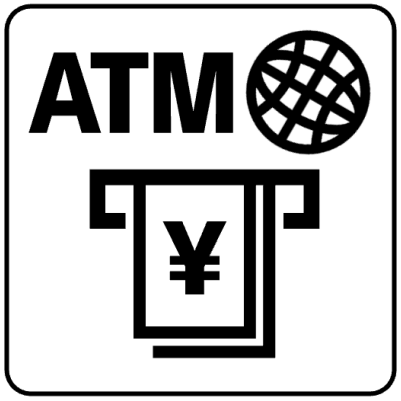
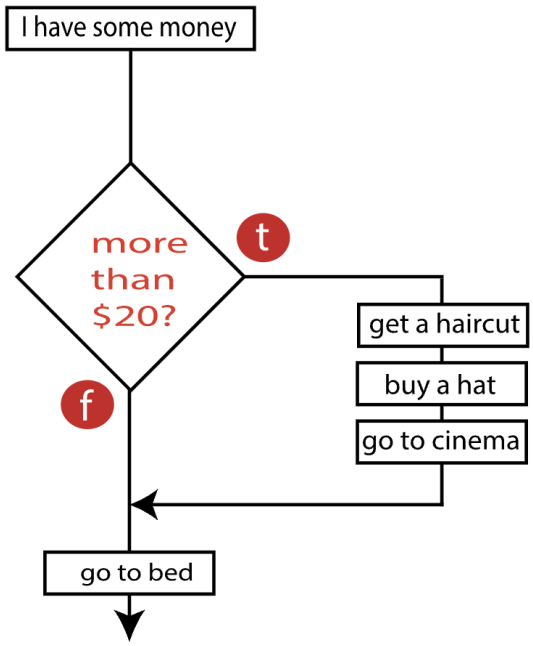
ATM SIMULATOR



A Smart ATM allows you to withdraw money. It also recommends things that you can do depending on the amount you withdraw. For example, the following flowchart helps us to visualise this process:



Let's code this smart ATM Simulator.

Task 1

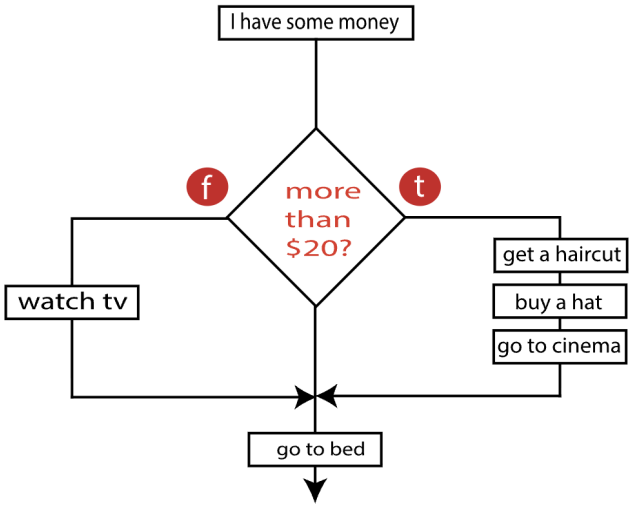
Copy the starter code into your IDE.

Analyze/Run the code and complete the table:

How many functions are in the program?	
How many variables are in the program?	
Which function is called first when we execute the program?	
What is the output when we enter **15** as the amount?	

Task 2

The logic of the simulator has changed. The new logic is:



The decision box now has 2 paths, a **true** path and a **false** path. In many programming languages we can think of this as **if...else...**

In Java the syntax is:

```
if(condition is true){
      //do something
}
else{
      //do something else
}
```

Adapt the code in the simulator so that:

```
if the amount is greater than 20, we get a haircut, buy a
hat, go to the cinema

else we watch tv
```

