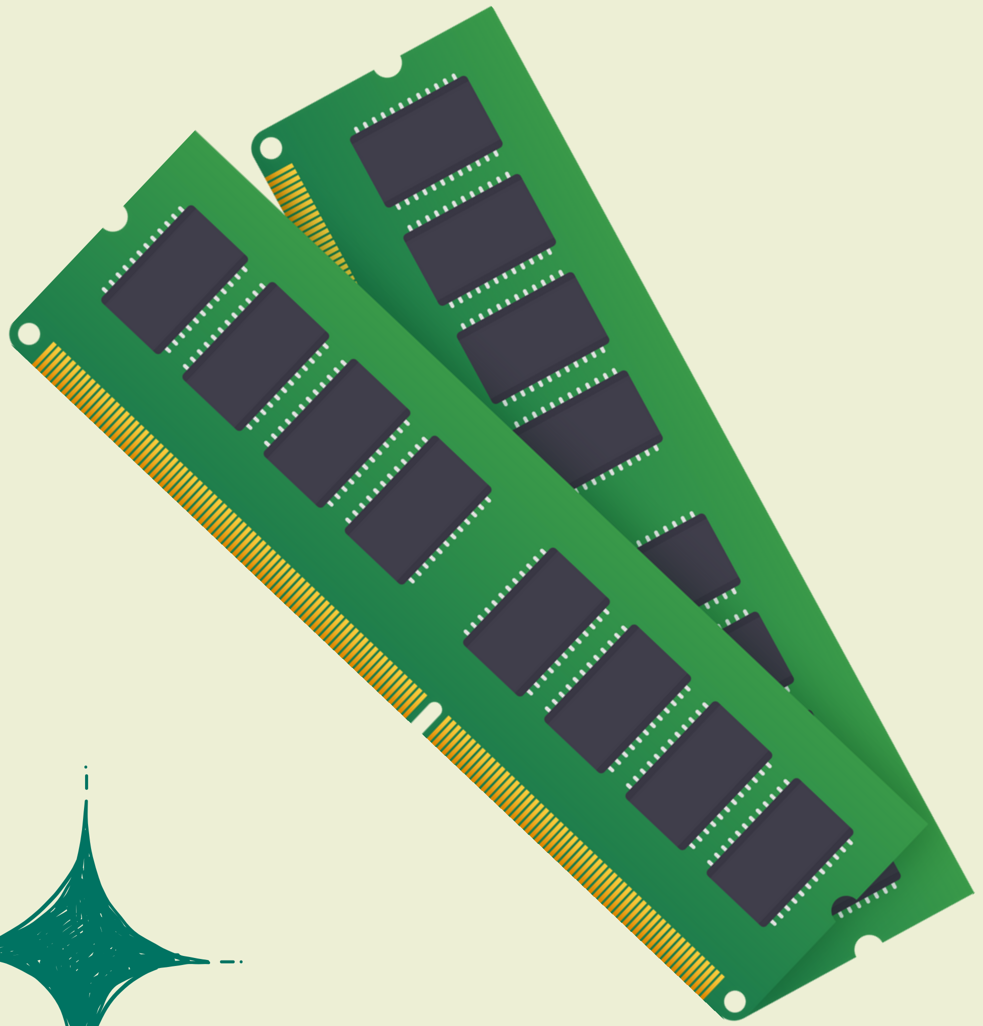


COMP1521 Memory

Ethan Haffenden

Quick Catch Up!

