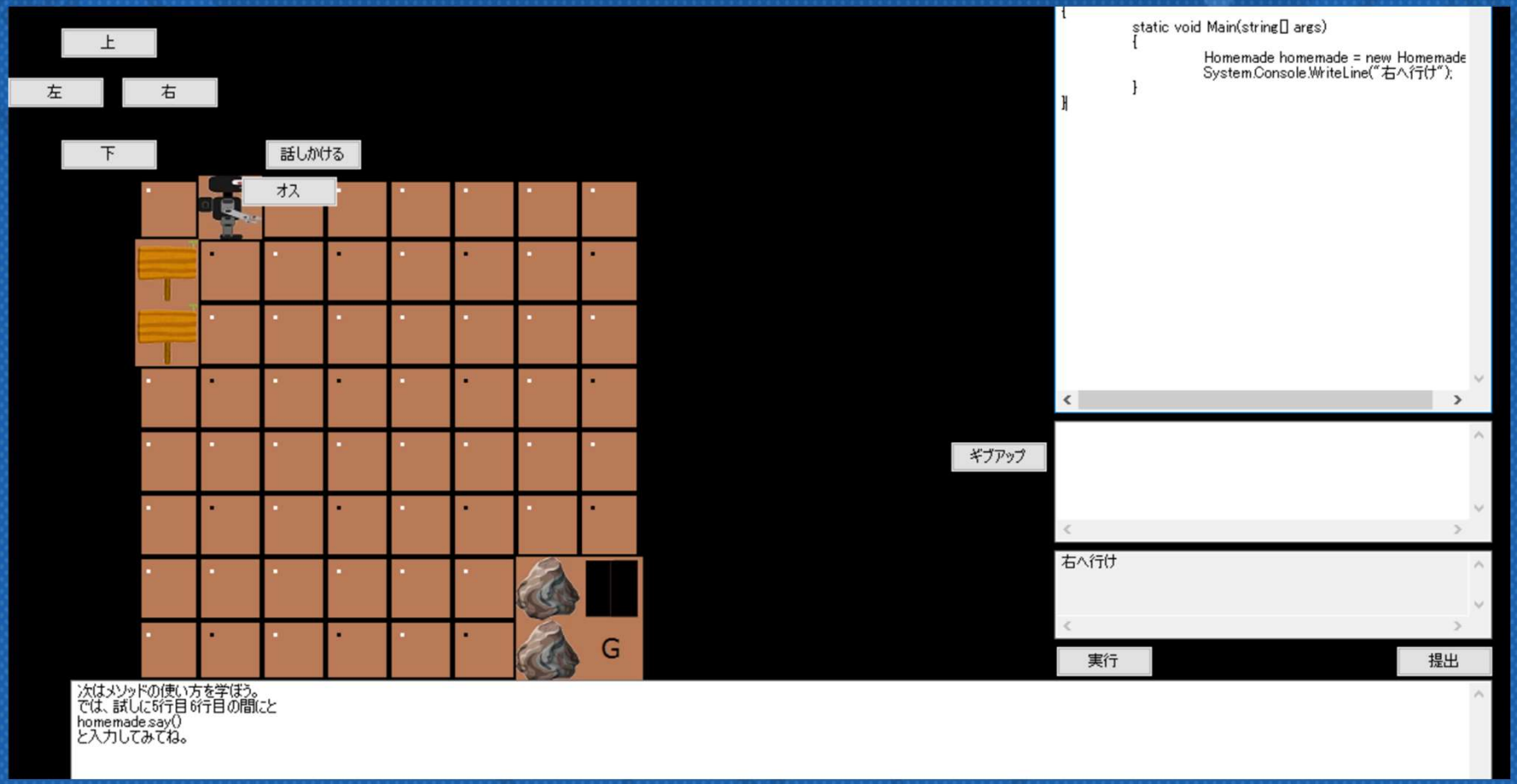


チーム名：kuwax
作品タイトル「プログラミングで楽しむRPG」

ターゲット

プログラミングを軽くやったことがあり、プログラミング言語C#の学習をしたい人をターゲットにしている。

プレイ人数

windowsパソコンを用いて一人で行うゲーム

ゲームの目的

C#のコードで命令を与えて、主人公をゴールまでたどり着かせることが目的。
その過程で、ユーザーはプログラミングを学べる。

ゲームクリア・勝利条件

主人公がゴールにたどり着くことがゲームクリアの条件。

ゲームオーバー条件

「ギブアップ」ボタンを押せば、そのステージをあきらめることになるため、ゲームオーバーになる。
スイッチを押すメソッドで、スイッチのない場所でそのメソッドを使用したらゲームオーバーとなるものもある。

開発進捗

完成品である。

備考・注意点

画像などのグラフィックが改善の余地がある。

操作説明

※足りない場合は、ご自身で操作説明のページを追加してください。

初めの画面で矢印キーでステージを選んで、エンターキーでそのステージの中に入れる。
以降は、指示があるようにc#のコードを書くことで主人公を操作して、ゴールにたどり着くようにする。
人によっては、もうすでに習得済みなところのコードを書きたくないかもしれないので、
「上」「右」「下」「左」「押す」「話しかける」などのボタンを用意している。このボタンを押せばコードを書かなくても主人公を操作できる。
「ギブアップ」を選択したら初めの画面に戻る。

