**Question 1.** Translate the following C-style code into equivalent 4 lines of MIPS assembly code. You are allowed to use the following registers: $t0, $t1, $t2 in your code

|  |
| --- |
| int a=9, b=5;  int c=a+4\*b; |

Reference code:

|  |
| --- |
| li $t0, 9  li $t1, 5  sll $t1, $t1, 2  add $t2, $t1, $t0 |

**Question 2.** Convert the following C-style code into equivalent MIPS assembly code. Suppose x and y are inputs by users. Suppose x<10 and y<10.

|  |
| --- |
| int x, y;  int count=0, sum=0;  while(count<x){  sum=sum+y;  count=count+1;  } |

Reference code:

|  |
| --- |
| .text  li $v0, 5 # read input x, stored in $t0  syscall  move $t0, $v0  li $v0, 5 # read input y, stored in $t1  syscall  move $t1, $v0  li $t2, 0 # store the value of count  li $t3, 0 # store the sum  while:  beq $t2, $t0, exit  add $t3, $t3, $t1  addi $t2, $t2, 1  j while  exit:  li $v0, 10 # exit  syscall |

**Question 3.** Write MIPS assembly code that do the computation: F=|A-B|+|C-D|. Suppose the value A is in register $t0, B is in register $t1, C is in register $t2, D is in register $t3, and the result F should be stored in register $t4.

Reference code:

|  |
| --- |
| .text  move $a1, $t0  move $a2, $t1  jal calculate # compute |A-B|    move $t7, $v0 # store |A-B| in t7  move $a1, $t2  move $a2, $t3  jal calculate # compute | C-D |    move $t8, $v0 # store |C-D| in t8  add $t4, $t7, $t8  j exit  calculate:  sub $t5, $a1, $a2 # compute $a1-$a2  move $t6, $t5  srl $t6, $t6, 31 # focus on the highest bit  bnez $t6, negate # if highest bit is 1, then jump  move $v0, $t5  jr $ra  negate:  sub $v0, $zero, $t5  jr $ra  exit:  li $v0, 10 # exit  syscall |

**Question 4.** Write MIPS assembly code to print “It is positive!” if the value in register $t0 is greater than 0, and to print “It is negative!” if the value in register $t0 is less than 0. Suppose the value in register $t0 is input by the user.

Reference code:

|  |
| --- |
| .data  pos: .asciiz “It is positive!\n”  neg: .asciiz “It is negative!\n”  .text  li $v0, 5 # read input value, stored in $t0  syscall  move $t0, $v0  bgtz $t0, print1 # the value in $t0 is greater than 0  blez $t0, print2 # the value in $t0 is not greater than 0  print1:  li $v0, 4 # print “It is positive!\n”  la $a0, pos  syscall  j exit  print2:  beq $t0, 0, exit # the value in $t0 is equal to 0  li $v0, 4 # print “It is negative!\n”  la $a0, neg  syscall  exit:  li $v0, 10 # exit  syscall |