@EthanH-02

EthanH-02/comp1521

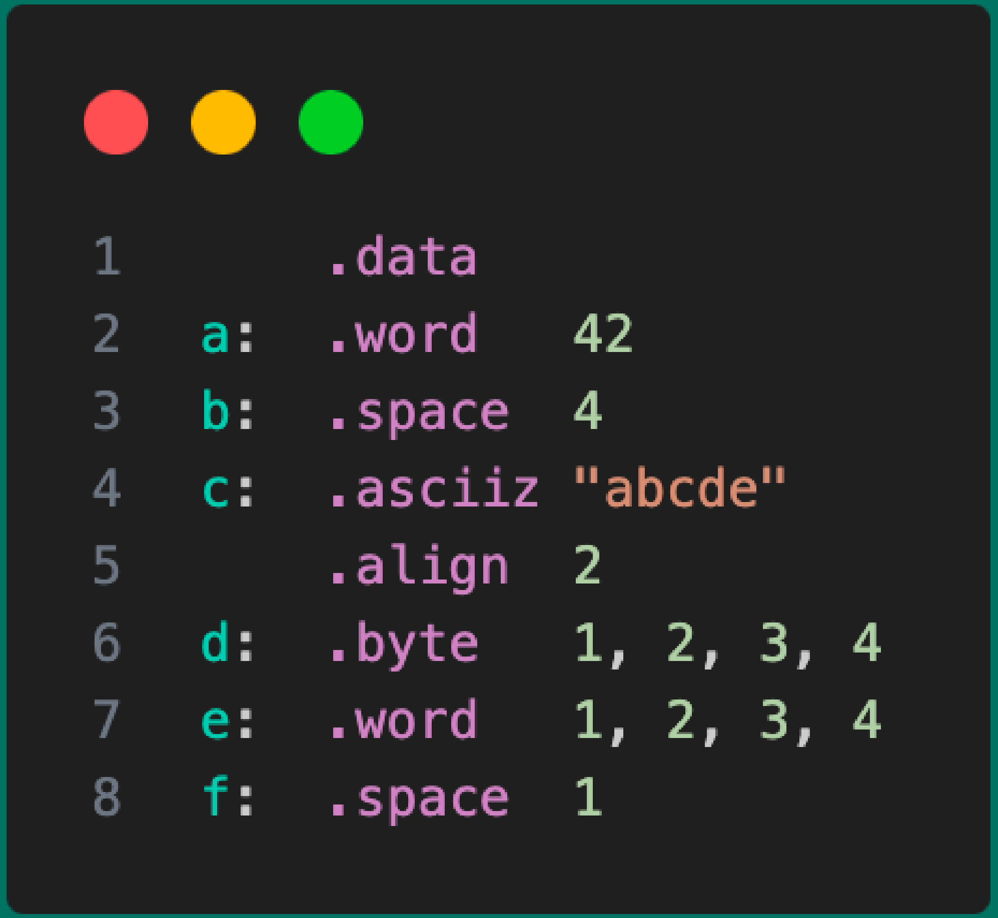
z5367576@ad.unsw.edu.au

Ethan

Often when writing large MIPS programs you will make errors that cause your program to misbehave. What tools are availble to help?

Question 2

If the following data segment of a MIPS program starts with the address
0x10010020
then what address are the following labels associated with and what is stored within each 4-byte memory cell?



```
      .data
a:    .word   42
b:    .space  4
c:    .asciiz "abcde"
      .align  2
d:    .byte   1, 2, 3, 4
e:    .word   1, 2, 3, 4
f:    .space  1
```

Question 3

Give MIPS directives to represent the following values:

a. `int u;`

b. `int v = 42;`

c. `char w;`

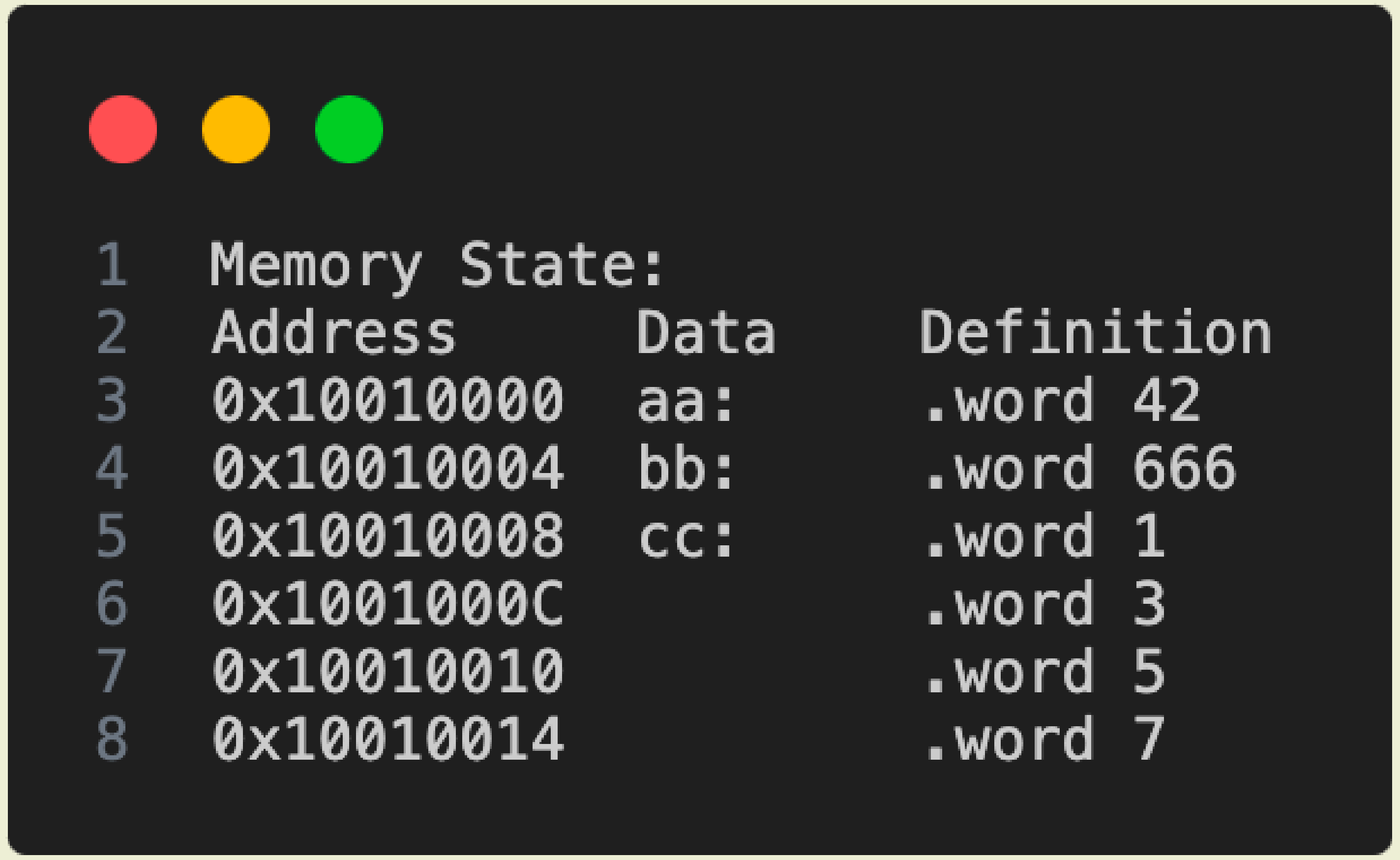
d. `char x = ‘a’;`

e. `double y;`

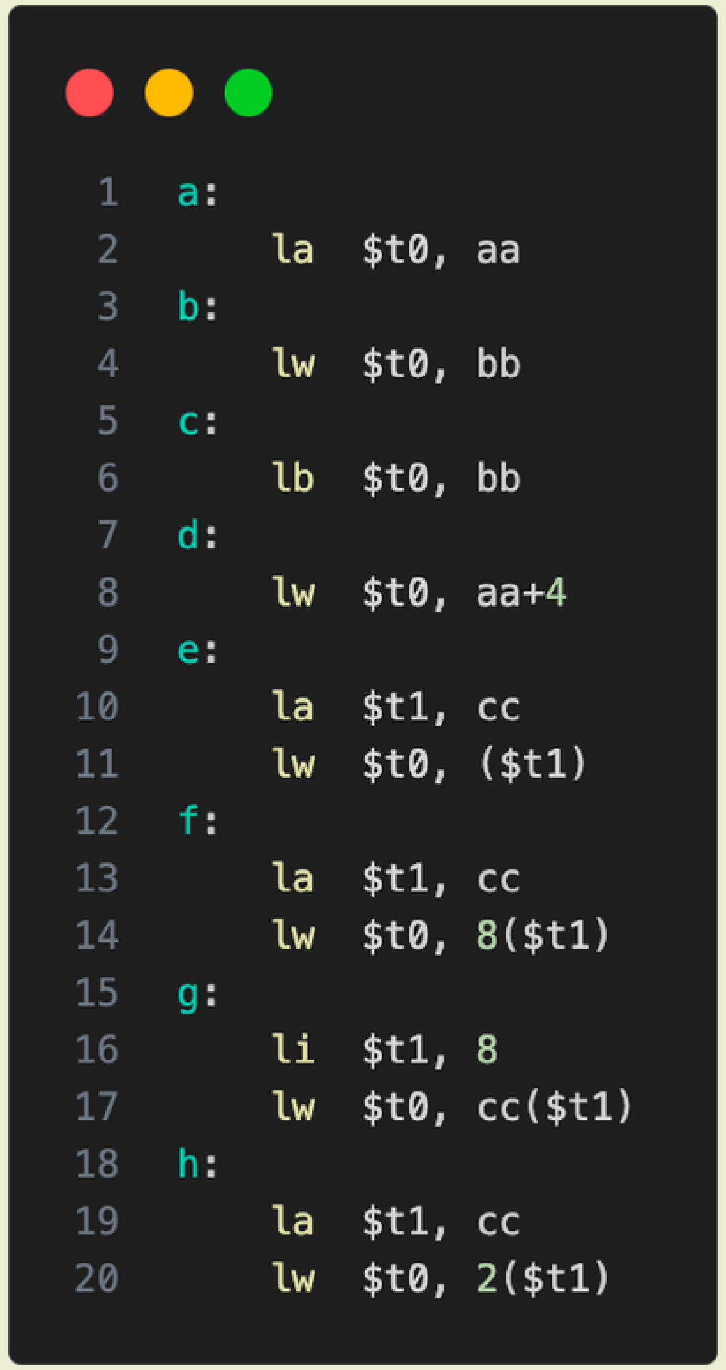
f. `int z[20];`

Question 4

Consider the following memory state, what addresses will be calibrated and loaded into the $t0 register, after each statement (or pairs of statements)



```
Memory State:
Address      Data     Definition
0x10010000   aa:      .word 42
0x10010004   bb:      .word 666
0x10010008   cc:      .word 1
0x1001000C            .word 3
0x10010010            .word 5
0x10010014            .word 7
```



```
a:
    la  $t0, aa
b:
    lw  $t0, bb
c:
    lb  $t0, bb
d:
    lw  $t0, aa+4
e:
    la  $t1, cc
    lw  $t0, ($t1)
f:
    la  $t1, cc
    lw  $t0, 8($t1)
g:
    li  $t1, 8
    lw  $t0, cc($t1)
h:
    la  $t1, cc
    lw  $t0, 2($t1)
```