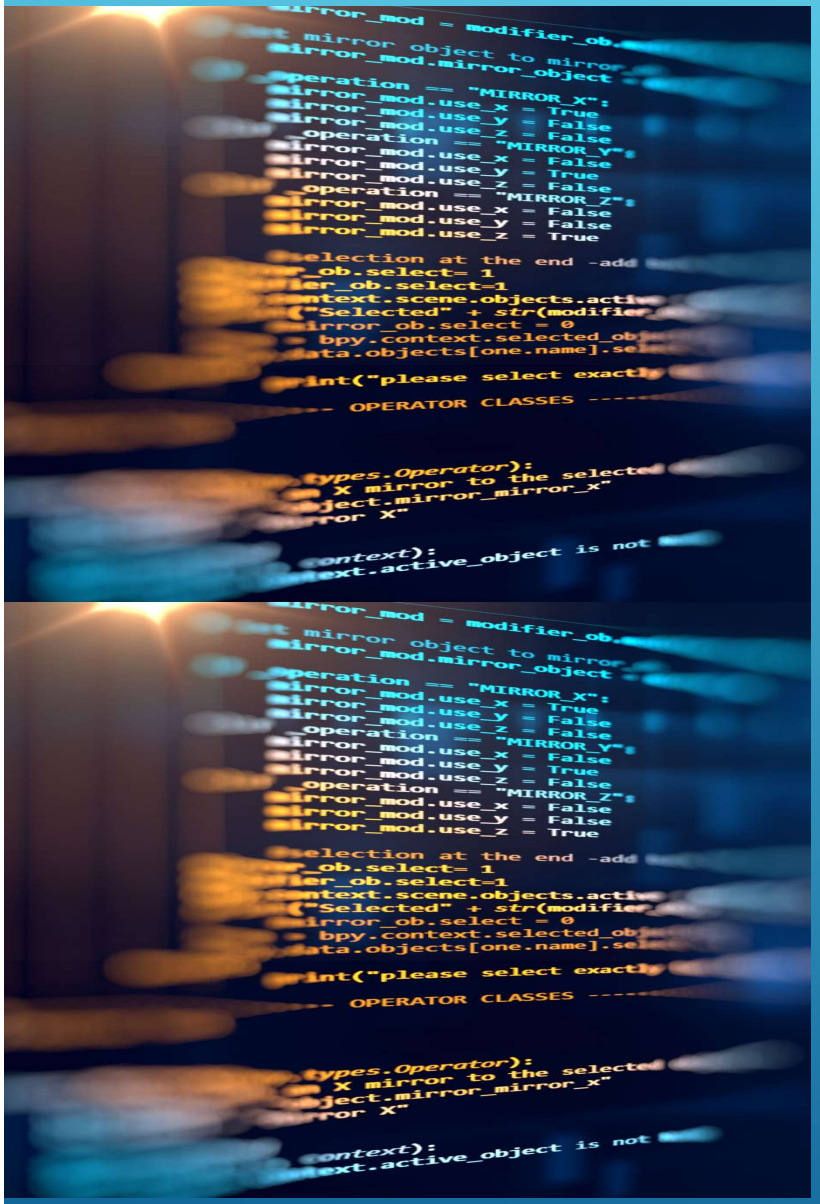
スタッフリスト欄

※足りない場合は、ご自身でページを追加してください。

・桑原頌明
①全般
②全般

以降のページに
企画書を追加してください

プログラミングで楽しむRPG



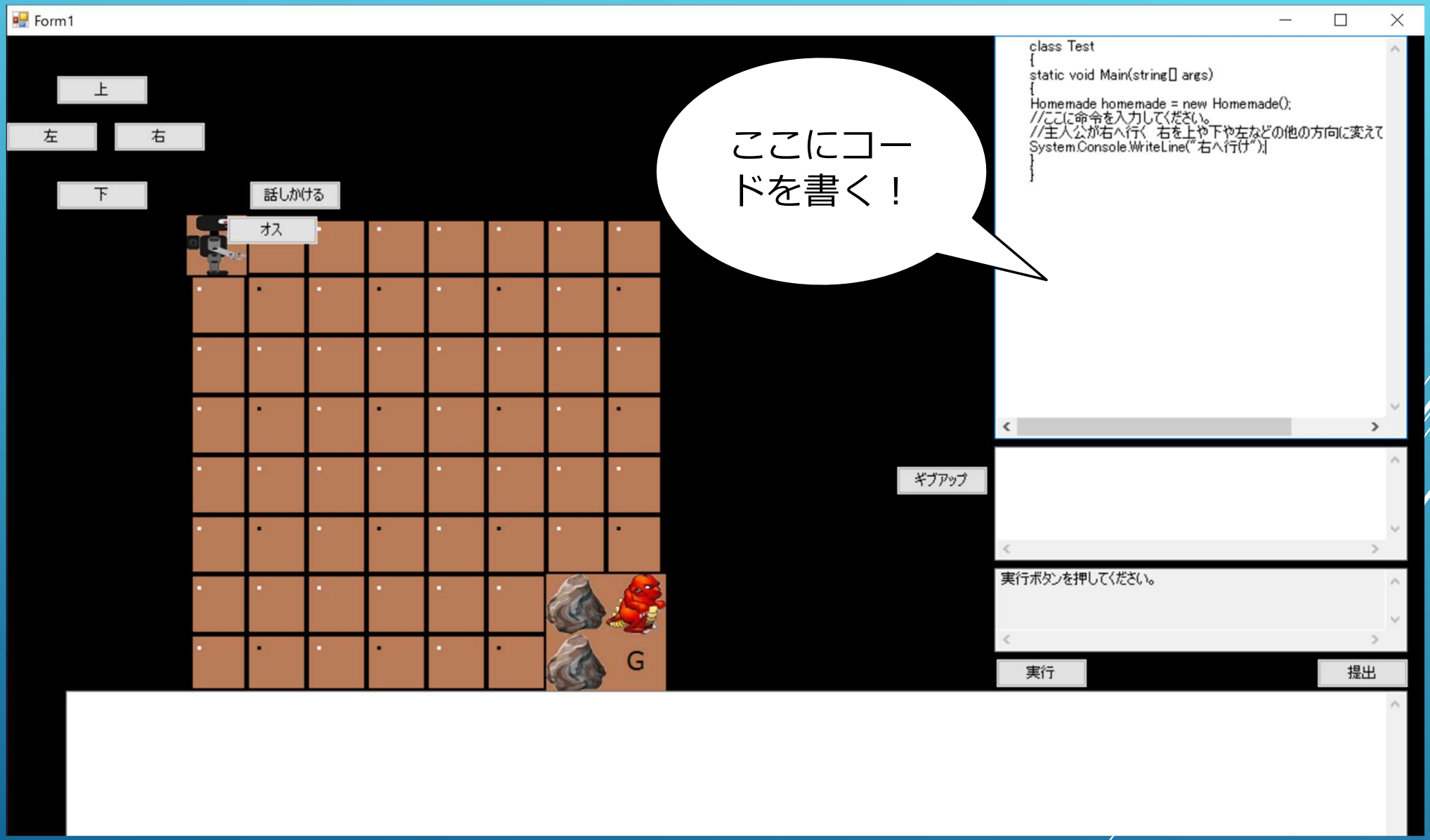
プログラミングを学べるゲームになっている！

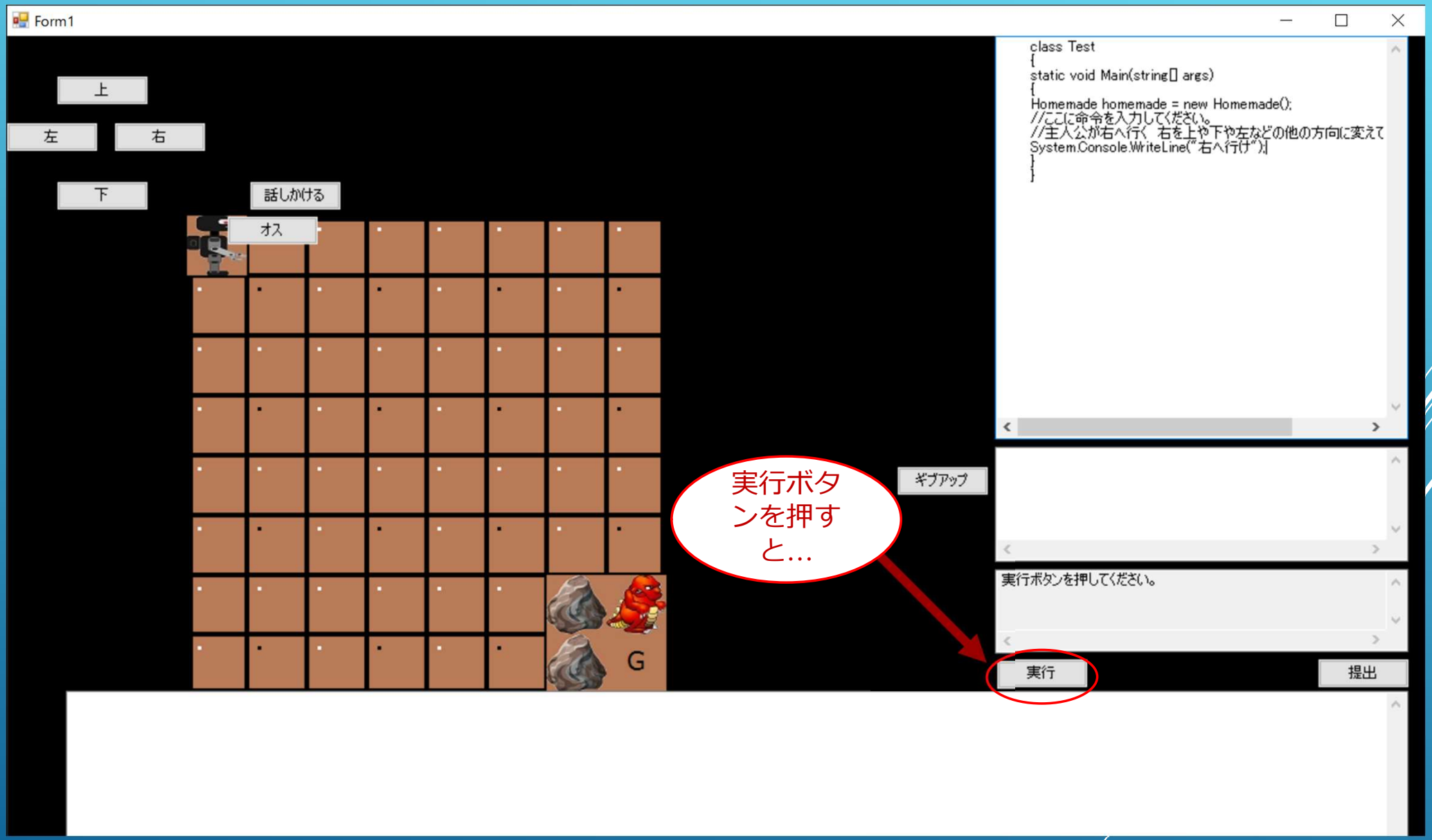
コードを書いて、主人公に命令を与えて、
迷路をクリアするゲームになっている！

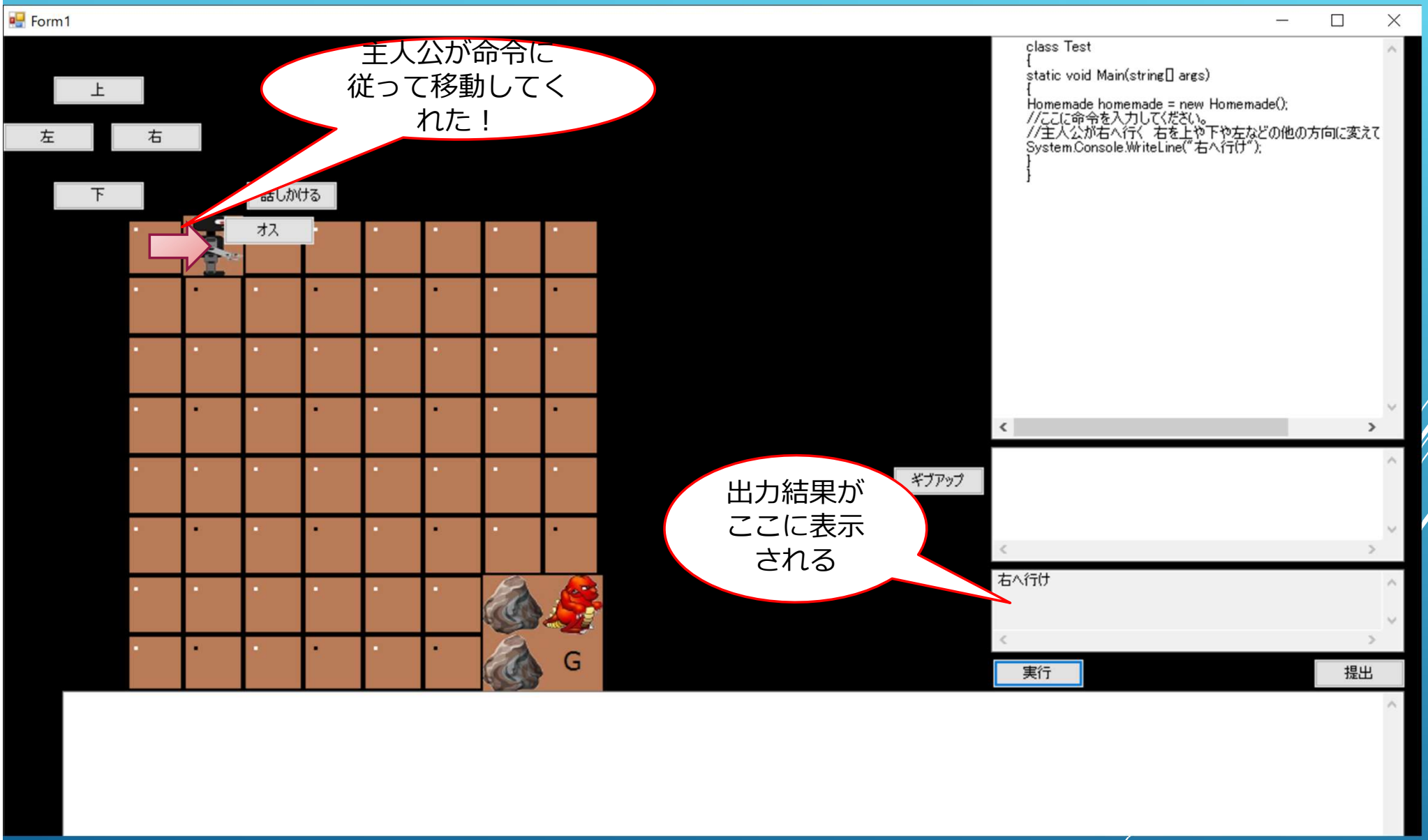
- C#でコードを書くと、その出力結果をもとに主人公が動いてくれる。

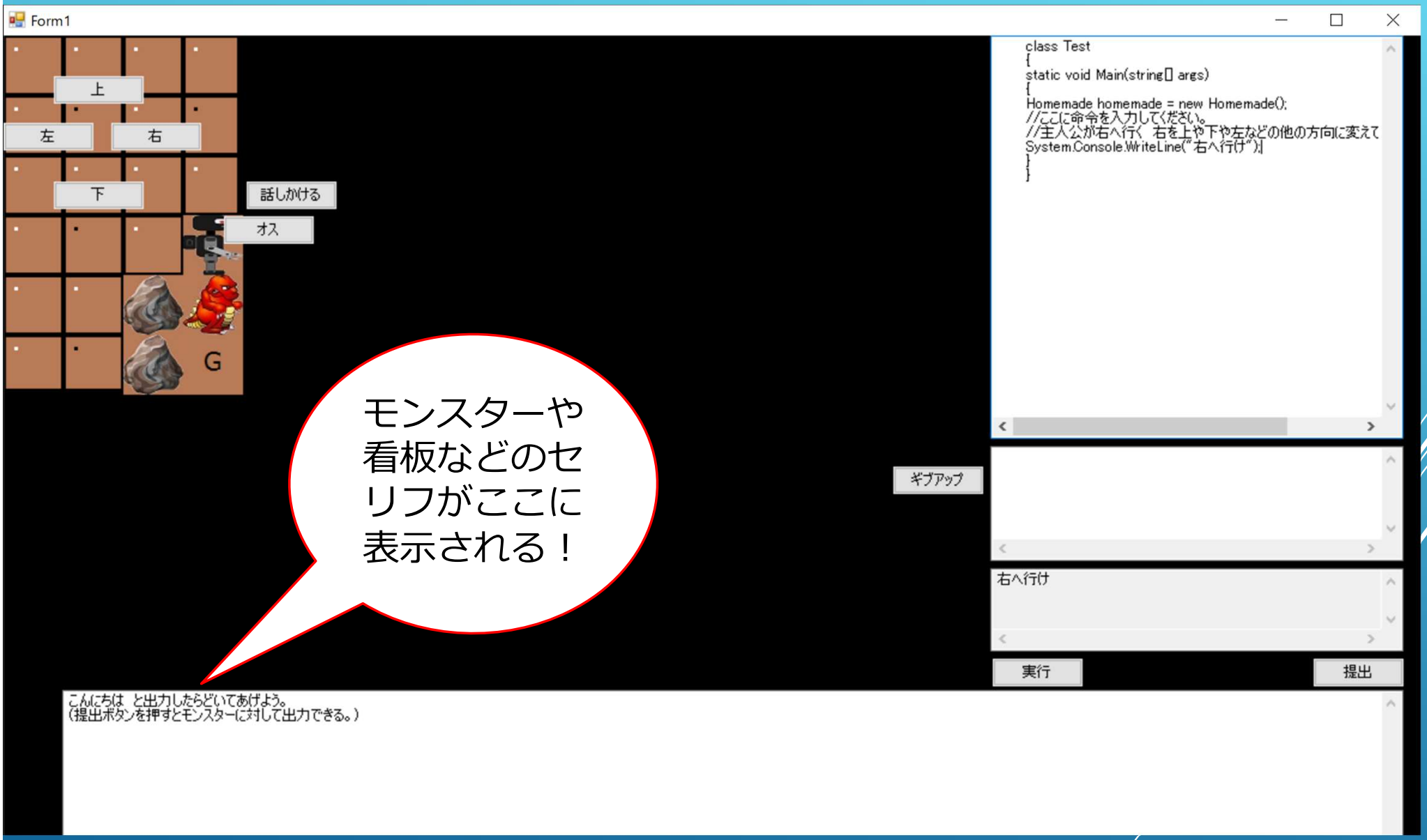
- Windowsに内蔵されているc#のコンパイラを呼び出して、その実行結果を取得する仕組みになっている。

