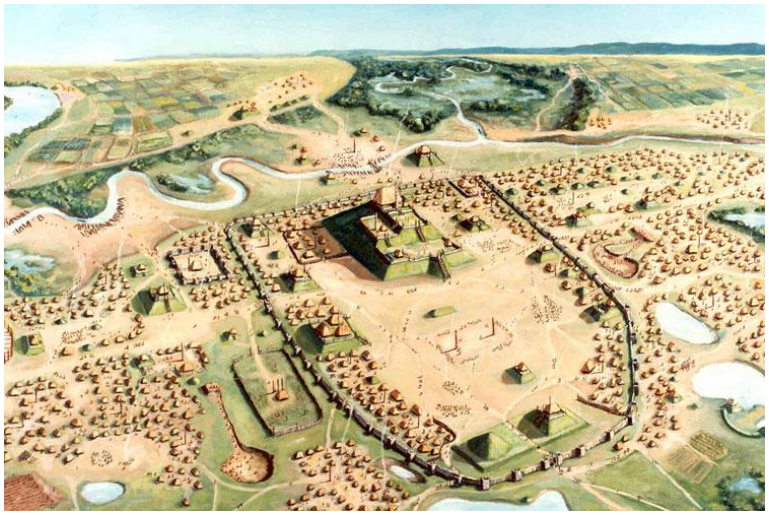
Scavenger Hunt

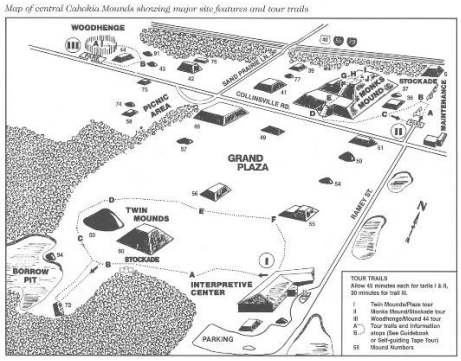
Step 1: Develop the map

- Have at least 12 places on the map. Decide how they are connected.
- Your place can be real or made-up. It can be large or small. It can be set in the future, present or past.
- You will need a starting point and an ending point (or winning location)



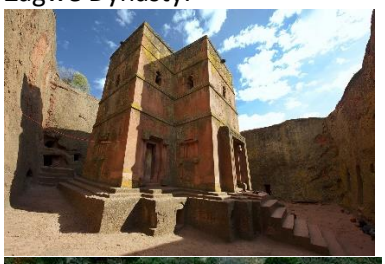
The City of Cahokia

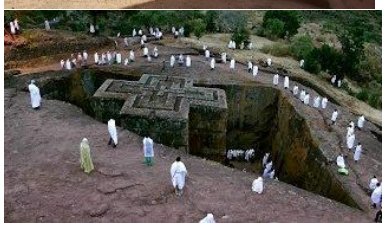
The ancient indigenous metropolis, located on the banks of a northern section of the Mississippi. It had 40,000 people at its 1050 BCE peak.