Question 5
Translation

Question 6
Translation

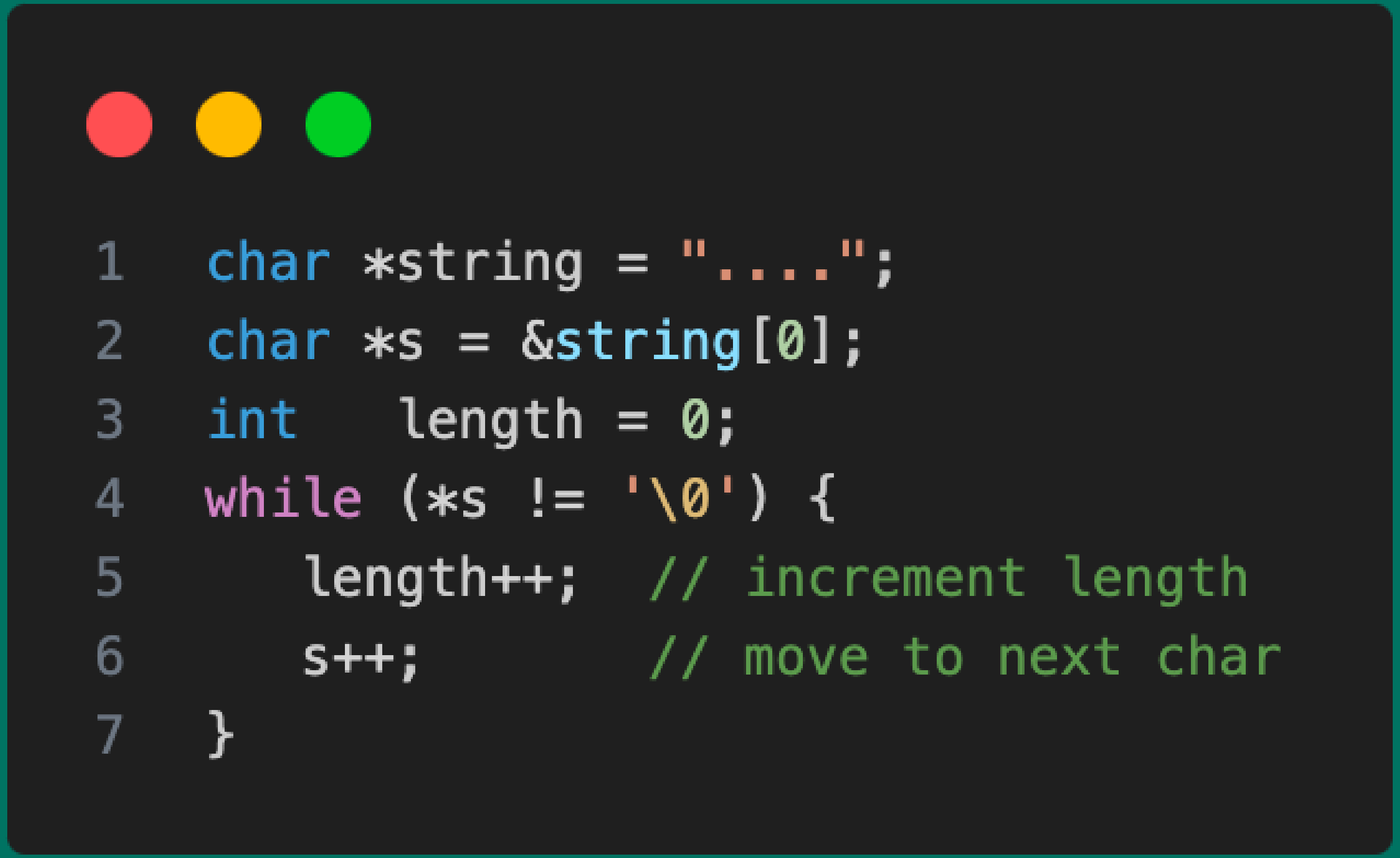
Question 8
Translation

Question 9.