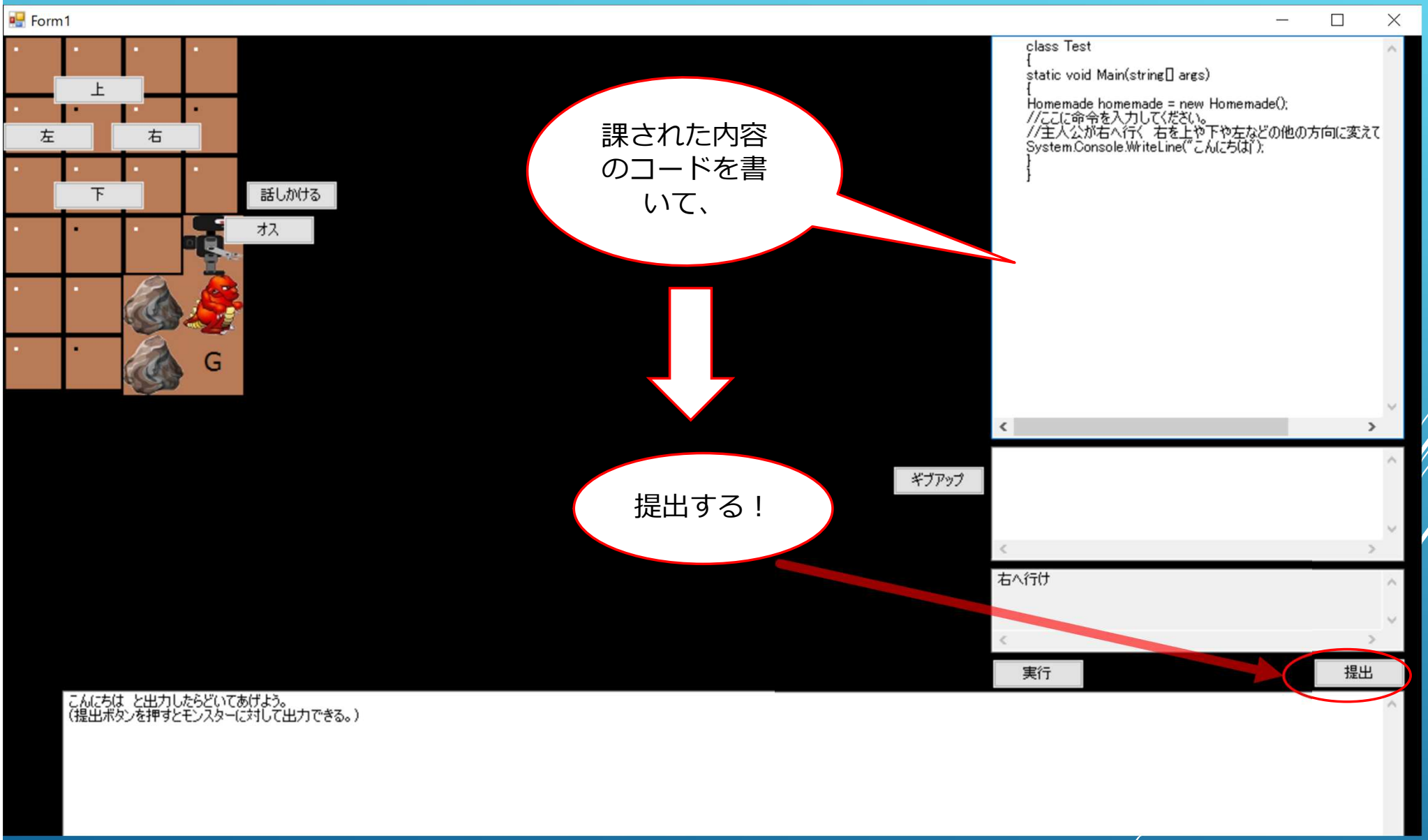
- ゲームをプレイしている様子を紹介する。

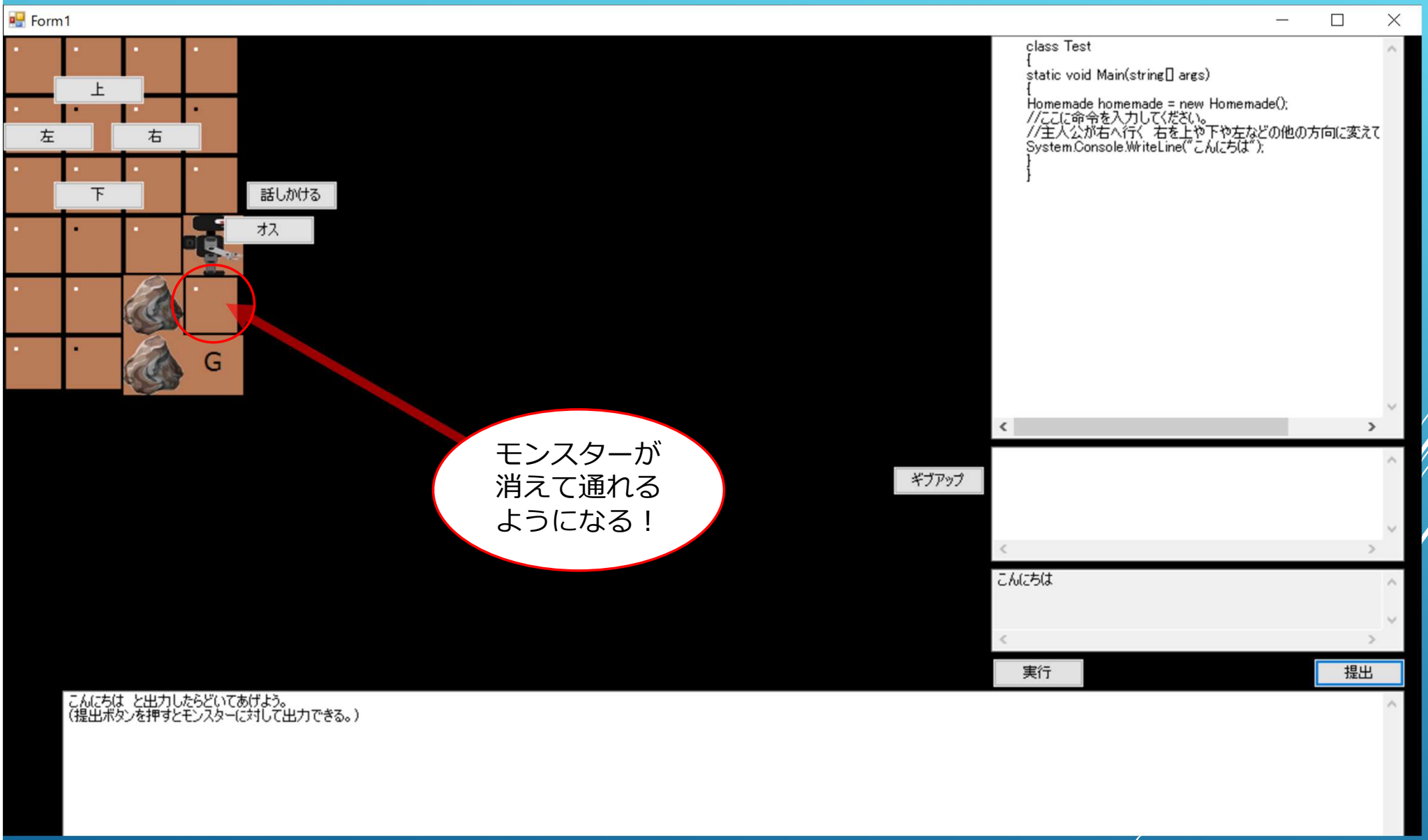

Form1
上
左
右
下
話しかける
オス
G
ここにコードを書く！
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("右へ行け");
}
}
ギブアップ
実行ボタンを押してください。
実行
提出

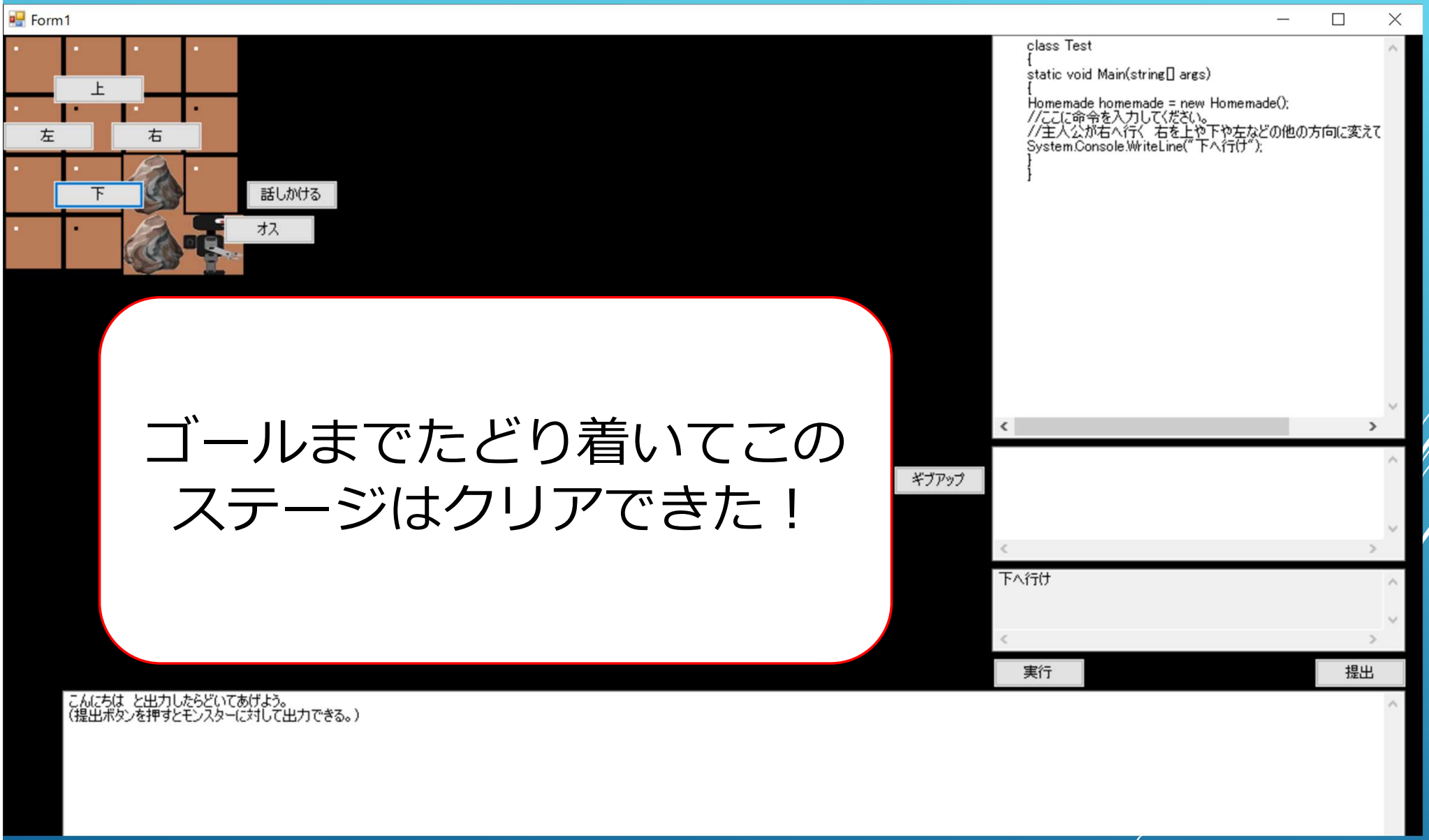

Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("右へ行け");
}
}
ギブアップ
実行ボタンを押すと…
実行ボタンを押してください。
実行
提出


Form1
主人公が命令に
従って移動してく
れた！
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("右へ行け");
}
}
ギブアップ
出力結果が
ここに表示
される
右へ行け
実行
提出


Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("右へ行け");
}
}
ギブアップ
右へ行け
実行
提出
こんにちは　と出力したらどいてあげよう。
(提出ボタンを押すとモンスターに対して出力できる。)
モンスターや
看板などのセ
リフがここに
表示される！


Form1
上
左
右
下
話しかける
オス
G
課された内容のコードを書いて、
提出する！
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("こんにちは");
}
}
ギブアップ
右へ行け
実行
提出
こんにちは　と出力したらどいてあげよう。
(提出ボタンを押すとモンスターに対して出力できる。)

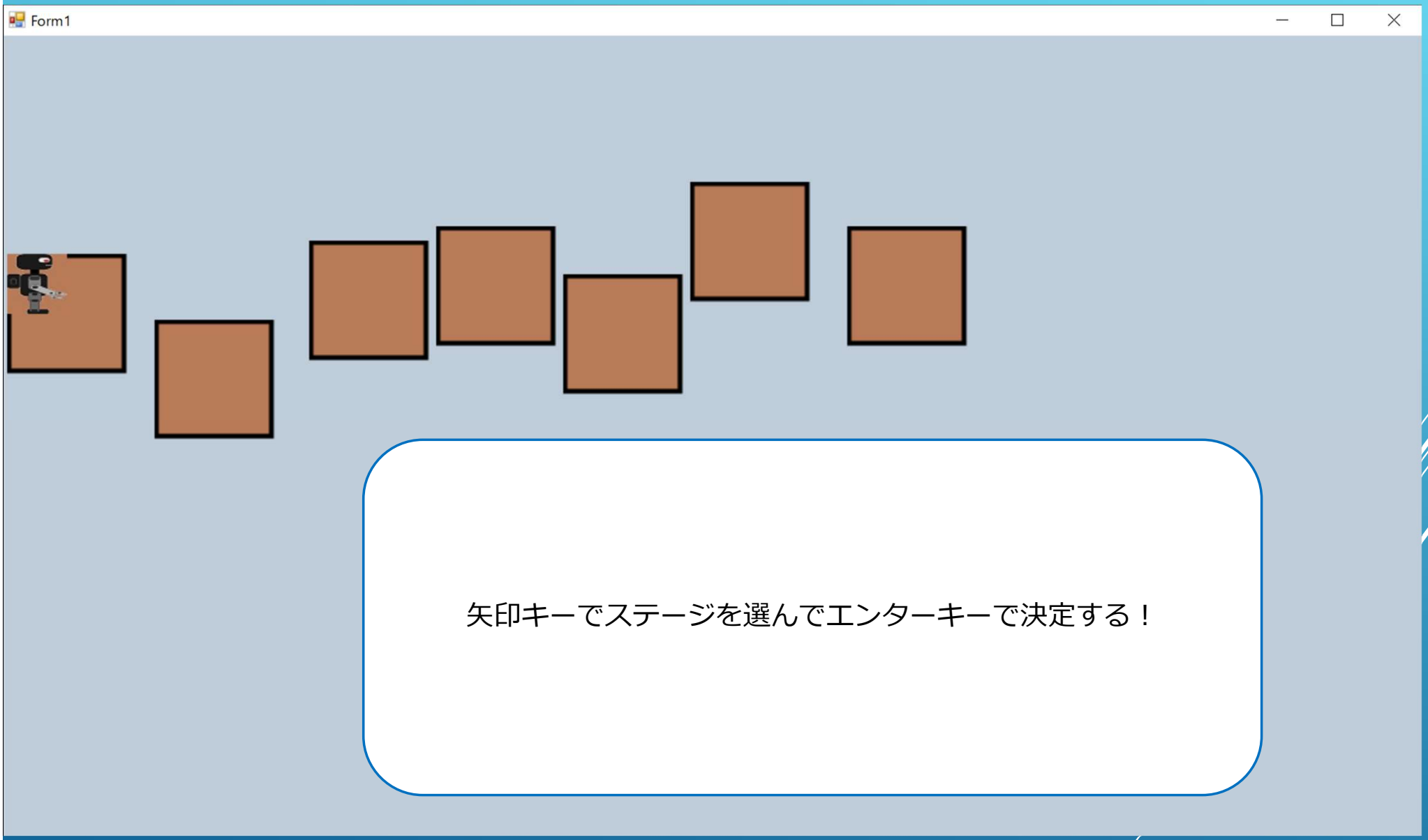

Form1
上
左
右
下
話しかける
オス
G
モンスターが
消えて通れる
ようになる！
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade();
//ここに命令を入力してください。
//主人公が右へ行く　右を上や下や左などの他の方向に変えて
System.Console.WriteLine("こんにちは");
}
}
ギブアップ
こんにちは
実行
提出
こんにちは　と出力したらどいてあげよう。
(提出ボタンを押すとモンスターに対して出力できる。)

