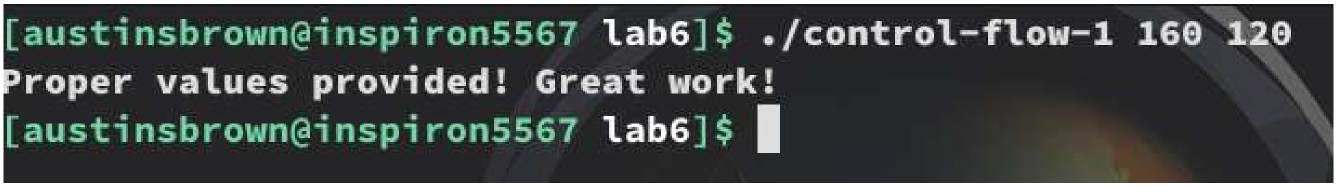
1. Control Flow

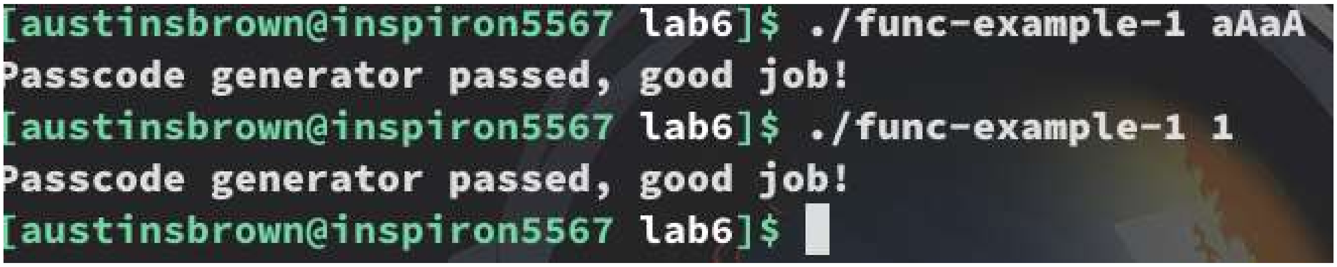
This program requires two integers as input. First it checks to see if the second arg is less than the first. It then checks to see if input2 * 2 is greater than input 1. The program then adds input1 and input2 together, subtracts by input1, and checks to see if the result is less than 100.



```
[austinsbrown@inspiron5567 lab6]$ ./control-flow-1 160 120
Proper values provided! Great work!
[austinsbrown@inspiron5567 lab6]$ 
```

2. func-example

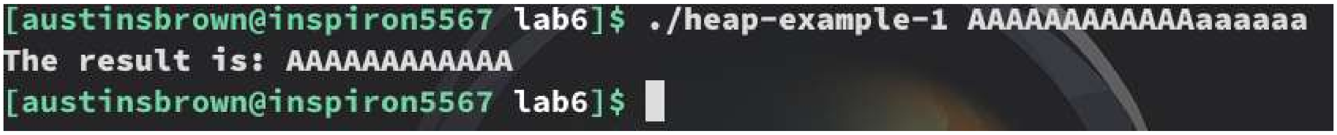
The program takes one string as input. It calls a function to extract the lowercase letters form the string and a function to extract the upper case letters. Strlen is called to check to see if the two are the same length. Supplying an integer also works since there are zero capital and lowercase letters.



```
[austinsbrown@inspiron5567 lab6]$ ./func-example-1 aAaA
Passcode generator passed, good job!
[austinsbrown@inspiron5567 lab6]$ ./func-example-1 1
Passcode generator passed, good job!
[austinsbrown@inspiron5567 lab6]$ 
```

