Computer Systems Fundamentals

yeet.c

A simple demonstration of goto in C.

```c
#include <stdio.h>

int main(void) {

    printf("In this essay I will\n");

    goto signature;

    printf("tell you why I deserve\n");
    printf("a 100 on this assignment.\n");
    printf("\n");

    printf("I have been working very\n");
    printf("hard on this assignment\n");
    printf("and I think I deserve a 100.\n");
    printf("\n");

    printf("I have been working on this\n");
    printf("assignment for a long time\n");
    printf("and I think I deserve a 100.\n");
    printf("\n");

    printf("It is my humble opinion\n");
    printf("that I deserve a 100 on\n");
    printf("this assignment.\n");
    printf("\n");

signature:
    printf("Kind regards,\n");
    printf("Abiram Nadarajah\n");
    printf("COMP1521 20T2 student\n");

    return 0;
}
```

print_if_even.c

Print a message only if a number is even.

```c
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 == 0) {
        printf("even\n");
    }

    return 0;
}
```

print_if_even.simple.c

Print a message only if a number is even.

```c
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 != 0) goto epilogue;
        printf("even\n");

epilogue:
    return 0;
}
```

print_if_even.s

Print a message only if a number is even.

```
        .text
main:
        # Locals:
        # - $t0: int n
        # - $t1: n % 2

        li      $v0, 4                  # syscall 4: print_string
        la      $a0, prompt_msg         #
        syscall                         # printf("Enter a number: ");

        li      $v0, 5                  # syscall 5: read_int
        syscall                         #
        move    $t0, $v0                # scanf("%d", &n);

        rem     $t1, $t0, 2             # if ((n % 2)
        bnez    $t1, epilogue   #     != 0) goto epilogue;

        li      $v0, 4                  # syscall 4: print_string
        la      $a0, even_msg           #
        syscall                         # printf("even\n");

epilogue:
        li      $v0, 0                  #
        jr      $ra                     # return 0;

        .data
prompt_msg:
        .asciiz "Enter a number: "
even_msg:
        .asciiz "even\n"
```

sum_100_squares.c

Calculate 1*1 + 2*2 + ... + 99*99 + 100*100

```c
#include <stdio.h>

int main(void) {
    int sum = 0;

    for (int i = 1; i <= 100; i++) {
        sum += i * i;
    }

    printf("%d\n", sum);

    return 0;
}
```

sum_100_squares.simple.c

Calculate 1*1 + 2*2 + ... + 99*99 + 100*100.

```c
#define UPPER_BOUND 100
#include <stdio.h>

int main(void) {
    int sum = 0;

loop_i_to_100__init:;
    int i = 0;
loop_i_to_100__cond:
    if (i > UPPER_BOUND) goto loop_i_to_100__end;
loop_i_to_100__body:
    sum += i * i;
loop_i_to_100__step:
    i++;
    goto loop_i_to_100__cond;
loop_i_to_100__end:

    printf("%d", sum);
    putchar('\n');

    return 0;
}
```

sum_100_squares.s

Calculate 1*1 + 2*2 + ... + 99*99 + 100*100

```
UPPER_BOUND = 100


        .text
main:
        # Locals:
        # - $t0: int sum
        # - $t1: int i
        # - $t2: temporary value

        li      $t0, 0                                  # int sum = 0;

loop_i_to_100__init:
        li      $t1, 1                                  # int i = 0;
loop_i_to_100__cond:
        bgt     $t1, UPPER_BOUND, loop_i_to_100__end    # while (i < UPPER_BOUND) {
loop_i_to_100__body:
        mul     $t2, $t1, $t1                           #   sum = (i * i) +
        add     $t0, $t0, $t2                           #         sum;
loop_i_to_100__step:
        addi    $t0, $t0, 1                             #   i++;
        b       loop_i_to_100__cond                     # }

loop_i_to_100__end:
        li      $v0, 1                                  # syscall 1: print_int
        move    $a0, $t0                                #
        syscall                                         # printf("%d", sum);

        li      $v0, 11                                 # syscall 11: print_char
        li      $a0, '\n'                               #
        syscall                                         # putchar('\n');

        li      $v0, 0
        jr      $ra                                     # return 0;
```

