

~\Documents\GitHub\Projects\evan\_baesler\_assignment3.cpp

```
1  #include <iostream>
2  #include <cstdlib>
3  #include <ctime>
4  #include <string>
5
6  using namespace std;
7
8  // Our function for the player to input their guess.
9  int playerTurn(int low, int high){
10
11     int output = 0;
12
13     // We use a true loop since return will exit this function entirely.
14     while (true)
15     {
16
17         cout << endl << "Your turn! Enter your guess: ";
18         cin >> output;
19
20         // Checks for letters or other non-numeric inputs.
21         if(cin.fail()){
22             cout << endl << "Invalid input! Please enter a number between 10 and 99." << endl;
23             cin.clear();
24             cin.ignore(256, '\n');
25         }
26
27         // Checks to make sure input is within bound.
28         else if(output >= high || output <= low){
29
30             cout << endl << "Invalid input! Please enter a number between " << low << " and " << high << "." << endl;
31
32         }
33
34         // If our input meets our bounds, we record it as the guess.
35         else if(output < high && output > low){
36
37             return(output);
38
39         }
40     }
41 }
42
43 }
44
45 int computerTurn(int low, int high){
46
47     int output = 0;
48
49     // Generates a randomized number with an offset starting at (low + 1) or 11 at the beginning to a high of (high - low - 1) or (99 - 10 - 1) = 88.
50     // Basically at initialization would be rand() % 88 + 11, or [0,87] + 11 for [11 , 98].
51     output = rand() % (high - low - 1) + low + 1;
52     cout << endl << "Computer guesses: ";
53     cout << output << endl;
54
55     return(output);
56
57 }
58
```

```

59 int main() {
60
61     // Initializes our randomization key at the beginning of the program execution.
62     srand(time(0));
63     bool playing = true;
64     int answer = 0;
65     int guessLog = 0;
66     bool closingStatement = true;
67     string closeCheck;
68
69     // Uses a true loop because the function will either be infinitely repeated, or broken by a return when program is exited.
70     while(true){
71
72         answer = rand() % 90 + 10;
73         int low = 10;
74         int high = 99;
75         playing = true;
76
77         cout << endl << "Welcome to the Guessing Game!" << endl;
78         cout << "A secret 2-digit code has been set between 10 and 99." << endl << endl;
79
80         // This loop represents our gameplay, and the messages above are only repeated when this loop is broken (another game is started.)
81         while(playing){
82
83             // We give the player the first turn, and then check their input. If the answer is correct in any case, there is a break and the loop ends.
84             guessLog = playerTurn(low,high);
85
86             if(guessLog == answer){
87
88                 playing = false;
89                 cout << endl << "You cracked the code!" << endl;
90                 break;
91
92             }
93             else if(guessLog < answer){
94
95                 cout << endl << "Too low!" << endl;
96                 low = guessLog;
97
98             }
99             else{
100
101                 cout << endl << "Too high!" << endl;
102                 high = guessLog;
103
104             }
105
106
107             // Follows the same check logic as above.
108             guessLog = computerTurn(low,high);
109
110             if(guessLog == answer){
111
112                 playing = false;
113                 cout << endl << "Computer cracked the code!" << endl;
114                 break;
115
116             }
117             else if(guessLog < answer){
118
119                 cout << endl << "Too low!" << endl;

```

```
120         low = guessLog;
121     }
122 }
123 else{
124
125     cout << endl << "Too high!" << endl;
126     high = guessLog;
127
128 }
129
130 }
131
132
133 // After a game is over, turns the boolean to true, iterates through a loop until a sufficient response is given, then sets it to false to start a new game or returns to exit the program.
134 closingStatement = true;
135
136 while(closingStatement){
137
138     cout << endl << "Would you like to play again? (y/n): ";
139     cin >> closeCheck;
140
141     if(closeCheck == "y" || closeCheck == "Y"){
142
143         closingStatement = false;
144         playing = false;
145
146     }
147     else if (closeCheck == "n" || closeCheck == "N"){
148
149         return(0);
150
151     }
152     else{
153
154         cout << endl << "Error: Invalid input!" << endl;
155
156     }
157
158
159 }
160
161 }
162
163
164 }
```

```

1 import random
2
3
4 # Our function for the player to input their guess.
5 def playerTurn(low, high):
6     output = 0
7
8     # We use a True loop since return will exit this function entirely.
9     while True:
10
11         # Checks for letters or other non-numeric inputs.
12         try:
13
14             output = int(input("Your turn! Enter your guess: "))
15
16         except ValueError:
17
18             print("\nInvalid input! Please enter a number between 10 and 99.\n")
19             continue
20
21         # Checks to make sure input is within bound.
22         if output >= high or output <= low:
23
24             print("\nInvalid input! Please enter a number between ", low, " and ", high, ".\n")
25
26
27         # If our input meets our bounds, we record it as the guess.
28         elif high > output > low:
29
30             return output
31
32
33 def computerTurn(low, high):
34     output = 0
35
36     # Generates a number between low + 1 and high - 1, to stay within bounds. Will only generate integers so no worries for error cases.
37     output = random.randint(low + 1, high - 1)
38     print("Computer guesses: ", output)
39
40     return output
41
42
43 def main():
44     playing = True
45     answer = 0
46     guessLog = 0
47     closingStatement = True
48     closeCheck = ""
49
50     # Uses a True loop because the function will either be infinitely repeated, or broken by a return when program is exited.
51     while True:
52
53         # We use a random number between 11 and 98 as these values are included in our random generation,
54         # and we do not want 10 or 99 as an option (unguessable.)
55         answer = random.randint(11, 98)
56         low = 10
57         high = 99
58         playing = True
59
60         print("\nWelcome to the Guessing Game!")
61         print("A secret 2-digit code has been set between 10 and 99.\n\n")
62
63         # This loop represents our gameplay, and the messages above are only repeated when this loop is broken (another game is started.)
64         while playing:
65
66             # We give the player the first turn, and then check their input. If the answer is correct in any case, there is a
67             # break and the loop ends.
68             guessLog = playerTurn(low, high)
69
70             if guessLog == answer:
71
72                 playing = False
73                 print("\nYou cracked the code!\n")
74                 break
75
76             elif guessLog < answer:
77
78                 print("\nToo low!\n")
79                 low = guessLog
80
81             else:
82
83                 print("\nToo high!\n")
84                 high = guessLog
85
86         # Follows the same check logic as above.
87         guessLog = computerTurn(low, high)
88
89         if guessLog == answer:
90
91             playing = False
92             print("\nComputer cracked the code!\n")
93             break
94
95         elif guessLog < answer:
96

```

```

97         print("\nToo low!\n")
98         low = guessLog
99
100     else:
101
102         print("\nToo high!\n")
103         high = guessLog
104
105     # After a game is over, turns the boolean to True, iterates through a loop until a sufficient response is given,
106     # then sets it to false to start a new game or returns to exit the program.
107     closingStatement = True
108
109     while closingStatement:
110
111         closeCheck = input("Would you like to play again? (y/n): ")
112
113         if closeCheck == "y" or closeCheck == "Y":
114
115             closingStatement = False
116             playing = False
117
118         elif closeCheck == "n" or closeCheck == "N":
119
120             exit()
121
122         else:
123
124             print("Error: Invalid input!\n")
125
126
127 main()

```