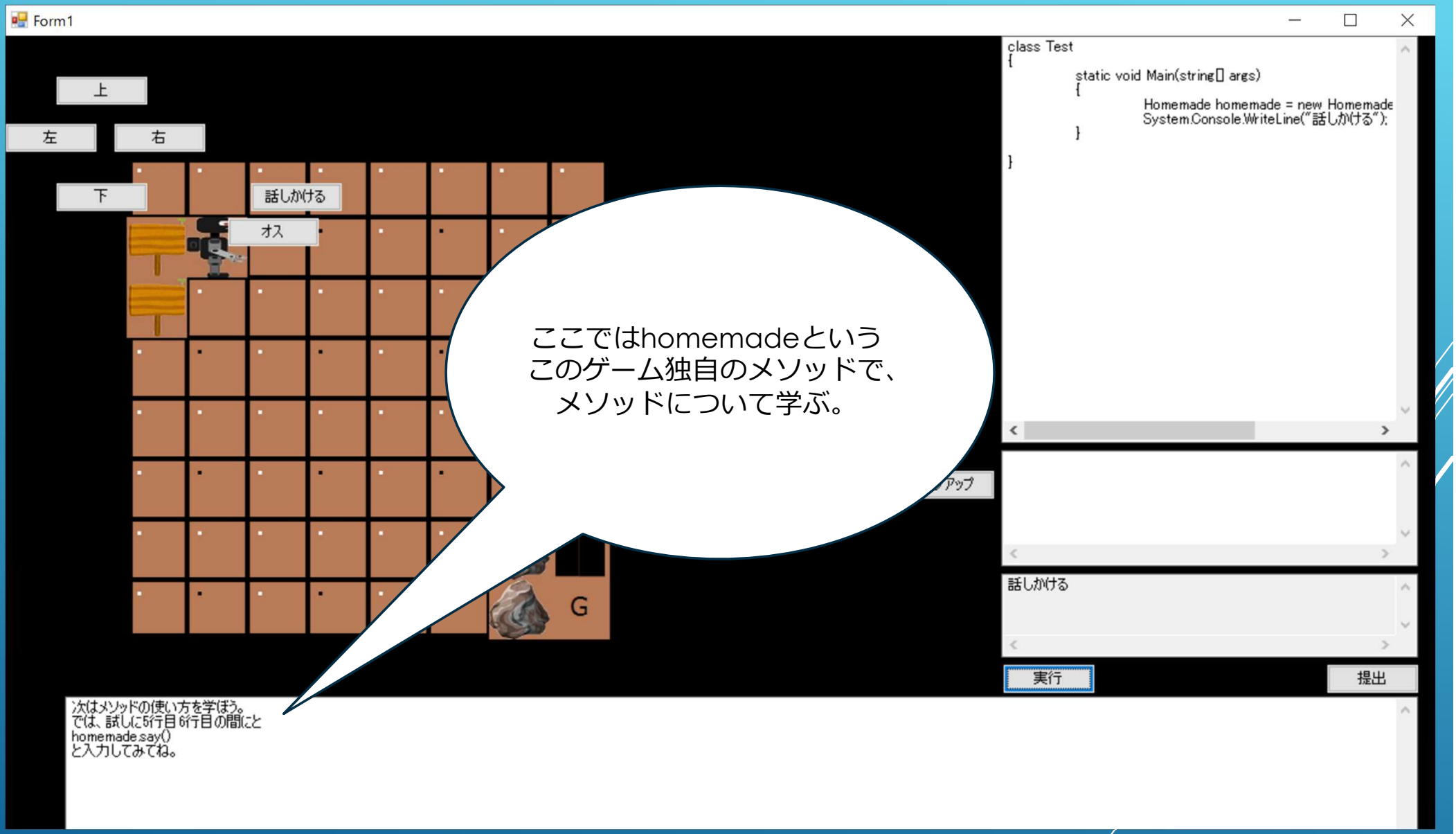


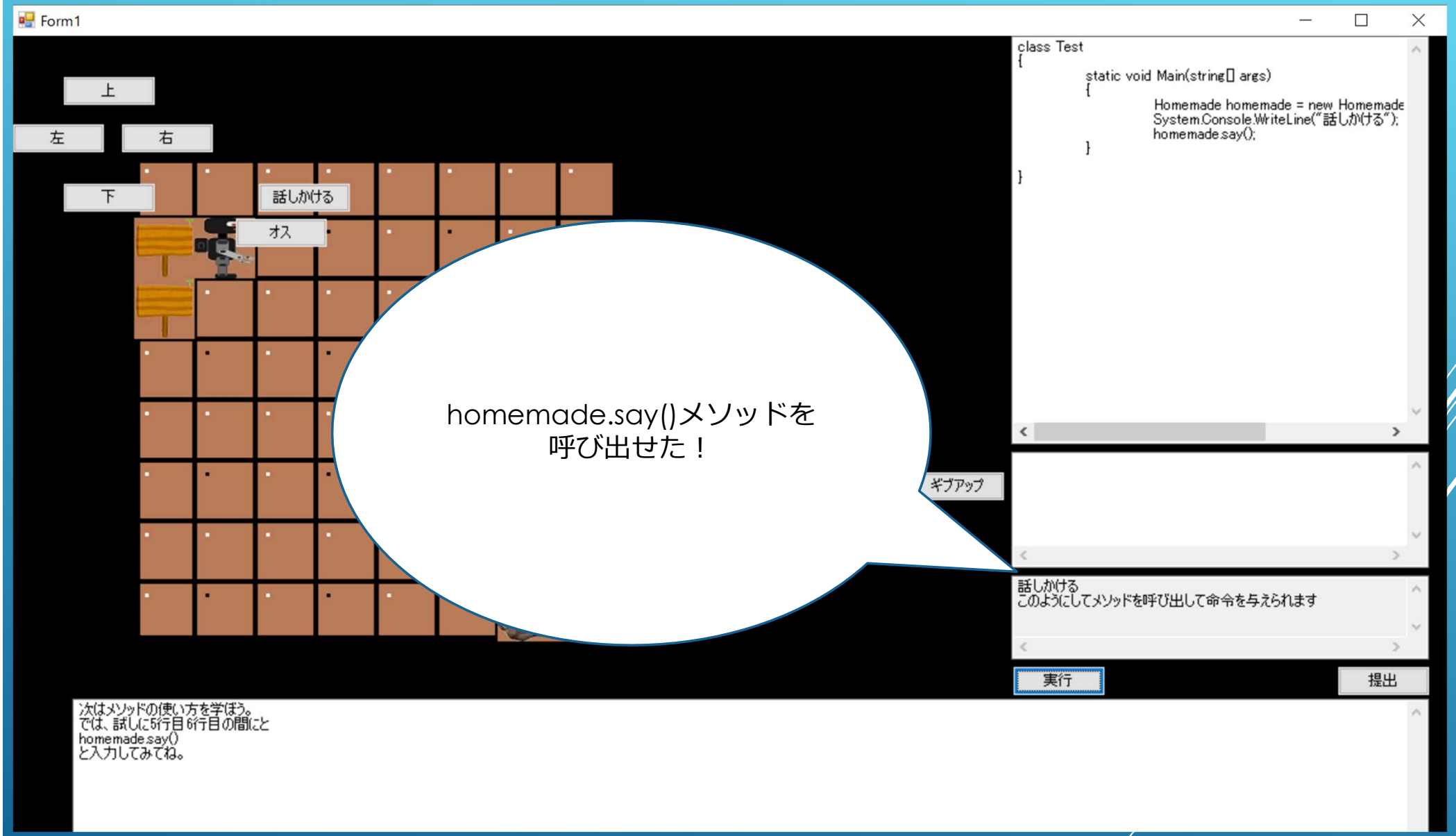
ゴールまでたどり着いてこのステージはクリアできた！

- このステージでは、出力の方法を学べたが、ステージごとによって学べる内容はさまざまである。
- 入力、条件分岐、繰り返しなどが学べるようになっている。

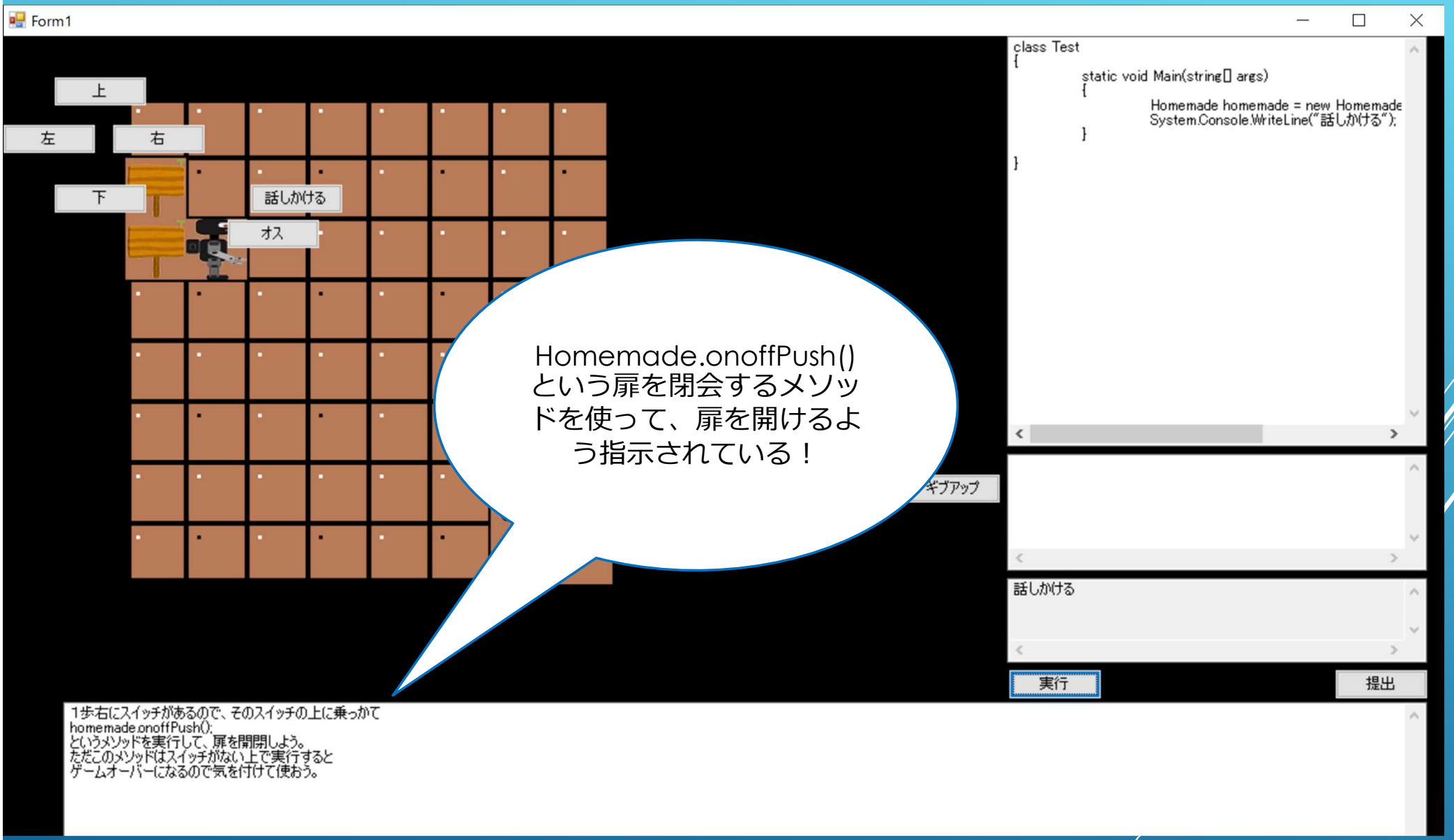
- 今から、各ステージごとの説明をする


Form1
矢印キーでステージを選んでエンターキーで決定する！

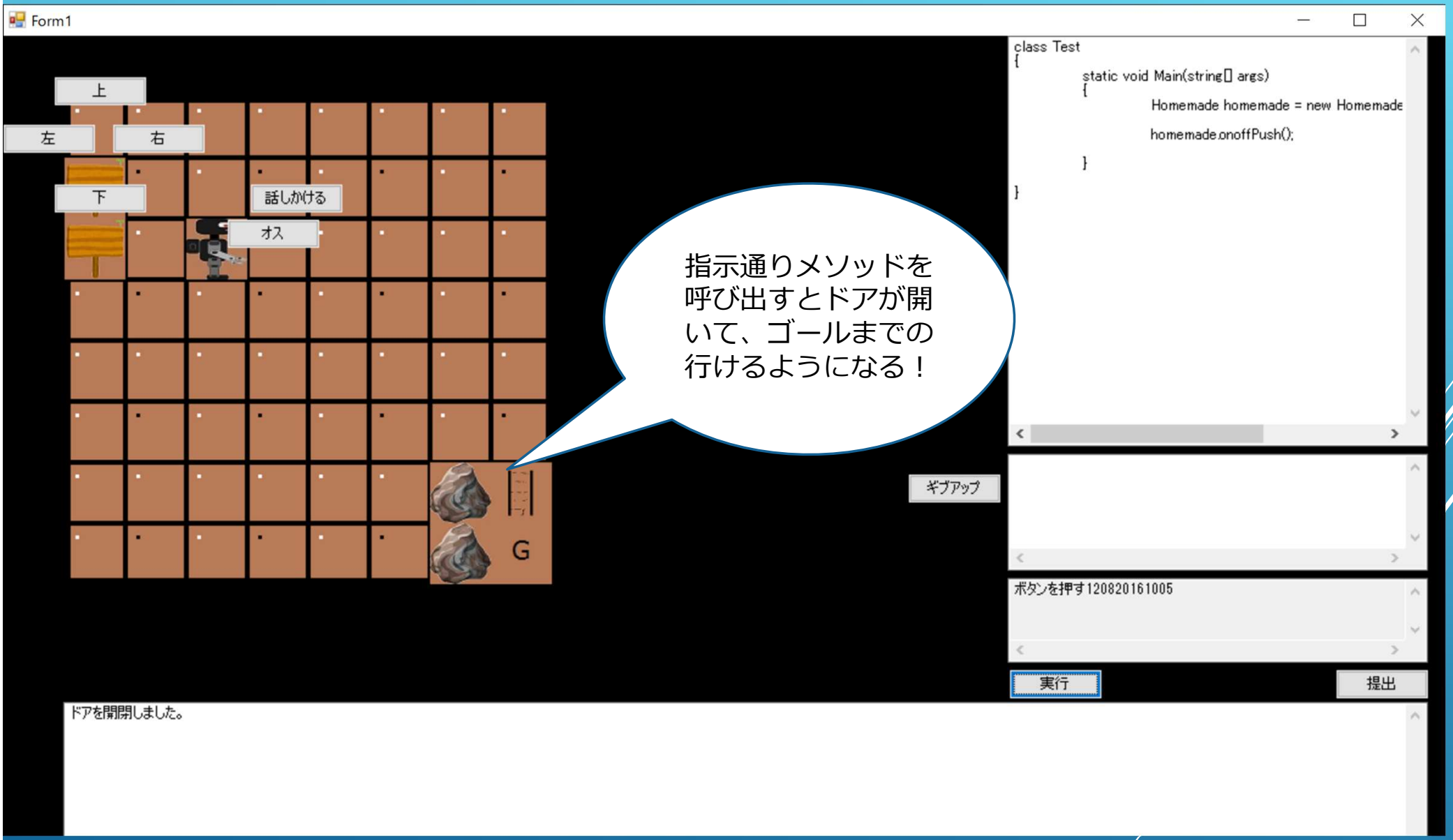

Form1
上
左
右
下
話しかける
オス
G
アップ
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
System.Console.WriteLine("話しかける");
}
}
話しかける
実行
提出
次はメソッドの使い方を学ぼう。
では、試しに5行目6行目の間にと
homemade.say()
と入力してみてね。
ここではhomemadeという
このゲーム独自のメソッドで、
メソッドについて学ぶ。



