

### **2.3 random class**

Computer programs, particularly games, are much more fun if there's some randomness involved. Unfortunately, we have no reliable way of producing truly random numbers. However, most programming languages, including Python, include a pseudorandom number generator. These generators use a set of steps to generate numbers that appear random. Thus, if you or I knew the exact set of steps the generator was using, we could reproduce every number the generator created. However, to the casual observer, the numbers produced would appear random. Random numbers in programs allow us to play games with some uncertainty (think dice) and allow us to simulate real life events that have some uncertainty (think stock market). Python makes using random numbers fairly easy. In order to do so, we must do the following import:

```
import random
```

At the beginning of our program (preferably in main), we must seed our random number generator as follows:

```
random.seed()
```

For now, the details of this function call are not important. Simply include this as one of the first lines of your main function in any program that uses random numbers.

From this point on, if you want a random number selected between two integers a and b, inclusive, simply make the following function call:

```
random.randint(a, b)
```

Since this function call returns a value, like a majority of the math functions, we must call it as part of a greater line of code, typically storing its return value in a variable.

The following program will use the random number generator to generate one random number in between 1 and 100, and allow two players to guess the number. The winner will be the player who comes closest to the number without guessing too high. If both players guess too high, or if both players guess the same number, the outcome will be a tie. In all other cases there will be a unique winner.

Though this example generates only one random number, a program is allowed to generate many random numbers, if necessary. In these cases, we still only seed the random number generator only once.

```
import random
```

```
def main():
```

```
    random.seed()
```

```
    secretNum = random.randint(1,100)
```

```

# Get the user input.
guess1 = int(input("Player 1, enter your guess(1-100).\n"))
guess2 = int(input("Player 2, enter your guess(1-100).\n"))

# Calculate how close both players are.
diff1 = secretNum - guess1
diff2 = secretNum - guess2

print("The correct number was",secretNum)

# Check all of the cases!
if diff1 < 0 and diff2 < 0:
    print("Both players bust. The game is a tie.")
elif diff2 < 0:
    print("Player 1 wins since Player 2 busted.")
elif diff1 < 0:
    print("Player 2 wins since Player 1 busted.")
elif diff1 < diff2:
    print("Player 1 is closer to the correct number and wins!")
elif diff1 == diff2:
    print("Both players guessed the same number and tie.")
else:
    print("Player 2 is closer to the correct number and wins!")

main()

```

In this program, we generate a single random number in between 1 and 100, inclusive and store it in `secretNum`. After that, we use the `if` statement to separate out several different cases to determine which of the two players has won. As this example illustrates, a problem that seems so simple to us intuitively can have a rather complex, detailed solution. Though our brain can carry out this "Price is Right" logic effortlessly, we see that when formalized, there are several conditions that need to be checked.

The following section will contain an example that utilizes multiple random numbers in the same program.