Variant 4

6.1903: Quiz 2 (11/2/2026)

Name:	**Kerberos:**

Quiz content is on **BOTH SIDES** of the quiz sheet. Enter your answers in the boxes designated for each problem. Additional scratch paper can be provided but will not be considered in grading.

1. Consider the following RISC-V assembly code. What would the value of `x8` be after executing this code sequence? Assume that beginning at address `0x400` there is an array of 10 integers where `a[0]=5, a[1]=6, a[2]=7` and so on with each value being 1 larger than the previous array element.

```
    li x1, 0
    addi x2, zero, 2
    li x8, 0
    j comp
loop: slli x4, x1, 2
    lw x6, 0x400(x4)
    add x7, x6, x2
    add x8, x8, x7
    addi x1, x1, 1
comp: blt x1, x2, loop
```

What would the value of `x8` be after executing this code sequence?

```
x8 = 0x
```

2. Consider the following C function and its corresponding RISC-V assembly code implementation.

```
int fun(int x, int y) {
    return x + y;
}
```

```
fun:
    add s4, a0, a1
    ret
```

List all registers that are handled in a way that violates the RISC-V calling convention. Enter NONE if there are no violations of the calling convention.

3.

A. What RISC-V instruction does the following encoding represent? If the instruction has an immediate value, then specify it in **hexadecimal**.

Instruction encoding: `0b0110_0011_1010_0100_0111_0011_1011_0111`

What is the assembly instruction that the above encoding represents?	

B. Given the following RISC-V assembly instruction, provide the 32-bit encoding of this instruction in hexadecimal notation. Assume that the label `dest` is **16** bytes after the address of the `bgeu` instruction.

RISC-V instruction: `bgeu x3, x7, dest`

32-bit encoding of the instruction in hexadecimal notation.	`0x`

4. Specify the resulting values of the requested registers and memory locations after executing the code below. Assume that the code halts after executing the `sw x2, 8(x1)` instruction.

```
. = 0x0
  li x1, 0x100
  li x2, 0x1234
  lw x4, 4(x1)
  srai x5, x4, 4
  xori x2, x2, 0xFF
  sw x2, 8(x1)

. = 0x100
.word 0x12345678
.word 0xDEDEDEDE
.word 0x87878787
```

`x5 = 0x`

`Mem[0x108] = 0x`