homemade.say()メソッドを
呼び出せた！

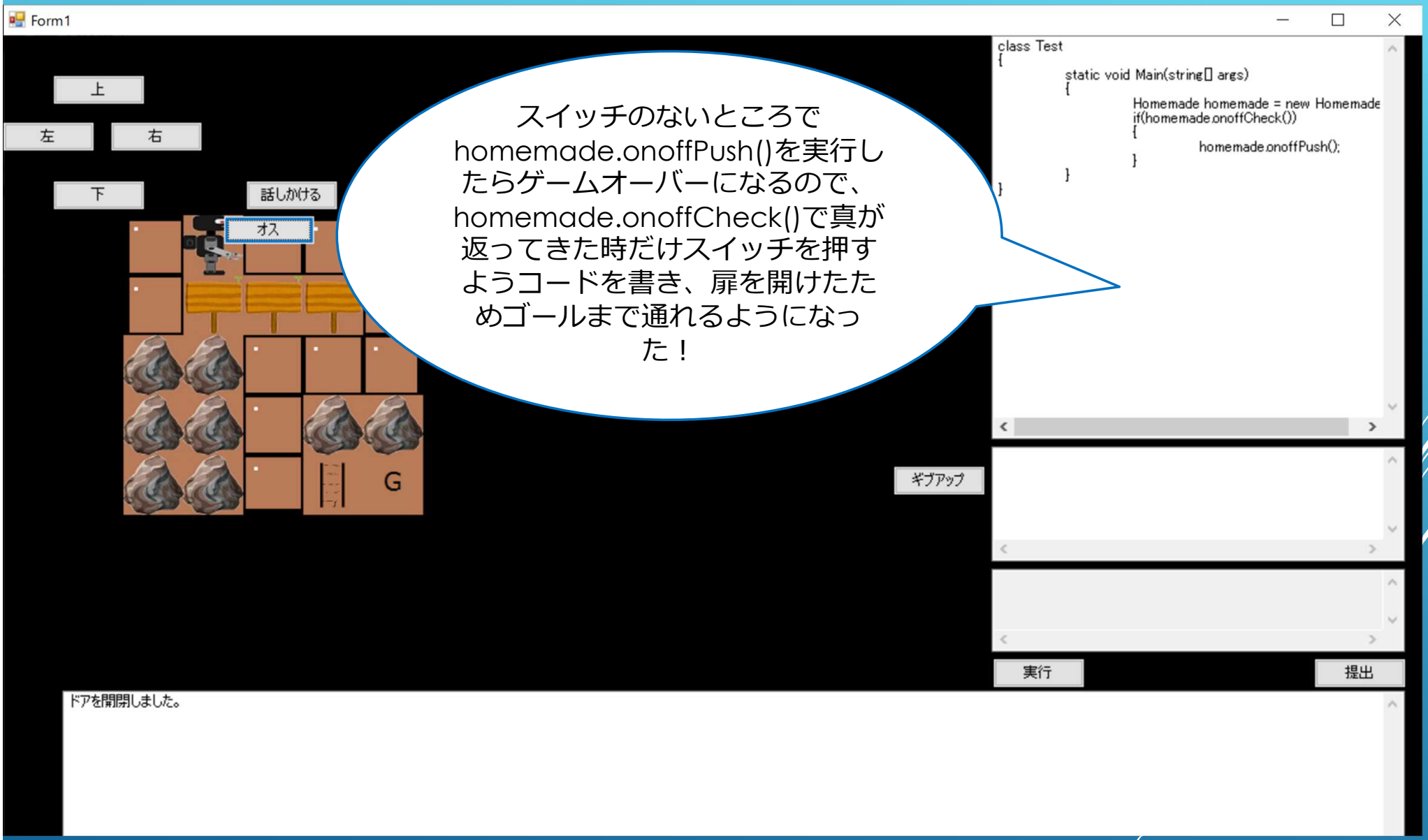


Homemade.onoffPush()という扉を閉会するメソッドを使って、扉を開けるよう指示されている！


Form1
上
左
右
下
話しかける
オス
G
指示通りメソッドを
呼び出すとドアが開
いて、ゴールまでの
行けるようになる！
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
homemade.onoffPush();
}
}
ギブアップ
ボタンを押す120820161005
実行
提出
ドアを開閉しました。

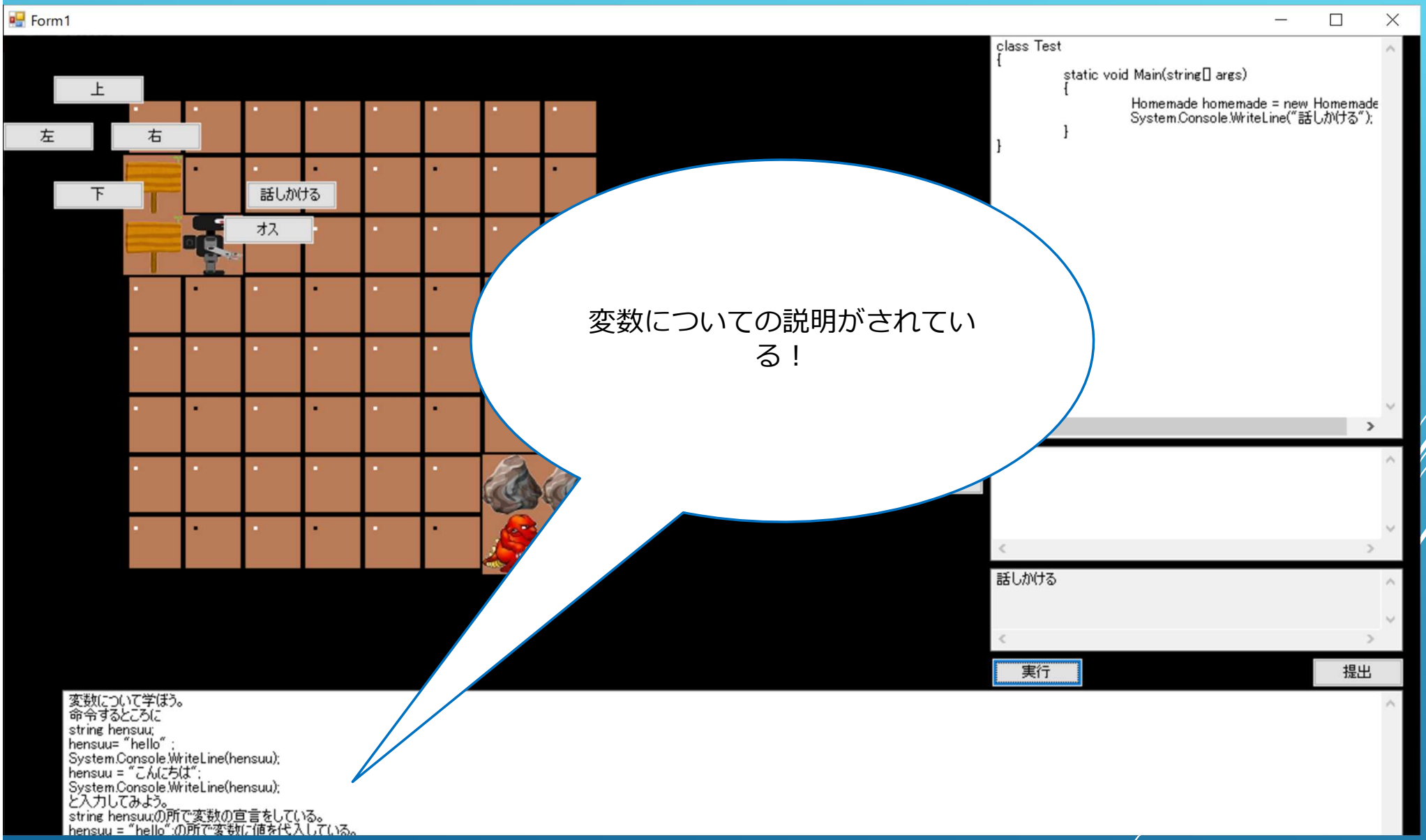
- このようにして、このステージではメソッドを学べた！


Form1
上
左
右
下
話しかける
オス
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
}
}
ここでは条件分岐について学ぶ。
If文の説明が書かれている。
ギブアップ
実行ボタンを押してください。
実行
提出
次は条件分岐について学ぼう。 if() というものを使うよ。
試しに命令を入力するところに
if(3+6 == 7+2) {
System.Console.WriteLine("同じ");
}
else {
System.Console.WriteLine("違う");
}
というコードを書いてみよう。

