```
.data

hellow: .asciiz "Hello world!\n"

.text

main:
  jal printhello
  jal printhello
  jal printhello

  li $v0, 10
  syscall

################## FUNCTIONS ###############

printhello:
  li $v0, 4
  la $a0, hellow
  syscall
  jr $ra      # return
```

```
.data

newline: .asciiz "\n"

.text

main:
  li $t0, 1

loop:
  bgt $t0, 10, ready
  move $a0, $t0
  jal square
  move $a0, $v0
  li $v0, 1
  syscall
  li $v0, 4
  la $a0, newline
  syscall
  addi $t0, $t0, 1
  j loop

ready:
  li $v0, 10
  syscall

################## FUNCTIONS ###############

square:
##############################################
#  Square of argument                        #
#  Input:   $a0: int                         #
#  Output:  $v0: int                         #
##############################################
  mul $v0, $a0, $a0
  jr $ra    # return
```

```
########### MACROS: ########################

.macro prints (%p) ## print string %p ##
  li $v0, 4
  la $a0, %p
  syscall
.end_macro

.macro printLF ## print newline ##
  prints newline
.end_macro

.macro printintln (%p) ## print as integer ##
  move $a0, %p
  li $v0, 1
  syscall
  printLF
.end_macro

.macro boe (%p1, %p2) ## branch-on-even ##
  and $at, %p1, 1
  beqz $at, %p2
.end_macro

.macro returnToOS ## return to OS ##
  li $v0, 10
  syscall
.end_macro

########### DATA SEGMENT: ##################

.data

newline: .asciiz "\n"

########### CODE SEGMENT: ##################

.text

########### MAIN: ###########################

main:
  li $t0, 1           # $t0: 'i'

loop:                 # for ($t0=1; $t0<=10; $t0++)
  bgt $t0, 10, exitloop
  boe $t0, continue   # if ($t0%2==0) continue;
  move $a0, $t0
  jal square          # $v0 = square($a0);
  printintln $v0      # printf("%d\n", $v0);
continue:
  addi $t0, $t0, 1    # $t0++;
  j loop
exitloop:
  returnToOS          # back to OS

########### FUNCTIONS: ######################

square:
###############################################
#  Square of argument                         #
#  Input:   $a0: int                          #
#  Output:  $v0: int                          #
###############################################
  mul $v0, $a0, $a0
  jr $ra               # return
```