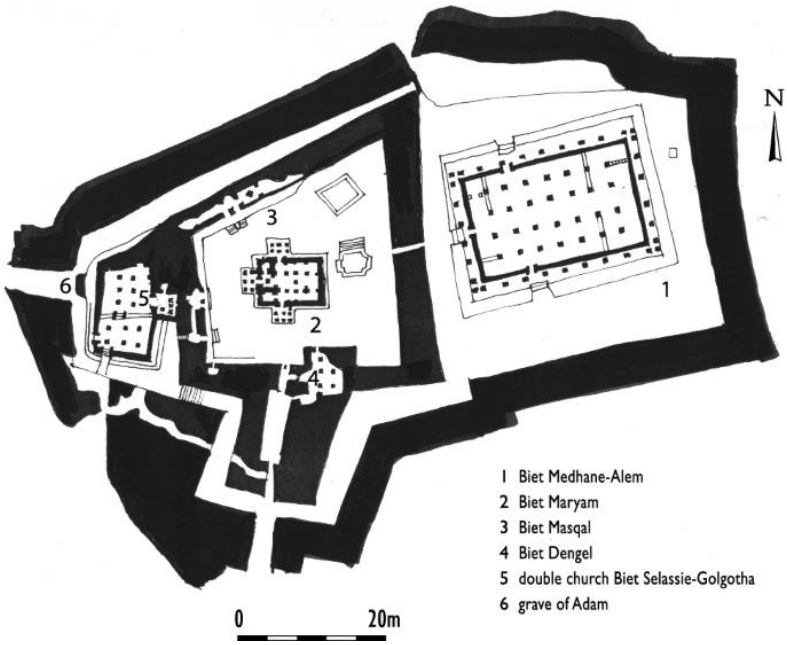


Rock-Hewn Churches of Ethiopia

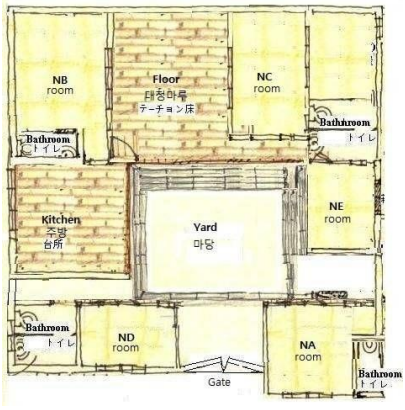
The eleven Rock-hewn Churches of Lalibela were built in the late-12th and early-13th century, commissioned by King Gebre Mesqel Lalibela of the Zagwe Dynasty.





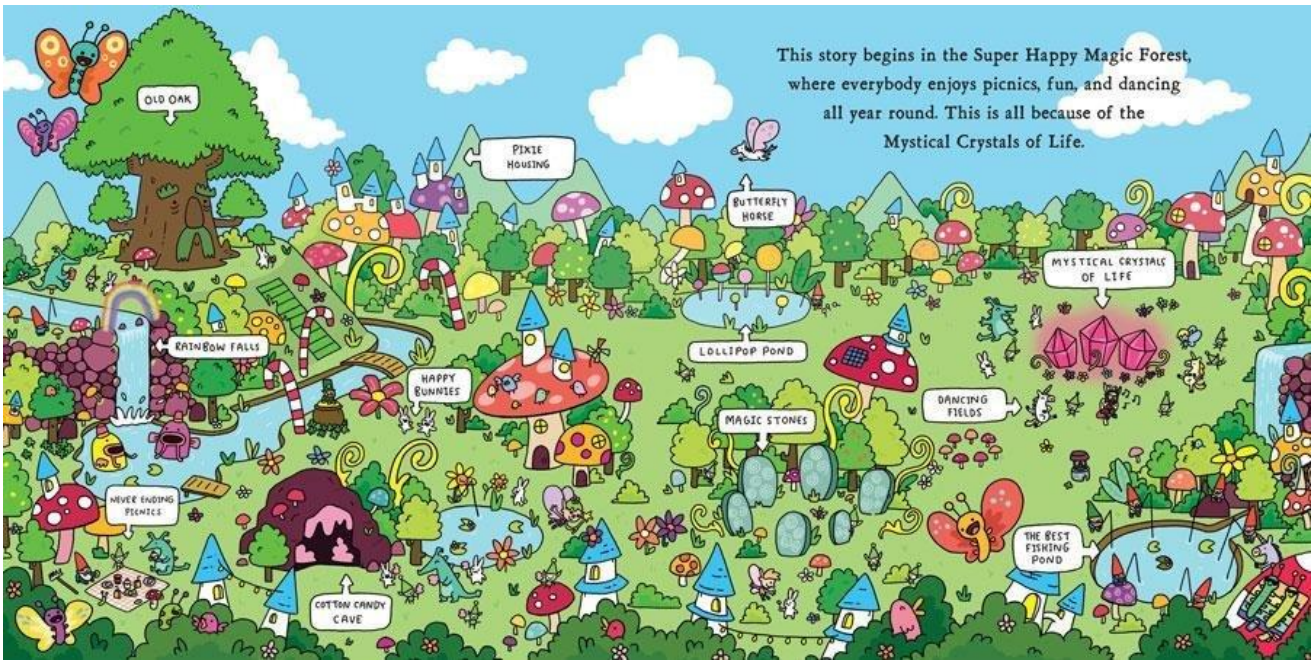


Alternatively, you can just make up your own map of whatever location you want.



Step 2: Develop the code.

For example:



This example program is set in the Super Happy Magic Forest, where everybody enjoys picnics, fun, and dancing all year round (as stated above).

As this student did, make a method for each place in your map.

The OldOak method is highlighted in the code next to this description.

Check that your code runs. It won't do anything, but it should run.