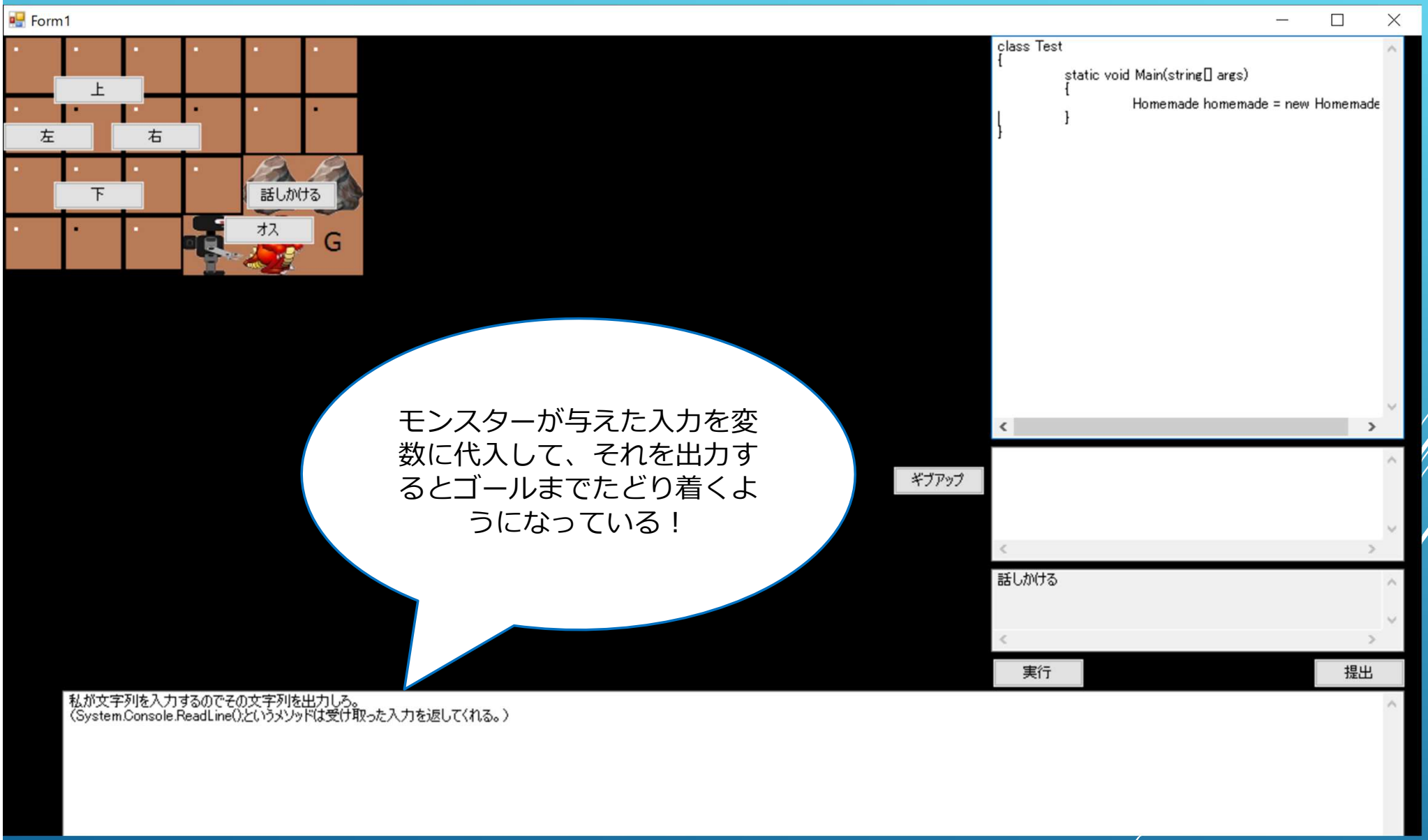


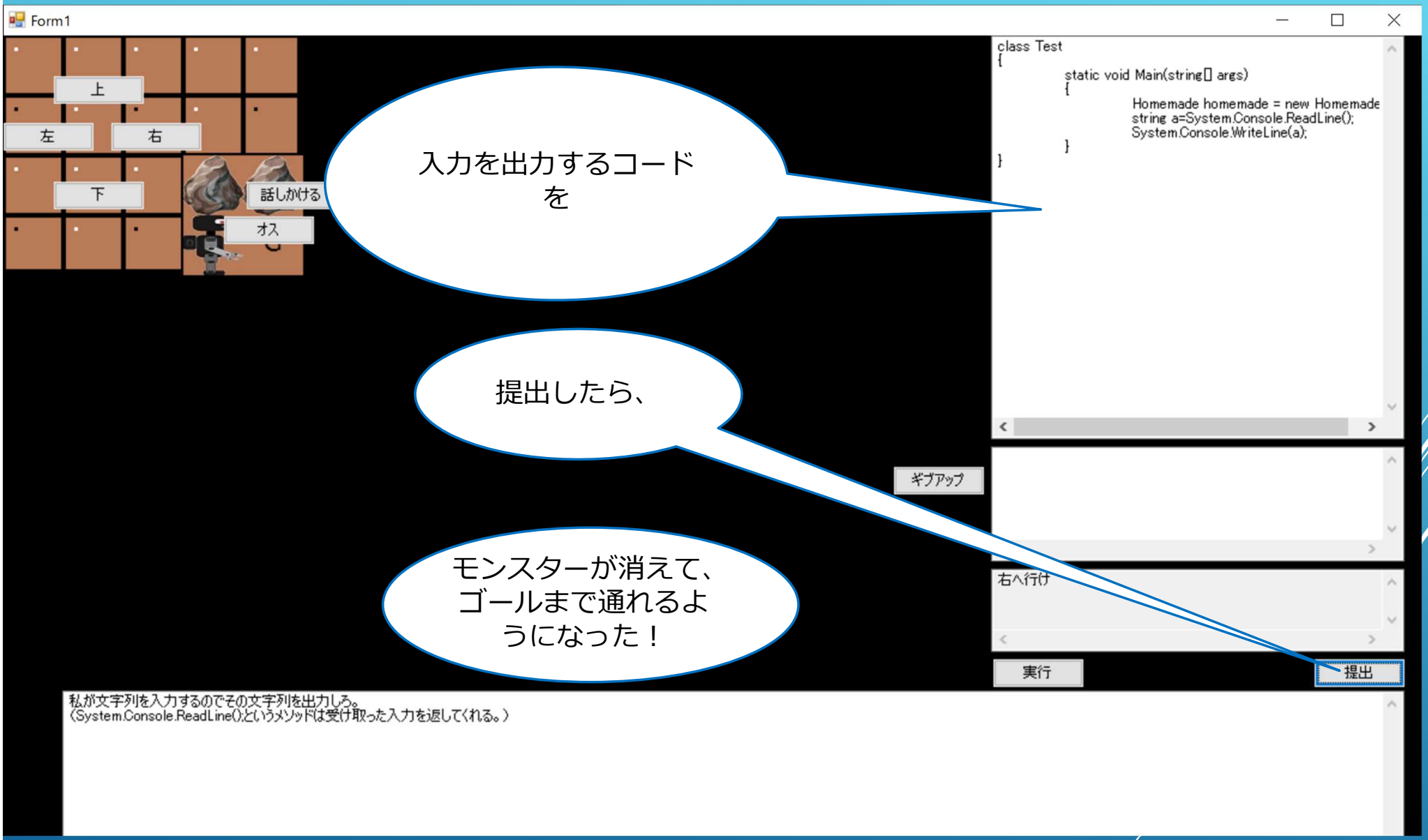
スイッチのないところで
homemade.onoffPush()を実行したらゲームオーバーになるので、
homemade.onoffCheck()で真が返ってきた時だけスイッチを押すようコードを書き、扉を開けたためゴールまで通れるようになった！

- このようにして、このステージでは条件分岐を学べた！

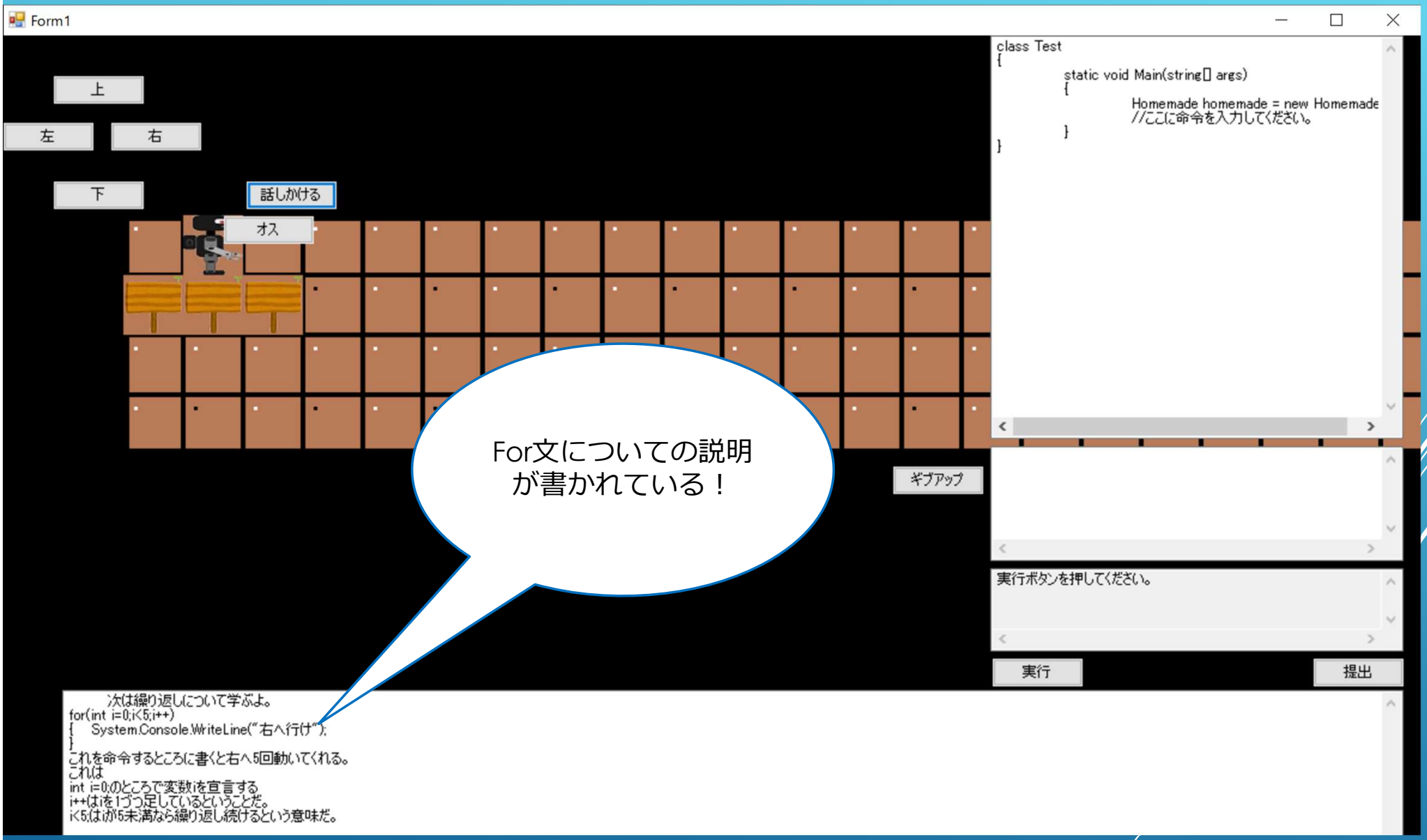


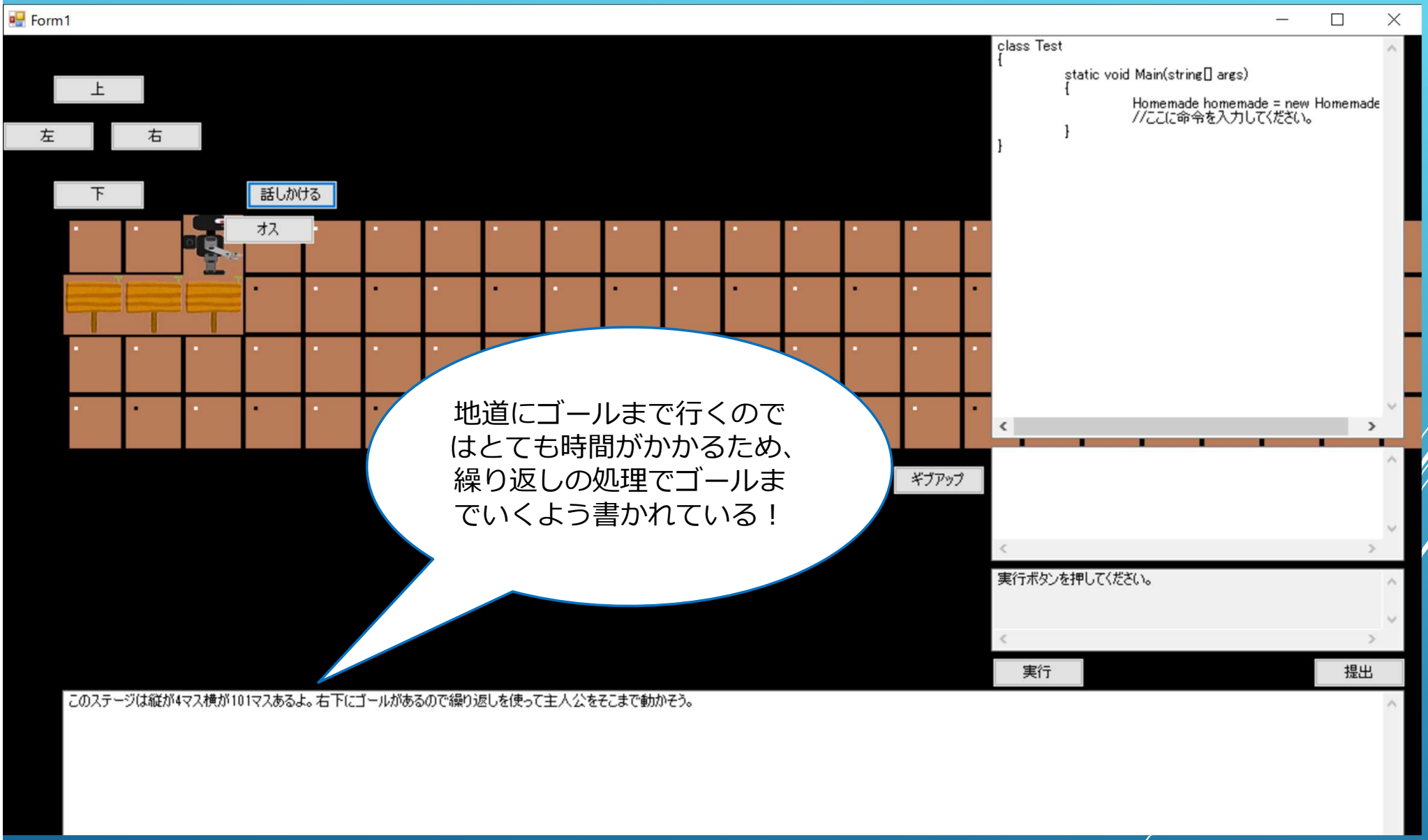
変数についての説明がされている！

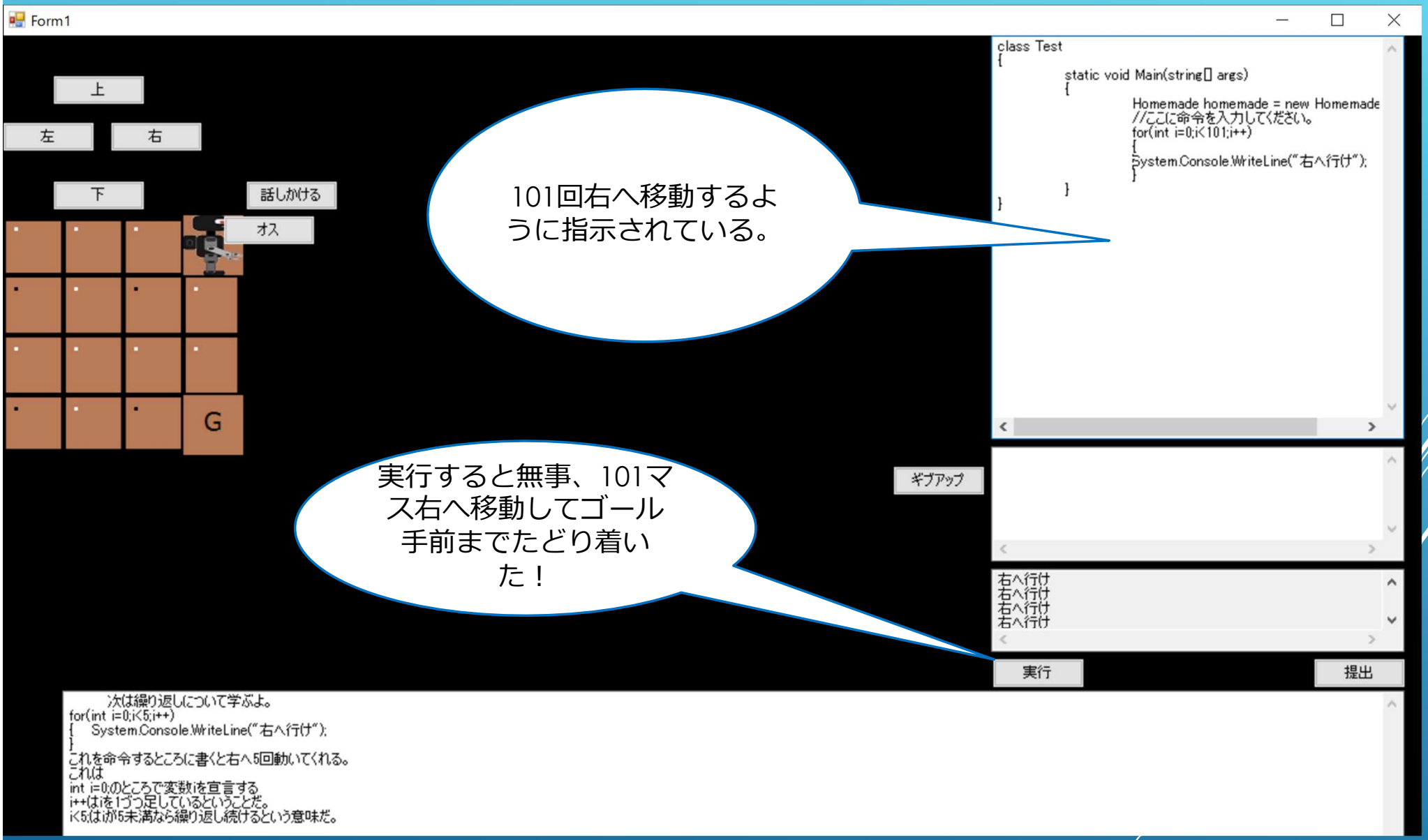

Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
}
}
モンスターが与えた入力を変数に代入して、それを出力するとゴールまでたどり着くようになっている！
ギブアップ
話しかける
実行
提出
私が文字列を入力するのでその文字列を出力しろ。
(System.Console.ReadLine();というメソッドは受け取った入力を返してくれる。)