3. heap-example

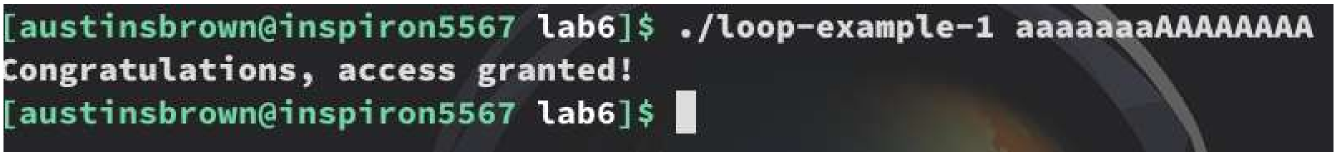
The program takes a string as input. It then checks to see if the input has exactly 12 capital letters. It does so by allocating space the size of the input string. The program iterates through the input character by character checking to see if the letter is capital. If so, it is added to the allocated string.



```
[austinsbrown@inspiron5567 lab6]$ ./heap-example-1 AAAAAAAAAAAAaaaaaa
The result is: AAAAAAAAAAAA
[austinsbrown@inspiron5567 lab6]$ 
```

4. loop-example

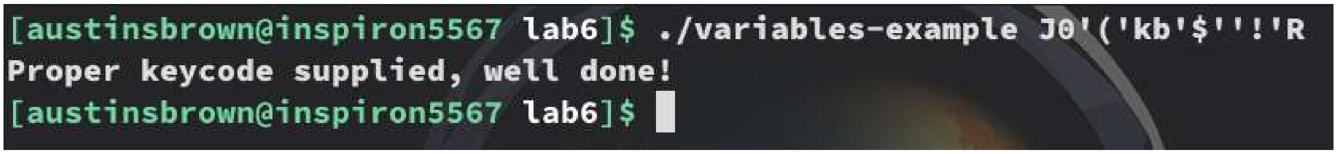
This program checks to see if an input string is exactly 15 characters long and has 8 capital letters.



```
[austinsbrown@inspiron5567 lab6]$ ./loop-example-1 aaaaaaaAAAAAAAA
Congratulations, access granted!
[austinsbrown@inspiron5567 lab6]$ 
```

5. variables-example

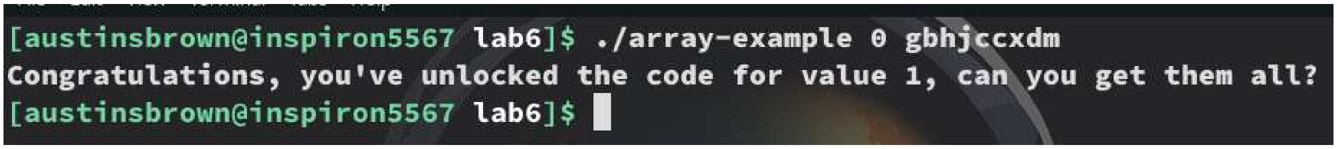
The program first check to see if the input string is at least 8 characters long. The program then goes into a while loop. The loop has an incrementor that goes from 0 to 8. The incrementor is multiplied by 8 and then right sifted by the constant 0xdeadbeeffacecafe. This is bit wise anded with the constant 0x3f. This is added to the character in the string “KeYpress” that corresponds to the incrementor. This is added to '\x01'. This is compared to the input character that corresponds to the incrementor.



```
[austinsbrown@inspiron5567 lab6]$ ./variables-example J0'('kb'$''!'R
Proper keycode supplied, well done!
[austinsbrown@inspiron5567 lab6]$ 
```

6. array-example

The program takes two inputs. An index and a pass code. The index is used to select a string from an array of strings. There are five strings in the array, so the index must be from 0 to 4. The strings are "hackadayu", "software", "reverse", "engineering", and "ghidra" in that order. The program then goes into a loop from from 0 to the length of the string that was chosen by the index. I’ll call the incrementor for the loop n. First the nth character of the string is stored to a variable. Then the n+1 character is stored to another variable. There is an if statement that makes sure that you don’t go out of bounds. These two characters are then compared. The greater one is subtracted by the lesser one. This ensures that the result won’t be negative. Then the result of this calculation is added to the character ‘`’ or 0x60. This is then compared to the user input.



```
[austinsbrown@inspiron5567 lab6]$ ./array-example 0 gbhjccxdm
Congratulations, you've unlocked the code for value 1, can you get them all?
[austinsbrown@inspiron5567 lab6]$ 
```