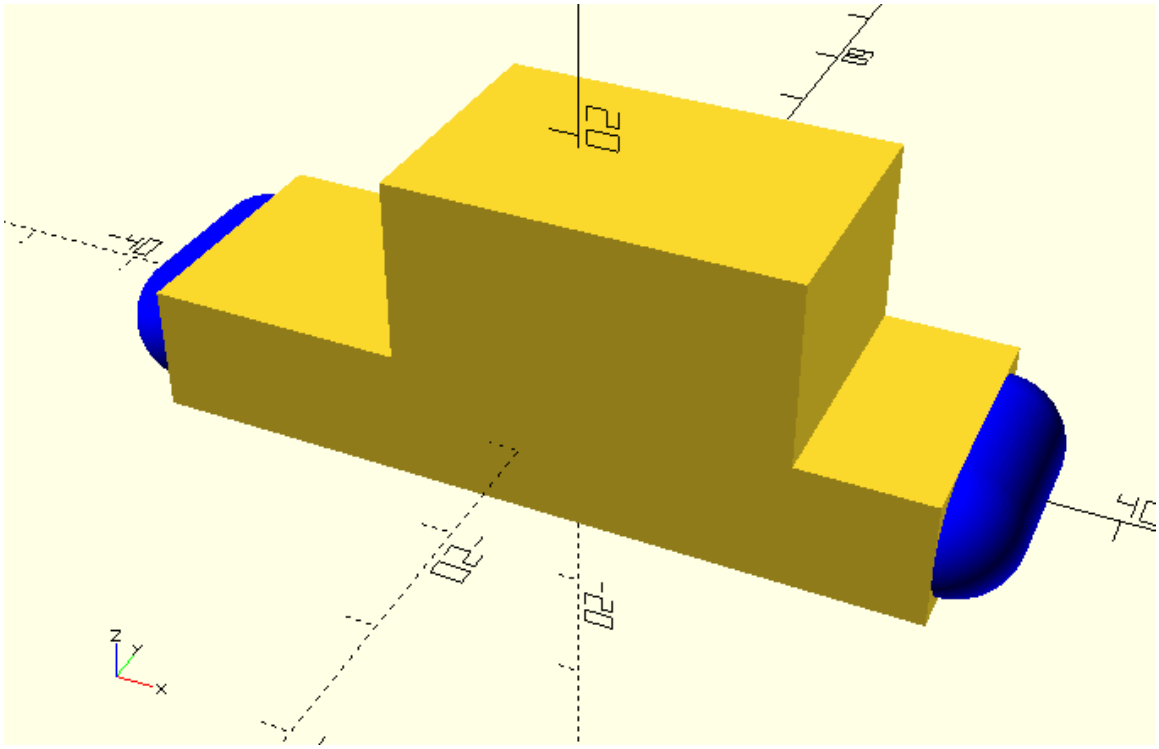


*base and top – front and rear bumper

…

body(top=true,front_bumper=true,rear_bumper=true);

…



Challenge

In this chapter you learned about conditional assignment of variables and simple if statements. Specifically, you learned how to conditionally modify dimensions and transformations of parts of your designs as well as how to conditionally include or exclude parts from them. It's time to put these two together in a single car model.

If you have been following along this tutorial you should have a vehicle_parts.scad script on your machine from a previous chapter. Open this script and update the body module according to the last example so that it has the ability to conditionally create the top of the body as well as a front and a rear bumper. Set default values for the newly added input parameters. Specifically set true, false and false as the default value of the top, front_bumper and rear_bumper variable accordingly. Save the changes and close the script.

…

```
module body(base_height=10, top_height=14, base_length=60, top_length=30, width=20,
top_offset=5, top=true, front_bumper=false, rear_bumper=false) {

  // Car body base

  cube([base_length,width,base_height],center=true);

  // Car body top

  if (top) {
```

```
translate([top_offset,0,base_height/2+top_height/2])cube([top_length,width,top_height],center=true);
  }
  // Front bumper
  if (front_bumper) {
    color("blue") {
      translate([-base_length/2,0,0])rotate([90,0,0]) {
        cylinder(h=width - base_height,r=base_height/2,center=true);
        translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
        translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
      }
    }
  }
  // Rear bumper
  if (rear_bumper) {
    color("blue") {
      translate([base_length/2,0,0])rotate([90,0,0]) {
        cylinder(h=width - base_height,r=base_height/2,center=true);
        translate([0,0,(width - base_height)/2])sphere(r=base_height/2);
        translate([0,0,-(width - base_height)/2])sphere(r=base_height/2);
      }
    }
  }
}
```

…

Given the following script that creates a car, make appropriate additions and modifications to the script in order to parameterize conditionally the design of the car. Specifically, you will need to define a body_version, a wheels_version, a top, a front_bumper and a rear_bumper variable that will be used for making design choices regarding the car's design. If necessary, review the previous examples and exercises of this chapter to remember what effect these

variables should have on the design of the car and how to implement them. Use the resulting script to create a version of the car that you like.

*Given script

```
use <vehicle_parts.scad>
$fa=1;
$fs=0.4;

// Variables
track = 30;
wheelbase=40;

// Body
body();
// Front left wheel
translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel();
 // Front right wheel
translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel();
// Rear left wheel
translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel();
// Rear right wheel
translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel();
// Front axle
translate([-wheelbase/2,0,0])axle();
// Rear axle
translate([wheelbase/2,0,0])axle();
```

*Modified script

```
use <vehicle_parts.scad>
```

```
$fa=1;
$fs=0.4;

// Variables
body_version = "l"; //s-short, n-normal, l-large, r-rectangular
wheels_version = "l"; //s-small, m-medium, l-large
top = true;
front_bumper = true;
rear_bumper = true;
track = 30;
wheelbase=40;

// Conditional assignments
// Body: base_length
base_length =
(body_version == "l") ? 80:
(body_version == "s") ? 60:
(body_version == "r") ? 65:70;

// Body: top_length
top_length =
(body_version == "l") ? 50:
(body_version == "s") ? 30:
(body_version == "r") ? 65:40;

// Body: top_offset
top_offset =
(body_version == "l") ? 10:
(body_version == "s") ? 5:
```

```
(body_version == "r") ? 0:7.5;


// Wheels: radius

wheel_radius =

(wheels_version == "l") ? 10:

(wheels_version == "m") ? 8:6;


// Wheels: width

wheel_width =

(wheels_version == "l") ? 8:

(wheels_version == "m") ? 6:4;


// Body

body(base_length=base_length, top_length=top_length, top_offset=top_offset, top=top,
front_bumper=front_bumper, rear_bumper=rear_bumper);

// Front left wheel

translate([-wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

 // Front right wheel

translate([-wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear left wheel

translate([wheelbase/2,-track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Rear right wheel

translate([wheelbase/2,track/2,0])rotate([0,0,0])simple_wheel(wheel_radius=wheel_radius,
wheel_width=wheel_width);

// Front axle

translate([-wheelbase/2,0,0])axle();

// Rear axle
```

translate([wheelbase/2,0,0])axle();