Form1
上
左
右
下
話しかける
オス
入力を出力するコード
を
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
string a=System.Console.ReadLine();
System.Console.WriteLine(a);
}
}
提出したら、
ギブアップ
モンスターが消えて、
ゴールまで通れるよ
うになった！
右へ行け
実行
提出
私が文字列を入力するのでその文字列を出力しろ。
(System.Console.ReadLine();というメソッドは受け取った入力を返してくれる。)

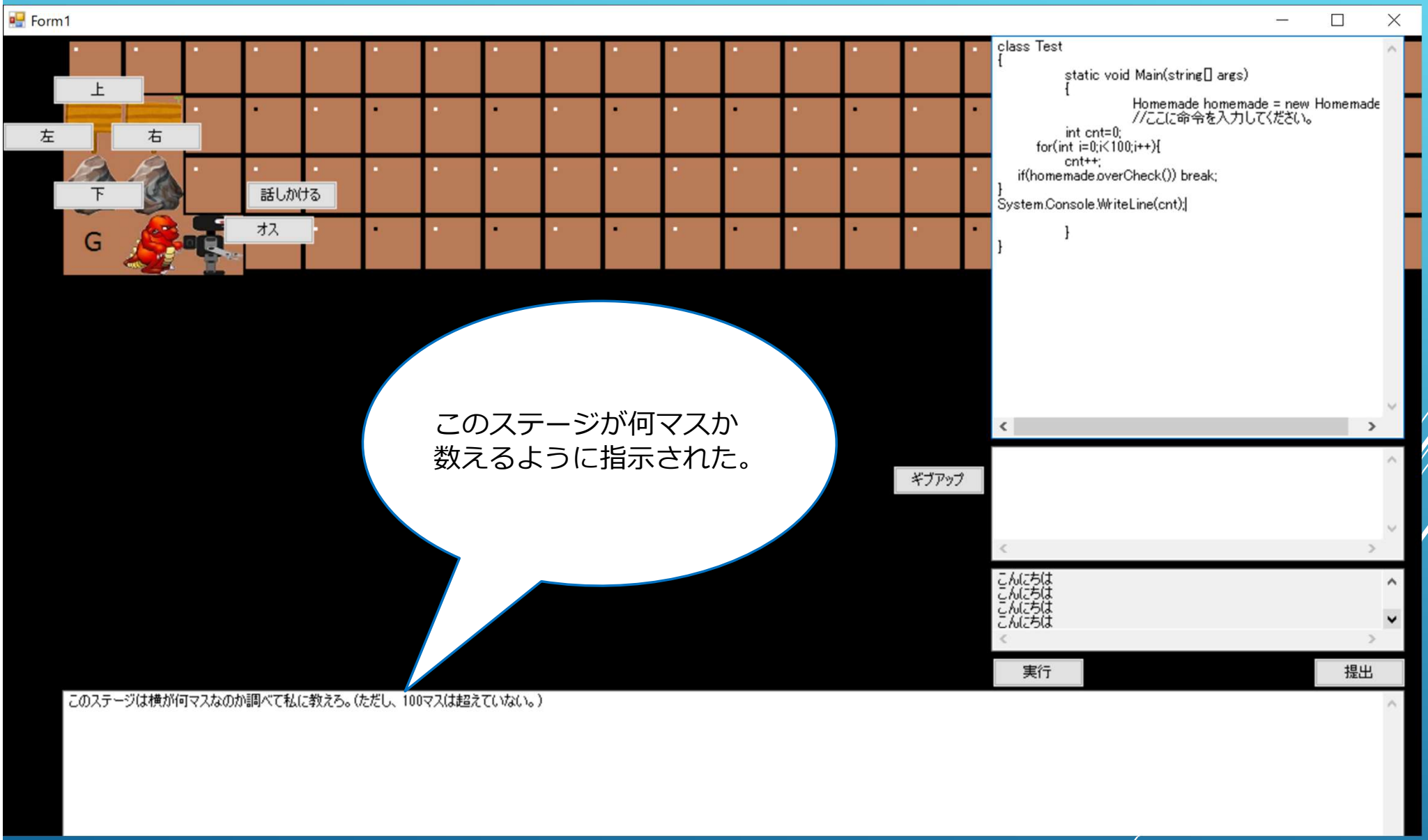
- このようにして、このステージでは変数と入力について学べた！

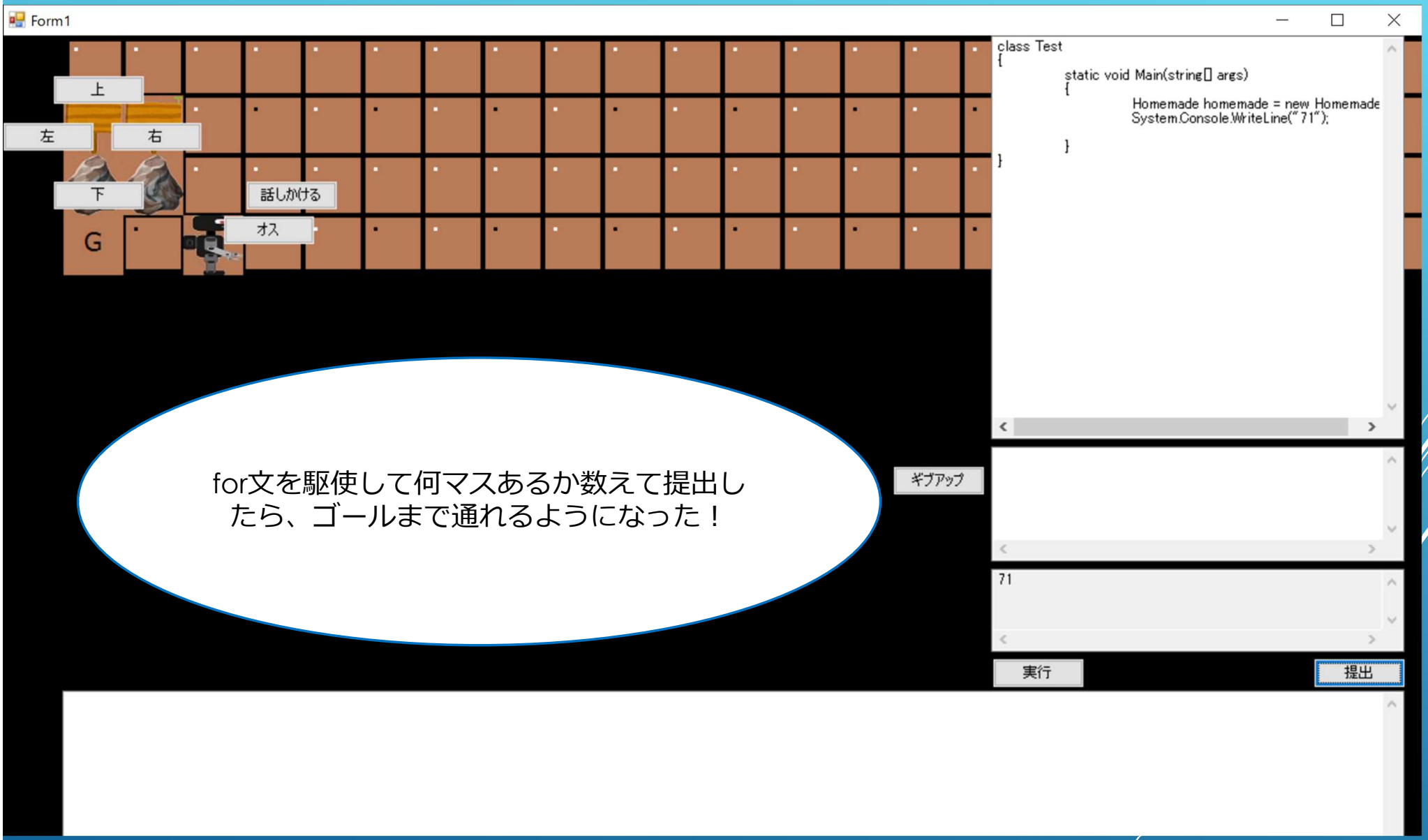

Form1
上
左
右
下
話しかける
オス
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
}
}
ギブアップ
実行ボタンを押してください。
実行
提出
For文についての説明
が書かれている！
次は繰り返しについて学ぶよ。
for(int i=0;i<5;i++)
{ System.Console.WriteLine("右へ行け");
}
これを命令するところに書くと右へ5回動いてくれる。
これは
int i=0;のところで変数iを宣言する
i++はiを1づつ足しているということだ。
i<5;はiが5未満なら繰り返し続けるという意味だ。


Form1
上
左
右
下
話しかける
オス
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
}
}
ギブアップ
実行ボタンを押してください。
実行
提出
このステージは縦が4マス横が101マスあるよ。右下にゴールがあるので繰り返しを使って主人公をそこまで動かそう。
地道にゴールまで行くので
はとても時間がかかるため、
繰り返しの処理でゴールま
でいくよう書かれている！

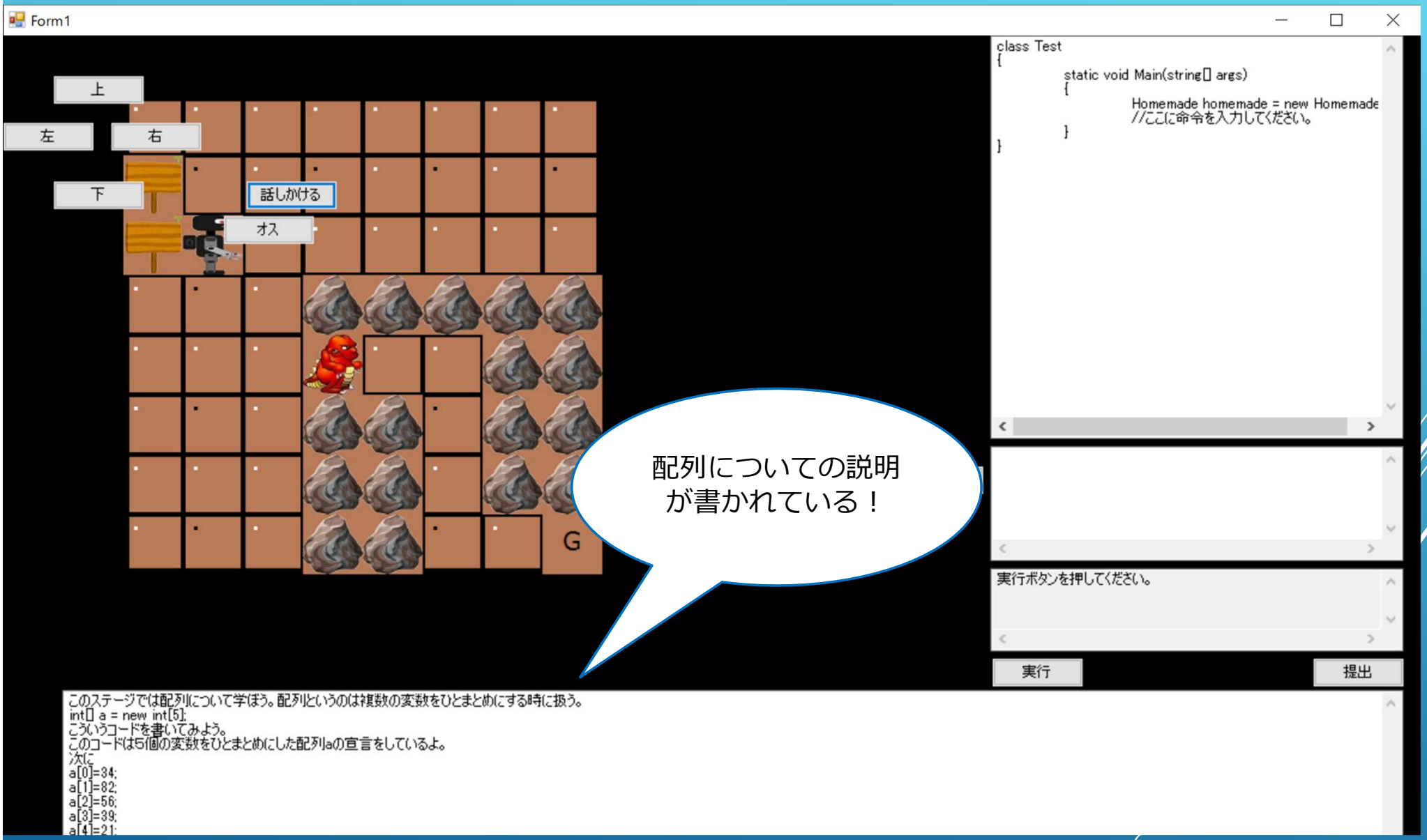

Form1
上
左
右
下
話しかける
オス
G
101回右へ移動するように指示されている。
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
for(int i=0;i<101;i++)
{
System.Console.WriteLine("右へ行け");
}
}
}
ギブアップ
実行すると無事、101マス右へ移動してゴール手前までたどり着いた！
右へ行け
右へ行け
右へ行け
右へ行け
実行
提出
次は繰り返しについて学ぶよ。
for(int i=0;i<5;i++)
{ System.Console.WriteLine("右へ行け");
}
これを命令するところに書くと右へ5回動いてくれる。
これは
int i=0;のところで変数iを宣言する
i++はiを1づつ足しているということだ。
i<5;はiが5未満なら繰り返し続けるという意味だ。

