# 1. Convert the following Java/C mathematical statements to MIPS assembly language.

## a = b + c \* d – e

We will assume the following:

a = $s0

b = $s1

c = $s3

d = $s4

e = $s5

mul $s0, $s3, $s4

add $s0, $s1, $s0

sub $s0, $s0, $s5

## a = a – 3

We will assume the following:

a = $s0

subi $s0, $s0, -3

## x = (y + z) \* (z - y)

We will assume the following:

X = $s0

Y = $s1

Z = $s2

We will be using one (1) temporary variable, $t0

Sub $t0, $s2, $s1

Add $s0, $s1, $s2

Mul, $s0, $s0, $t0

## f = (1 - x) \* (1 + x) + 100\*(y + x\*x)\*(y - x\*x)

1 is stored in $s0

100 is stored in $s1

X is $s2

Y is $s3

F will be stored in $s4

We will be using the following temporary variable:

mul $t0, $s2, $s2 # x^2

sub $t1, $s3, $t0 # y – x^2

add $t0, $s3, $t0 # y + x^2

mul $t0, $t0, $t1 # (y+x^2) \* (y-x^2)

mul $t0, $s1, $t0 # 100 \* <above>

addi $t1, $s2, 1 # 1 + x

sub $t2, $s0, $s2 # 1 – x

mul $t2, $t2, $t1 # (1-x)(1+x)

add $s4, $t2, $t0 # store additions into ‘f’

# 2. Showing all work, convert the following numbers:

## 1101001 from binary to decimal (hint: use powers of two)

Value = (1 \* 2^6) + (1 \* 2^5) + (0 \* 2^4) + (1 \* 2^3) + (0 \* 2^2) + (0 \* 2^1) + (1 \* 2^0)

= (1 \* 64) + (1 \* 32 ) + (0 \* 16) + (1 \* 8) + (0 \* 4) + (0 \* 2) + (1 \* 1)

= 64 + 32 + 0 + 8 + 0 + 0 + 1

= 105

## 0111 1100 0100 1001 0110 1010 1011 1101 from binary to hexadecimal

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Binary | 0111 | 1100 | 0100 | 1001 | 0110 | 1010 | 1011 | 1101 |
| Hex | 7 | C | 4 | 9 | 6 | A | B | D |

0b0110,1100,0100,1001,0110,1010,1011,1101 = 0x7C496ABD

## 18324 from decimal to binary (hint: use division by two)

18324 / 2 = 9162 r0

9162 / 2 = 4581 r0

4581 / 2 = 2290 r1

2290 / 2 = 1145 r0,

1145 / 2 = 572 r1

572 / 2 = 286 r0

286 / 2 = 143 r0

143 / 2 = 71 r1,

71 / 2 = 35 r1

35 / 2 = 17 r1

17 / 2 = 8 r1

8 / 2 = 4 r0,

4 / 2 = 2 r0

2 / 2 = 1 r0

1 / 2 = 0 r1

0d18324 = 0b 100,0111,1001,0100

## Convert the result from c) to hexadecimal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Binary | 0100 | 0111 | 1001 | 0100 |
| Hex | 4 | 7 | 9 | 4 |

0b0100,0111,1001,0100 = 0x4794

# 3. Assume you have an array of 10 integers declared and initialized using the following C statement:

# int arr[10] = { 20, 18, 16, 14, 12, 10, 8, 6, 4, 2 };

# Write corresponding assembly statements for the following C statements assuming the starting address of arr is stored in $s1:

## int x = arr[2] + arr[6]; //assume x is a register

Assume x will be stored in $s0.

Add $s0, 2($s1), 6($s1)

## arr[7] = arr[4];

This will utilize a temporary register location, $t0

Lw 4($s1), $t0

Sw 7($s1), $t0

## arr[0] = arr[3] + arr[6] \* 3;

$t0 will contain the constant 3

$t1, $t2 will also be used for this example

Lw $t1, 6($s1) # load arr[6]

Mult $t1, $t1, $t0 # multipley arr[6] by 3

Lw $t2, 3($s1) # load arr[3]

Add $t1, $t2, $t1 # add arr[3] and (arr[6] \* 3) together

Sw $t1, 0($s1) # store result in arr[0]