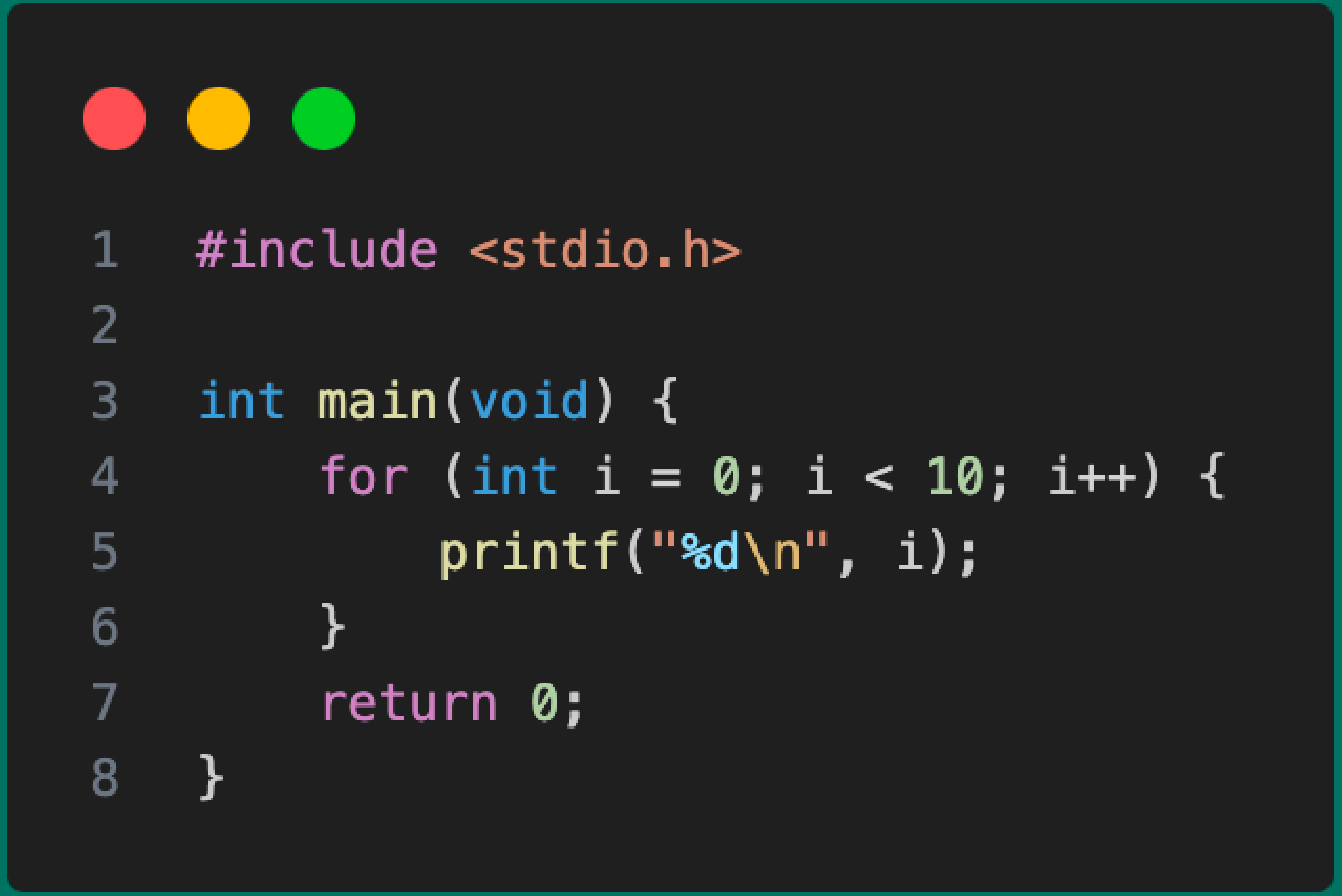
The loop attached determines the length of a string, a '\0' - terminated char array

Write MIPS assembly to implement this loop.