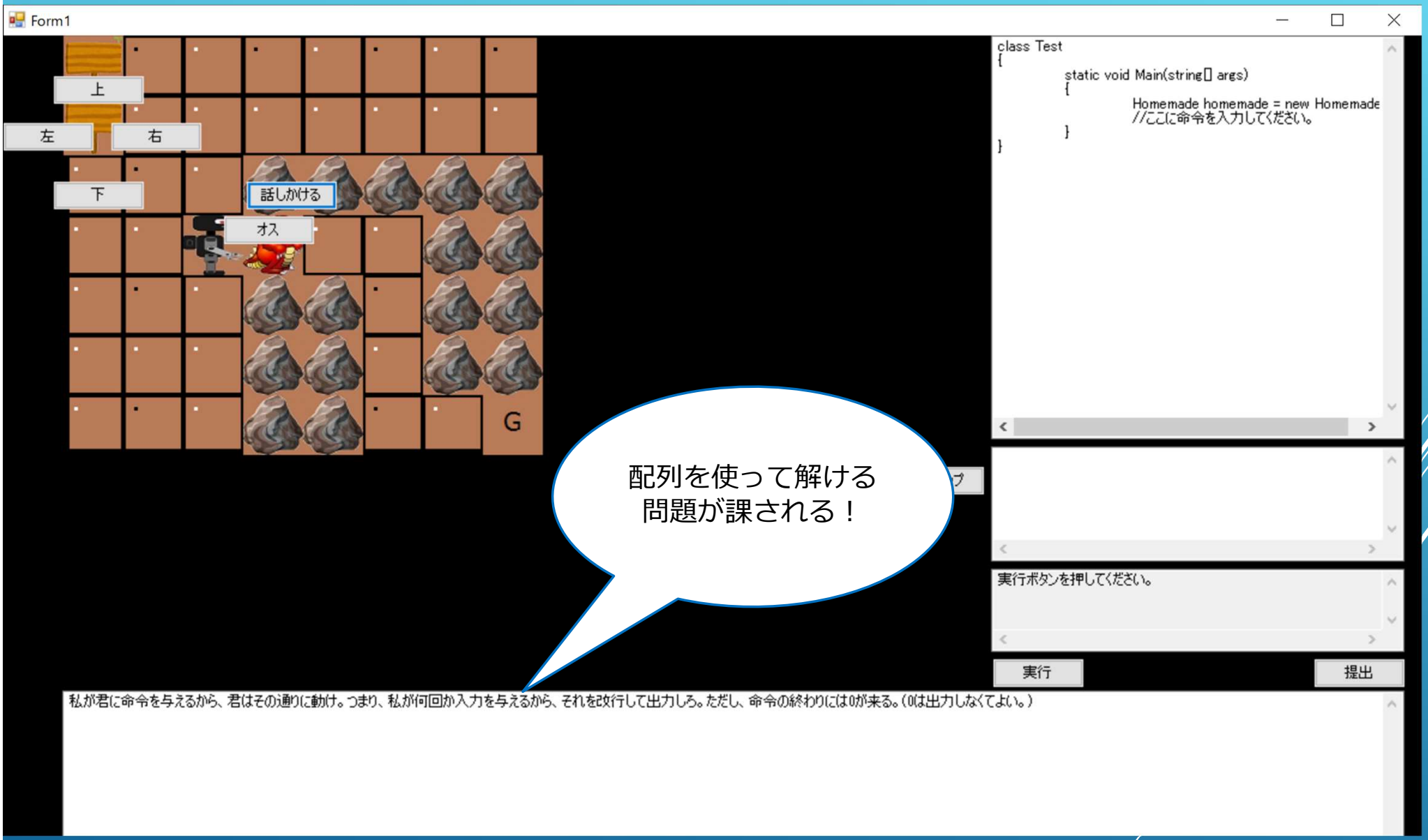
- このようにして、このステージでは繰り返しの処理を学べるようになっている。
- このステージはゴールまで横に101マスあるが、ステージ内容のデータを編集すれば簡単に、千マスにも１万マスにも変更できる。


Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
int cnt=0;
for(int i=0;i<100;i++){
cnt++;
if(homemade.overCheck()) break;
}
System.Console.WriteLine(cnt);
}
}
このステージが何マスか
数えるように指示された。
ギブアップ
こんにちは
こんにちは
こんにちは
こんにちは
実行
提出
このステージは横が何マスなのか調べて私に教えろ。(ただし、100マスは超えていない。)

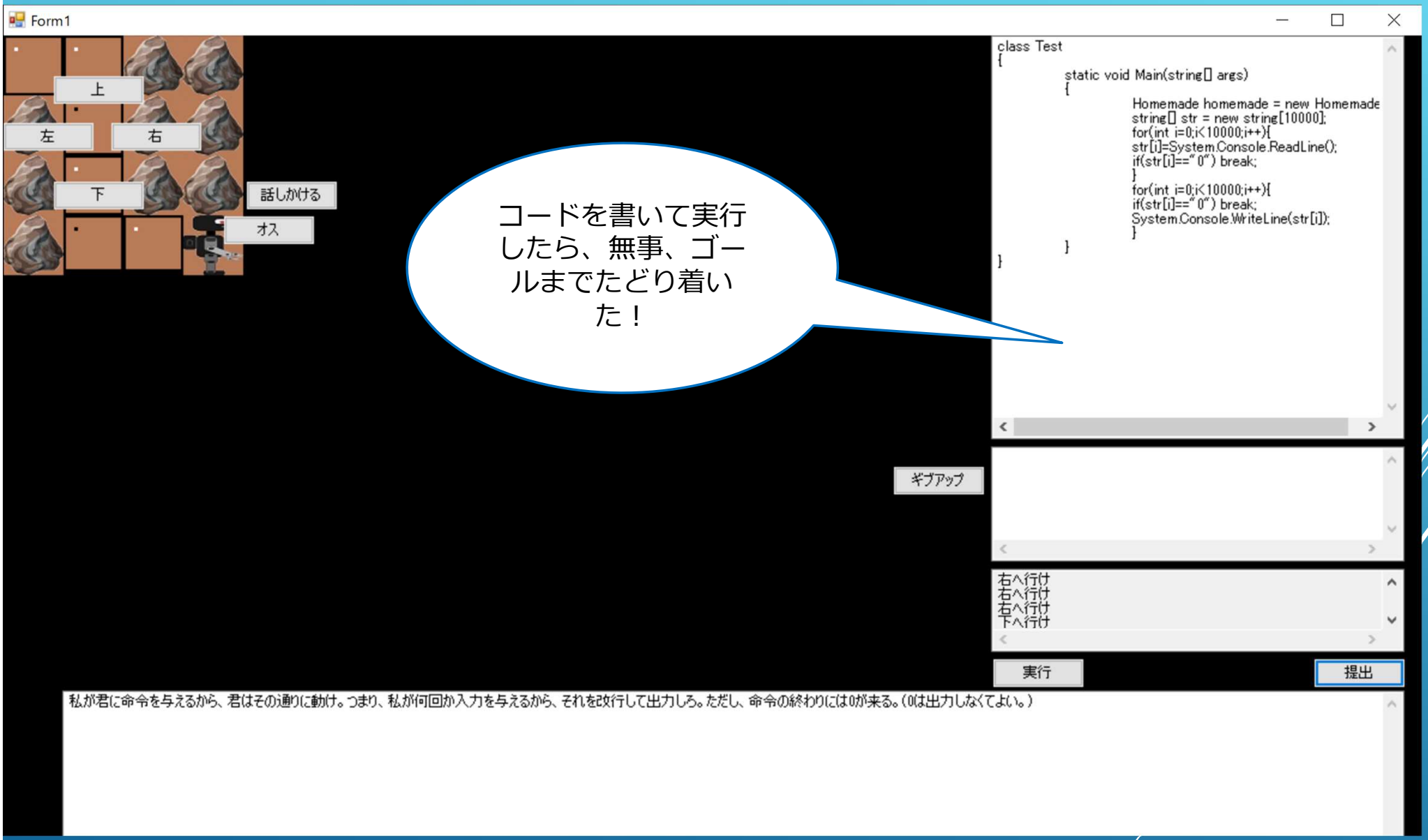

Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
System.Console.WriteLine("71");
}
}
for文を駆使して何マスあるか数えて提出し
たら、ゴールまで通れるようになった！
ギブアップ
71
実行
提出

- このようにして、このステージでは繰り返しについて再び学べた！


Form1
上
左
右
下
話しかける
オス
G
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
//ここに命令を入力してください。
}
}
配列についての説明
が書かれている！
実行ボタンを押してください。
実行
提出
このステージでは配列について学ぼう。配列というのは複数の変数をひとまとめにする時に扱う。
int[] a = new int[5];
こういうコードを書いてみよう。
このコードは5個の変数をひとまとめにした配列aの宣言をしているよ。
次に
a[0]=34;
a[1]=82;
a[2]=56;
a[3]=39;
a[4]=21;

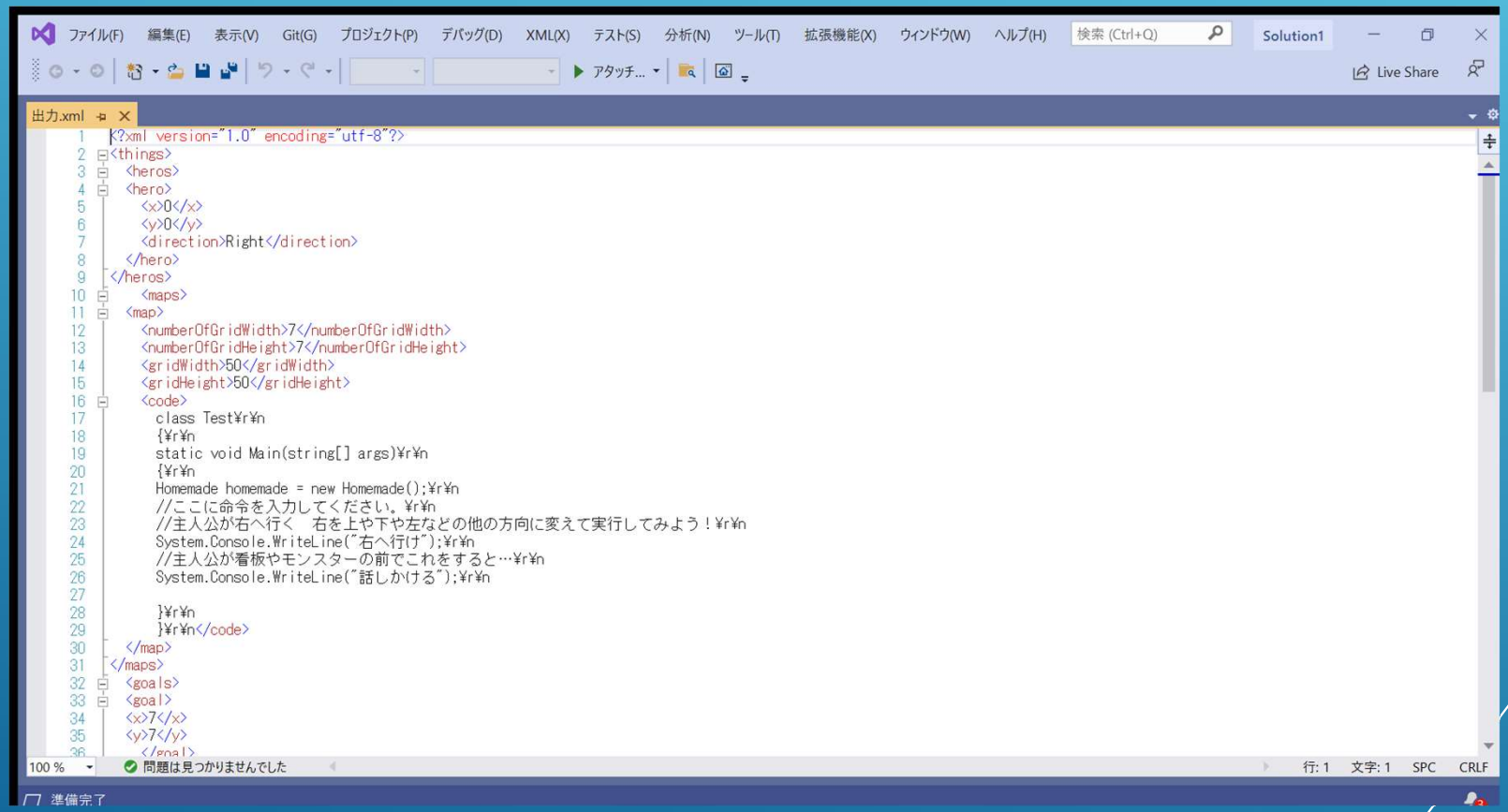


配列を使って解ける
問題が課される！


Form1
上
左
右
下
話しかける
オス
コードを書いて実行したら、無事、ゴールまでたどり着いた！
class Test
{
static void Main(string[] args)
{
Homemade homemade = new Homemade
string[] str = new string[10000];
for(int i=0;i<10000;i++){
str[i]=System.Console.ReadLine();
if(str[i]=="0") break;
}
for(int i=0;i<10000;i++){
if(str[i]=="0") break;
System.Console.WriteLine(str[i]);
}
}
}
ギブアップ
右へ行け
右へ行け
右へ行け
下へ行け
実行
提出
私が君に命令を与えるから、君はその通りに動け。つまり、私が何回か入力を与えるから、それを改行して出力しろ。ただし、命令の終わりには0が来る。(0は出力しなくてよい。)

- このようにして、このステージでは配列を学ぶことができた！

- xmlを読み取ってステージが作られるため、xmlを編集することで各ステージは自由に変更できる仕様になっている！



- ここで紹介したステージはあくまで１例であって、xmlにより自分でも簡単にステージは変更できるため、このゲームは無限の可能性が秘めている！

～終わり～