count_to_10.c

Count from 1 to 10 with a loop.

```c
#include <stdio.h>

int main(void) {
    for (int i = 1; i <= 10; i++) {
        printf("%d\n", i);
    }
    return 0;
}
```

count_to_10.simple.c

Count from 1 to 10 with a loop.

```c
#include <stdio.h>

int main(void) {

loop_i_to_10__init:;
    int i = 1;
loop_i_to_10__cond:
    if (i > 10) goto loop_i_to_10__end;

loop_i_to_10__body:
    printf("%d", i);
    putchar('\n');
loop_i_to_10__step:
    i++;                        // i = i + 1;
    goto loop_i_to_10__cond;

loop_i_to_10__end:
    return 0;
}
```

count_to_10.s

Count from 1 to 10 with a loop.

```
        .text
main:
        # Locals:
        # - $t0: int i

loop_i_to_10__init:
        li      $t0, 1                          # int i = 1;
loop_i_to_10__cond:
        bgt     $t0, 10, loop_i_to_10__end      # if (i > 10) goto loop_i_to_10__end;

loop_i_to_10__body:
        li      $v0, 1                          # syscall 1: print_int
        move    $a0, $t0                        #
        syscall                                 # printf("%d", i);

        li      $v0, 11                         # syscall 11: print_char
        li      $a0, '\n'                       #
        syscall                                 # putchar('\n');

loop_i_to_10__step:
        addi    $t0, $t0, 1                     # i = i + 1;
        b       loop_i_to_10__cond

loop_i_to_10__end:
        li      $v0, 0
        jr      $ra                             # return 0;
```

six.c

Print a message only if a number is divisible by 2 and 3.

```c
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 == 0 && n % 3 == 0) {
        printf("six\n");
    }

    return 0;
}
```

six.simple.c

Print a message only if a number is divisible by 2 and 3.

```c
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 != 0) goto epilogue;
    if (n % 3 != 0) goto epilogue;
        printf("six-ish\n");

epilogue:
    return 0;
}
```

six.s

Print a message only if a number is divisible by 2 and 3.

```
        .text
main:
        # Locals:
        # - $t0: int n
        # - $t1: n % 2
        # - $t2: n % 3

        li      $v0, 4                  # syscall 4: print_string
        la      $a0, prompt_msg         #
        syscall                         # printf("Enter a number: ");

        li      $v0, 5                  # syscall 5: read_int
        syscall                         #
        move    $t0, $v0                # scanf("%d", &n);

        rem     $t1, $t0, 2             # if ((n % 2)
        bnez    $t1, epilogue           #     != 0) goto epilogue;

        rem     $t2, $t0, 3             # if ((n % 3)
        bnez    $t2, epilogue           #     != 0) goto epilogue;

        li      $v0, 4                  # syscall 4: print_string
        la      $a0, six_msg            #
        syscall                         # printf("six-ish\n");

epilogue:
        li      $v0, 0                  #
        jr      $ra                     # return 0;

        .data
prompt_msg:
        .asciiz "Enter a number: "
six_msg:
        .asciiz "six-ish\n"
```

two_three.c

Print a message only if a number is divisible by 2 or 3.

```
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 == 0 || n % 3 == 0) {
        printf("two-three-ish\n");
    }

    return 0;
}
```

two_three.simple.c

Print a message only if a number is divisible by 2 or 3.

```c
#include <stdio.h>

int main(void) {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    if (n % 2 == 0) goto two_three_print;
    if (n % 3 == 0) goto two_three_print;
    goto epilogue;
two_three_print:
        printf("two-three-ish\n");

epilogue:
    return 0;
}
```