Assume s is implemented as $t0, and length is $t1. And assume that '\0' can be a value of 0

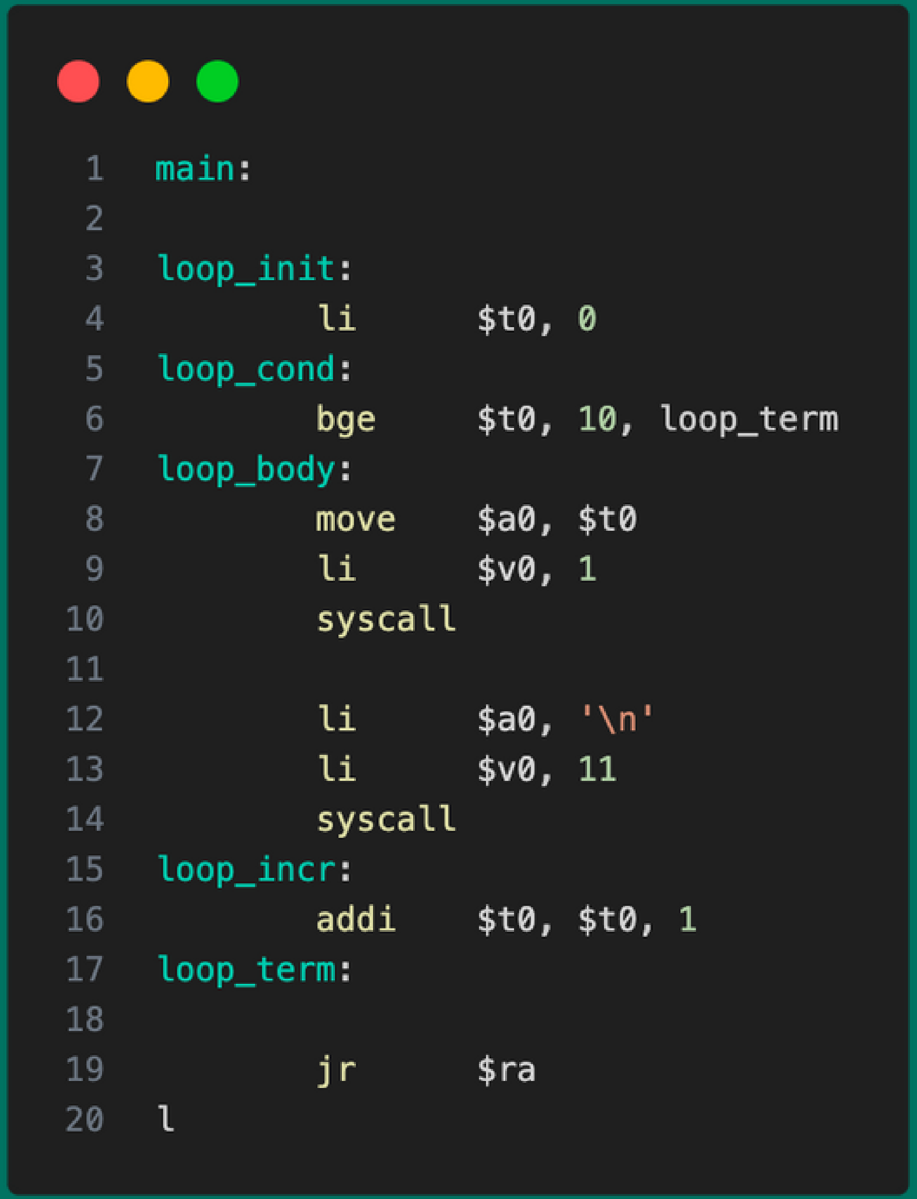


```c
char *string = "....";
char *s = &string[0];
int    length = 0;
while (*s != '\0') {
    length++;  // increment length
    s++;       // move to next char
}
```



```c
#include <stdio.h>

int main(void) {
    for (int i = 0; i < 10; i++) {
        printf("%d\n", i);
    }
    return 0;
}
```



```
main:

loop_init:
        li      $t0, 0
loop_cond:
        bge     $t0, 10, loop_term
loop_body:
        move    $a0, $t0
        li      $v0, 1
        syscall

        li      $a0, '\n'
        li      $v0, 11
        syscall
loop_incr:
        addi    $t0, $t0, 1
loop_term:

        jr      $ra
l
```