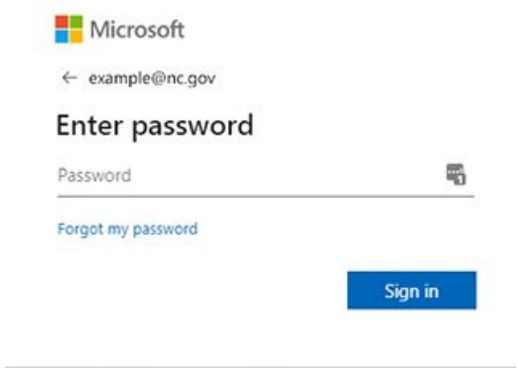
Project 2: Security Lock

By: Rose and Isaac

Goals of the Security Lock



We want this security lock to:

- Contain two inputs: the employee identification number (16 bits) and employee's password (4 bits)
- The company will contain 20 employees, and we need to construct the database to store that memory
- The output is one bit (0/1) that means (denied/allowed)

Starting at the Database

```
# Set entry as main
.globl main

.data
# Employee IDs
# 16 bits, min 0, max 65535
eids: .word 1, 4, 19, 240, 564, 46641, 6888, 43534, 16989, 12602, 48381, 56703, 54094, 26643, 53314, 24920, 8362, 25825, 42408, 4648

# Passwords
# 8 bits, min 0, max 255
pwds: .word 189, 18, 12, 53, 75, 123, 212, 32, 92, 11, 53, 54, 91, 223, 154, 177, 244, 88, 67, 69
```

- The first requirement that we must have is the data structure .word to store both the 16 bit employee IDs and the 8 bit passwords

- For the program to work, we also need to add the inputs for both the employee ID and the passwords from the user

```
# Inputs
ineid: .word 240
inpwd: .word 53
```

A look at main

```
.text
# Jump to main to avoid unwanted calls (because sometimes globl doesn't set entry point right)
j main
```

- The first thing we do after declaring main is to jump down to it

- After we make it to main, we take the inputs from the employee ID and the password, and jump to searching the EID list
- s3 is declared to help traverse the tree, as it represents the current loop address, for the find_eid function

```
# Starting position, sets up inputs and jumps to find_eid
main:
        # $s1 = input eid
        # $s2 = input pwd
        # $s3 = loop offset
        lw $s1, ineid
        lw $s2, inpwd
        li $s3, 0
        j find_eid
```

Confirming the EID

- The first thing that happens is the eid list being loaded
- After this, we traverse the tree by adding s3 to it, traversing the tree by passing through every ID and stores it in t2 to check
- s3 has four added to it, as it is the length of .word in bytes
- After checking t2 if it matches, it jumps to eid matched if it does
- If it doesn't match, then find_eid is called again

- When eid is matched, we load the password list, and then take the offset from the employee search and add s3 to it to find where the password is
- After going to the proper point in the list, it checks to see if it matches the list or not

```
# Loops through eids and checks for input eid. If found, jump eid_matched, else exit
find_eid:
        # Max loop offset in bytes
        li $t0, 80
        find_eid_loop:
        # If s3 >= 640, j exit
        bge $s3, $t0, exit

        # Load from eids with offset of s3 into t2.
        # We do this by loading the address of eids to t4, adding s3 to it, and then loading t4 to t2
        la $t4, eids
        add $t4, $t4, $s3
        lw $t2, ($t4)

        # If s1 == t2, j eid_matched
        beq $s1, $t2, eid_matched

        # Add 4 to s1 (length of word in bytes)
        addi $s3, 4

        # Jump start of loop
        j find_eid_loop


eid_matched:
        # Load from pwds with offset of s3 into t2.
        # We do this by loading the address of pwds to t4, adding s3 to it, and then loading t4 to t2
        li $t4, 0
        la $t4, pwds

        add $t4, $t4, $s3
        lw $t2, ($t4)

        # If s2 == t2, j success
        beq $s2, $t2, success

        # Else j exit
        j exit
```

Confirming the EID

- If we reach the end of the tree, the program is exited and we return zero and make a syscall to exit the program

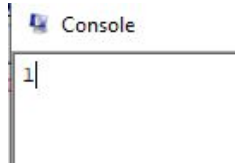
```
exit:
        # Load 4 to v0 (syscall for print) and 0 to a0 (auxilary for syscall)
        li $v0, 1
        li $a0, 0
        syscall

        # Load 10 to v0 (syscall for exit execution)
        li $v0, 10
        syscall
```

- If the program succeeds, we return a one and make a syscall to exit the program

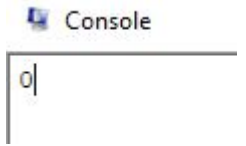
```
success:
        # Load 4 to v0 (syscall for print) and 1 to a0 (auxilary for syscall)
        li $v0, 1
        li $a0, 1
        syscall

        # Load 10 to v0 (syscall for exit execution)
        li $v0, 10
        syscall
```

Test Cases

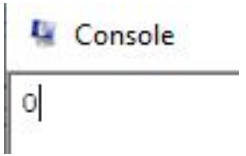
Inputs: 240, 53



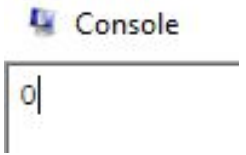
Inputs: 53, 240



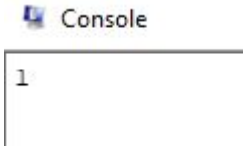
Inputs: 1, 69



Inputs: 564, 99999



Inputs: 564, 75



Have a Great Winter Break!