CSCU9V4 Practical 3
Where to begin, where to end?

Introduction

Before moving on to tackle some of the real quircks in C, it is important first to ensure familiarity with the basics. Today's goal: get past the agony of arrays so that they are second nature!

Background

Recall from lecture the program that looks for repeat digits in a sequence of digits,

```
#include <stdbool.h>   /* C99 only */
#include <stdio.h>

int main(void)
{
  bool digit_seen[10] = {false};
  int digit;
  long n;

  printf("Enter a number: ");
  scanf("%ld", &n);    /* Not '1', but lower-case L! */

  while (n > 0) {
    digit = n % 10;
    if (digit_seen[digit])   /* This conditional can be */
      break;                 /* re-written without the  */
    digit_seen[digit] = true;/* How might this be done? */
    n /= 10;
  }

  if (n > 0)
    printf("Repeated digit\n");
  else
    printf("No repeated digit\n");

  return 0;
}
```

Generating Useful Output

The default program only tells us if a digit has been repeated. Instead, copy (or retype) the source into your `repdigits.c` and modify the program so that it shows which digits (if any) were repeated:

```
Enter a number: 939577
Repeated digit(s): 7 9
```

HINT: the array no longer stores booleans.

"More Power!"

Repeat digits alone tell only part of the story. Save a new version of your source in `repdigit2.c`, and modify the program so that it prints a table showing the number of times each digit appears in the number:

```
Enter a number: 41271092
Digit:       0 1 2 3 4 5 6 7 8 9
Occurrences: 1 2 2 0 1 0 0 1 0 1
```

The First Step Towards Gamedom

Copy of your source into a new program, say `repdigit3.c,` and modify the program so that the user can enter as many numbers as desired for testing and evaluation. The program should terminate when the user enters a number that is less than or equal to `0`.

You're done!!

But if wanting additional fun…

How might the program be modified to use the expression `(int)(sizeof(a)/sizeof(a[0])` or a macro with this value? Make sure to try this out!

Could you declare and initialise a chess board? Try declaring an 8 x 8 char array that includes an initializer to put the starting positions of a chess game on the board (one char per array element):

```
r n b q k b n r
p p p p p p p p
  .   .   .   .
.   .   .   .
  .   .   .   .
.   .   .   .
P P P P P P P P
R N B Q K B N R
```