Homework 2 - Trick or Treating

You have decided that halloween isn't too childish, even for great programmers such as yourselves. You DREAD the idea of getting up from your chair, making a costume, and worst of all - going outside though. Thus, you decide you are just going to write a VB program to simulate halloween's best part, the candy.

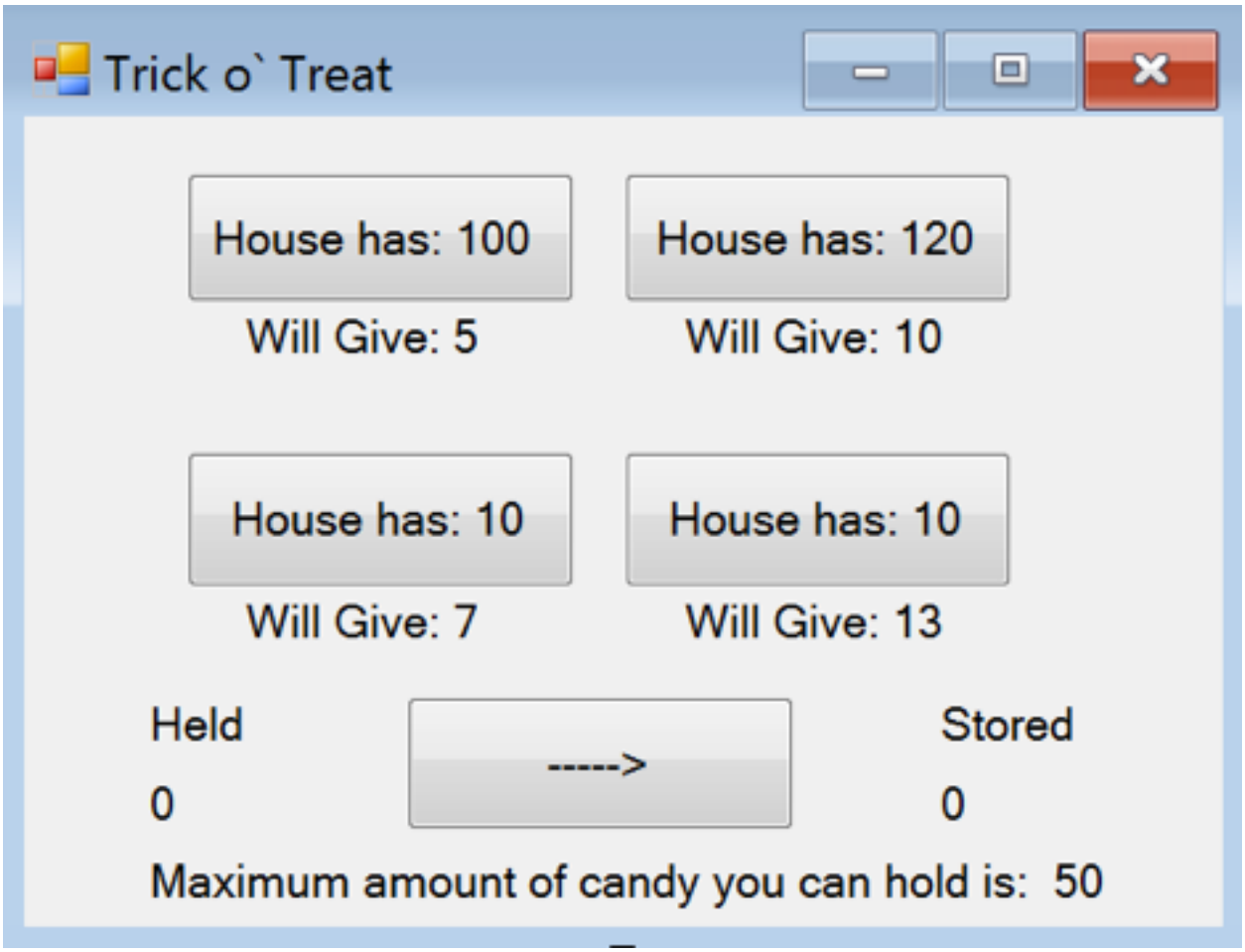
Your program is going to simulate the process of going from house to house trick or treating collecting candy. Each of the four houses will be a separate button on a form and you "visit" it by simply clicking on the button. Each house has its own set amount of stored candy; Again, this is not one total pool of candy, it is split by house. When you visit a house they will give you part of their candy, make sure that they actually **have** this candy to give you.

Additionally you can only hold a limited amount of candy but you can click a separate button to drop off your candy at home. Whenever you are about to receive candy make sure you have the room to accept it.

This homework reviews the concepts of variables, scope, conditional statements, and use of variable to keep track of states. Your held candy and your stored candy should be implemented with global/module level variables. Each of the houses' candy pools can EITHER be implemented as static variables or as global/module level variables. Do not allow the candy to go negative or create candy out of nothing.

Consider this case: Under the usual guidelines of this homework, if a house has 20 candy left and they usually give away 50 candies, they would not give away their last 20 candies. For ***extra credit***, make houses give away the last of their candy.

Here is a sample of what your form could look like:



(You can make it fancier if you want with images or Halloween colors. Do **not** make it garish or hard to read with bad color choices or messy/distracting images)

Static Variable Example

Note: this is not required for the homework but is a much cleaner way to do it. As a general rule you should avoid global variables whenever possible.

If you want to use static variables for your homework to keep track of the house's candy pools this example will help you understand what a static variable is and how it differs from a normal variable.

Copy and paste the following into the button click sub procedure for a button named "btnExample".

```
Dim myNormalCounter As Integer = 100
myNormalCounter = myNormalCounter - 1
btnExample.Text = Convert.toString(myNormalCounter)
```

Try to click the button a few times. You will notice that the button will show 99 forever. Now try changing the "Dim" to either "Static Dim" or just "Static" and click the button.

```
Static myNormalCounter As Integer = 100
myNormalCounter = myNormalCounter - 1
btnExample.Text = Convert.toString(myNormalCounter)
```

Now it will count down from 99 one at a time. Why?

A normal variable has a lifetime (called it's scope). When the sub procedure that created the variable ends, the variable is destroyed. The next time the sub is called the variable is created again with its default value (in this case 100).

A static variable is just like a normal variable with one crucial change. It's lifetime is extended, it will never be reinitialized. The first time you call the sub the variable is created and initialized to the value specified. The next time you call the function, the variable still exists and will not be reset/reinitialized to the value specified.

It is important to see that static variables are similar to module level variables in that they are only created once. However they are different in that you can only access them inside of their scope, unlike module level variables which can be accessed anywhere. This is sometimes useful in that you can use the same name in different places to refer to different variables, just like local variables. If any of this confuses you, come to office hours!