**Decaf Program:**

void main() {

 int c;

 string s;

 s = "hello";

 c = test(4, 5);

 Print(c);

 Print(s);

}

int test(int a, int b) {

 return a + b;

}

**TAC Output:**

main:

 BeginFunc 24 ;

 \_tmp0 = "hello" ;

 s = \_tmp0 ;

 \_tmp1 = 4 ;

 \_tmp2 = 5 ;

 PushParam \_tmp2 ;

 PushParam \_tmp1 ;

 \_tmp3 = LCall \_test ;

 PopParams 8 ;

 c = \_tmp3 ;

 PushParam c ;

 LCall \_PrintInt ;

 PopParams 4 ;

 PushParam s ;

 LCall \_PrintString ;

 PopParams 4 ;

 EndFunc ;

\_test:

 BeginFunc 4 ;

 \_tmp4 = a + b ;

 Return \_tmp4 ;

 EndFunc ;

**MIPS Output:**

 # standard Decaf preamble

 .text

 .align 2

 .globl main

 main:

 # BeginFunc 24

 subu $sp, $sp, 8 # decrement sp to make space to save ra, fp

 sw $fp, 8($sp) # save fp

 sw $ra, 4($sp) # save ra

 addiu $fp, $sp, 8 # set up new fp

 subu $sp, $sp, 24 # decrement sp to make space for locals/temps

 # \_tmp0 = "hello"

 .data # create string constant marked with label

 \_string1: .asciiz "hello"

 .text

 la $t2, \_string1 # load label

 sw $t2, -16($fp) # spill \_tmp0 from $t2 to $fp-16

 # s = \_tmp0

 lw $t2, -16($fp) # fill \_tmp0 to $t2 from $fp-16

 sw $t2, -12($fp) # spill s from $t2 to $fp-12

 # \_tmp1 = 4

 li $t2, 4 # load constant value 4 into $t2

 sw $t2, -20($fp) # spill \_tmp1 from $t2 to $fp-20

 # \_tmp2 = 5

 li $t2, 5 # load constant value 5 into $t2

 sw $t2, -24($fp) # spill \_tmp2 from $t2 to $fp-24

 # PushParam \_tmp2

 subu $sp, $sp, 4 # decrement sp to make space for param

 lw $t0, -24($fp) # fill \_tmp2 to $t0 from $fp-24

 sw $t0, 4($sp) # copy param value to stack

 # PushParam \_tmp1

 subu $sp, $sp, 4 # decrement sp to make space for param

 lw $t0, -20($fp) # fill \_tmp1 to $t0 from $fp-20

 sw $t0, 4($sp) # copy param value to stack

 # \_tmp3 = LCall \_test

 jal \_test # jump to function

 move $t2, $v0 # copy function return value from $v0

 sw $t2, -28($fp) # spill \_tmp3 from $t2 to $fp-28

 # PopParams 8

 add $sp, $sp, 8 # pop params off stack

 # c = \_tmp3

 lw $t2, -28($fp) # fill \_tmp3 to $t2 from $fp-28

 sw $t2, -8($fp) # spill c from $t2 to $fp-8

 # PushParam c

 subu $sp, $sp, 4 # decrement sp to make space for param

 lw $t0, -8($fp) # fill c to $t0 from $fp-8

 sw $t0, 4($sp) # copy param value to stack

 # LCall \_PrintInt

 jal \_PrintInt # jump to function

 # PopParams 4

 add $sp, $sp, 4 # pop params off stack

 # PushParam s

 subu $sp, $sp, 4 # decrement sp to make space for param

 lw $t0, -12($fp) # fill s to $t0 from $fp-12

 sw $t0, 4($sp) # copy param value to stack

 # LCall \_PrintString

 jal \_PrintString # jump to function

 # PopParams 4

 add $sp, $sp, 4 # pop params off stack

 # EndFunc

 # (below handles reaching end of fn body with no explicit return)

 move $sp, $fp # pop callee frame off stack

 lw $ra, -4($fp) # restore saved ra

 lw $fp, 0($fp) # restore saved fp

 jr $ra # return from function

 \_test:

 # BeginFunc 4

 subu $sp, $sp, 8 # decrement sp to make space to save ra, fp

 sw $fp, 8($sp) # save fp

 sw $ra, 4($sp) # save ra

 addiu $fp, $sp, 8 # set up new fp

 subu $sp, $sp, 4 # decrement sp to make space for locals/temps

 # \_tmp4 = a + b

 lw $t0, 4($fp) # fill a to $t0 from $fp+4

 lw $t1, 8($fp) # fill b to $t1 from $fp+8

 add $t2, $t0, $t1

 sw $t2, -8($fp) # spill \_tmp4 from $t2 to $fp-8

 # Return \_tmp4

 lw $t2, -8($fp) # fill \_tmp4 to $t2 from $fp-8

 move $v0, $t2 # assign return value into $v0

 move $sp, $fp # pop callee frame off stack

 lw $ra, -4($fp) # restore saved ra

 lw $fp, 0($fp) # restore saved fp

 jr $ra # return from function

 # EndFunc

 # (below handles reaching end of fn body with no explicit return)

 move $sp, $fp # pop callee frame off stack

 lw $ra, -4($fp) # restore saved ra

 lw $fp, 0($fp) # restore saved fp

 jr $ra # return from function