two_three.s

Print a message only if a number is divisible by 2 or 3.

```
        .text
main:
        # Locals:
        # - $t0: int n
        # - $t1: n % 2
        # - $t2: n % 3

        li      $v0, 4                  # syscall 4: print_string
        la      $a0, prompt_msg         #
        syscall                         # printf("Enter a number: ");

        li      $v0, 5                  # syscall 5: read_int
        syscall                         #
        move    $t0, $v0                # scanf("%d", &n);

        rem     $t1, $t0, 2             # if ((n % 2)
        beqz    $t1, two_three_print    #     == 0) goto two_three_print;

        rem     $t2, $t0, 3             # if ((n % 3)
        beqz    $t2, two_three_print    #     == 0) goto two_three_print;

        b epilogue                      # goto epilogue;

two_three_print:
        li      $v0, 4                  # syscall 4: print_string
        la      $a0, two_three_msg      #
        syscall                         # printf("two-three-ish\n");

epilogue:
        li      $v0, 0                  #
        jr      $ra                     # return 0;

        .data
prompt_msg:
        .asciiz "Enter a number: "
two_three_msg:
        .asciiz "two-three-ish\n"
```

forever_23.c

Example of break/continue use

```c
#include <stdio.h>

int main(void) {
    for (int n = 0; n < 100; n++) {
        if (n % 3 == 0) {
            continue;
        }
        if (n % 23 == 0) {
            break;
        }

        printf("%d\n", n);
    }

    return 0;
}
```

forever_23.simple.c

Example of break/continue use

```c
#include <stdio.h>

int main(void) {
    int n;

    n = 0;
forever_23_loop_top:
    if (n > 100) goto forever_23_loop_end;

        if (n % 3 == 0) goto forever_23_loop_next;

        if (n % 23 == 0) goto forever_23_loop_end;

        printf("%d", n);
        putchar('\n');

forever_23_loop_next:
    n = n + 1;
    goto forever_23_loop_top;

forever_23_loop_end:

    return 0;
}
```

forever_23.s

Example of break/continue use

```
        .text
main:
        # Locals:
        # - $t0: int n
        # - $t1: n % 2
        # - $t2: n % 23

forever_23_loop_init:
        li      $t0, 0                          # int n = 0;


forever_23_loop_top:

        rem     $t2, $t0, 3                     # if ((n % 3)
        beqz    $t2, forever_23_loop_next       #     == 0) goto forever_23_loop_next;

        rem     $t1, $t0, 23                    # if ((n % 23)
        beqz    $t1, forever_23_loop_end        #     == 0) goto forever_23_loop_end;

        li      $v0, 1                          # syscall 1: print_int
        move    $a0, $t0                        #
        syscall                                 # printf("%d", n);

        li      $v0, 11                         # syscall 11: print_char
        li      $a0, '\n'                       #
        syscall                                 # putchar('\n');

forever_23_loop_next:
        addi    $t0, $t0, 1                     #   n++;
        b forever_23_loop_top;                  # goto forever_23_loop_top;

forever_23_loop_end:

epilogue:
        li      $v0, 0                          #
        jr      $ra                             # return 0;
```

do_while.c

Do while example

```
#include <stdio.h>

int main(void) {
    int val;
    do {
        printf("val? ");
        scanf("%d", &val);
        printf("%d",val);
        printf("\n");
    } while (val > 0);
}
```

do_while.s

read/write characters until the user types a '!'

val is represented by $t0

```
    .text
main:

loop_start:             # do {

    la   $a0, prompt    #   printf("val? ");
    li   $v0, 4
    syscall

    li   $v0, 5         #   scanf("%d",&val);
    syscall
    move $t0,$v0

    move $a0, $t0       #   printf("%d",val);
    li   $v0, 1
    syscall

    li   $a0, '\n'      #   printf("\n");
    li   $v0, 11
    syscall

    blt  $t0,1,loop_end # } while (val > 0);
    b    loop_start
loop_end:

    li   $v0, 0         # return 0
    jr   $ra

# read 10 numbers into an array then print the 10 numbers

    .data

prompt:
    .asciiz "val? "
```