Show Ms. Gorski when you are done this step.

```
public class SuperHappyMagicForest {
    public static void main(String args[]) {
        new SuperHappyMagicForest();
    }
    public SuperHappyMagicForest() {
        //You will fill this in later on!
    }
    public void OldOak() {
        //You will fill this in later on!
    }
    public void PixieHousing() {
        //This would need to be filled in!
    }
    public void HappyBunnies() {
        //You will fill this in later on!
    }
    public void RainbowFalls() {
        //You will fill this in later on!
    }
    public void ButterflyHorse() {
        //You will fill this in later on!
    }
    public void LollipopPond() {
        //You will fill this in later on!
    }
    public void DancingFields() {
        //You will fill this in later on!
    }
    public void MysticalCrystalsOfLife() {
        //You will fill this in later on!
    }
    public void MagicStones() {
        //You will fill this in later on!
    }
    public void CottonCandyCave() {
        //You will fill this in later on!
    }

    public void NeverEndingPicnic() {
        //You will fill this in later on!
    }
}
```

Step 3: Begin coding the map

- Begin at the starting point in your adventure.
- Code it together.
- This is an example of a starting point.
- Note that it calls the next method, sending the user to the next place.

```
public class SuperHappyMagicForest {
   public static void main(String args[]) {
       new SuperHappyMagicForest();
   }
   public SuperHappyMagicForest() {
       System.out.println("Welcome to the SUPER HAPPY MAGIC FOREST!!!\n");
       System.out.println("Here everybody enjoys picnics, fun, and dancing all year round.");
       System.out.println("Enjoy the rainbows, sunshine and good times.\n");
       OldOak();
   }

   public void OldOak() {
       System.out.println("You have found the OLD OAK.");
       System.out.println("If it is wisdom to you seek, well, you might have some luck.");
       System.out.println("It is the wisest oak tree in the whole forest.");
       System.out.println("If you just want acorns, there's a bunch of those too. \n");

       System.out.println("Where would you like to go next?");
       System.out.println("(p) Pixie Housing");
       System.out.println("(h) Happy Bunnies");
       System.out.println("(r) Rainbow Falls");

       char choice = IO.inputChar("\nYour choice? (p/h/r) ");
       System.out.println();
       if(choice =='p')
        PixieHousing();
       else if (choice =='h')
        HappyBunnies();
       else
        RainbowFalls();
   }
```

Step 4: Divide the methods.

- Divide the remaining methods between the two of you.
- Each of you should code the methods. Keep talking together about the story as you code.
- Put the methods together when you are done.
- Test to make sure you can visit all of the locations.
- Make sure your code runs.
- Show Ms. Gorski when you are done this step.

Pickups

- You can declare global variables as shown below. If the user has been to a location where the pick-up is, you can increment the counter.
- Add in three items that your user needs to find in the map.
- When they get to the winning location, you can check that they have picked up all of the items they need to exit the code using an if.
- If they have all of the items, then they win. Otherwise, give them a hint as to want they still need to find.

```
public class SuperHappyMagicForest {

   int acorn=0;

   public static void main(String args[]) {
       new SuperHappyMagicForest();
   }
   public SuperHappyMagicForest() {
       System.out.println("Welcome to the SUPER HAPPY MAGIC FOREST!!!\n");
       System.out.println("Here everybody enjoys picnics, fun, and dancing all year round.");
       System.out.println("Enjoy the rainbows, sunshine and good times.\n");
       OldOak();
   }

   public void OldOak() {
       System.out.println("You have found the OLD OAK.");
       System.out.println("If it is wisdom to you seek, well, you might have some luck.");
       System.out.println("It is the wisest oak tree in the whole forest.");
       System.out.println("If you just want acorns, there's a bunch of those too. \n");
       acorn++;

       System.out.println("Where would you like to go next?");
       System.out.println("(p) Pixie Housing");
       System.out.println("(h) Happy Bunnies");
       System.out.println("(r) Rainbow Falls");

       char choice = IO.inputChar("\nYour choice? (p/h/r) ");
       System.out.println();
       if(choice =='p')
        PixieHousing();
       else if (choice =='h')
        HappyBunnies();
       else
        RainbowFalls();
   }
```

Let Ms. Gorski know when you have completed this code.

Reset

- Let the user choose to play again.
- Bring them back to the start of the map.
- Reset all of the pickup variables to zero.

```
acorn=0;
```

- Also, add some sort of timer to note how long it takes for the students to complete the program. This is an example:

  http://www.gorskicompsci.ca/JavaNotes/ExtraFeatures/TimingAnEvent.pdf

Let Ms. Gorski know when you have completed this code.