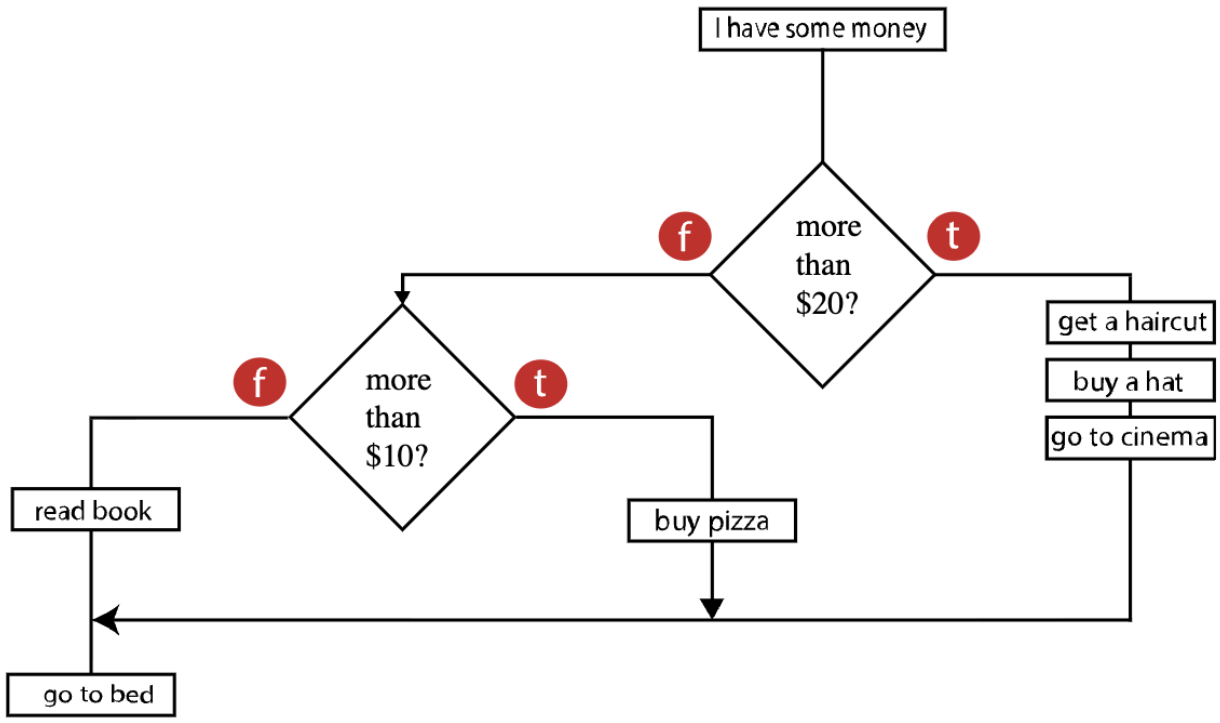
Test your code by entering different amounts. Do you get the expected result?

Complete the table:

Input amount	output
10	
30	

Task 3

The logic of the simulator has changed again. The new logic is:



This time there are 2 decision boxes.
The first one checks if the amount is greater than 20.
If it isn't, another decision box checks if the amount is greater than 10.
If it isn't, then there is one final option, **read a book**.

In this scenario there are more than 1 decision box. Our code needs to evaluate both of them. In Java, we can do this:

```
if(condition1 is true){
     //do something
}else if(condition2 is true){
     //do something else
}else{
     //none of the above conditions are true, so do this!
}
```

This is a pretty funky chain of if...else... statements. Can you see how it works?

Look at the flowchart above and see if you can adapt the atmSimulator function to follow this logic.

Test your code using this table:

Input amount	Expected output
25	Get a haircut Buy a hat Go to the cinema Go to bed
15	Buy pizza Go to bed
5	Read a book Go to bed

Does your code work as expected?

Task 4

If the amount is greater than 20 the **bigSpender()** function is called

Else if the amount is greater than 10 the **mediumSpender()** function is called

Else if when the amount is greater than 5 the **smallSpender()** function is called

Else the **broke()** function is called.

Redesign the program to take advantage of these functions.

You can use your own ideas to code these functions.

Make sure you **call** them when the atmSimulator needs them.

Note on comparison operators in Java

The code in this unit explores if statements. If statements evaluate conditions eg:

```
if(amount > 20)
```

This statement uses the ***greater than*** operator, >

Here are other common comparison operators used by Java:

>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to[1]
!=	Not equal to

Submit your .java file and this worksheet as instructed

[1] Note: = is an assignment operator used to assign data to a variable; == is a comparison operator used to determine if 2 values are the same