HONOR CODE

- I have not used any online resources during the exam.
- I have not obtained any help either from anyone in the class or outside when completing this exam.
- No sharing of notes/slides/textbook between students.
- NO SMARTPHONES.
- **CANVAS ANSWERS WILL BE LOCKED AFTER 1ST TRY.**

Questions Sheet.

Read all of the following information before starting the exam:

- For each question fill out the appropriate choice or write text on Canvas page. Also type clearly on in the exam on the appropriate text.
- IF THE MULTIPLE CHOICE ANSWER IS WRONG WE WILL MARK THE ANSWER WRONG. IF THE MULTIPLE-CHOICE ANSWER IS CORRECT, WE WILL READ THE WRITTEN PORTION.
- Show all work, clearly and in order, if you want to get full credit.
- We reserve the right to take off points if we cannot see how you logically got to the answer (even if your final answer is correct).
- Circle or otherwise indicate your final answers.
- Please keep your written answers brief; be clear and to the point.
- I will take points off for rambling and for incorrect or irrelevant statements.

A. Easy. Lets C. [6 Points]

Q1-6

Grayscale color values can be represented as an ascii value between 0---255. Consider square images of $N \times N$ pixels.
We can organize these 2D images into a 1D array of N^2 elements.
Each element is an 8 bit number.

```
char *img = malloc(sizeof(char) * 4);
img[0] = 0xA;
img[1] = 0xB;
img[2] = 0xC;
img[3] = 0xD;
```

Fill out the following function `tile` . It returns a new, larger image array, which is the same image tiled rep times in both the x and y direction. You may or may not need all of the lines. For a better idea of what must be accomplished, consider the above example:

```
char *t_img = tile(img, 2, 2);
// The contents of tiled_image
would then look like:
[0xA, 0xB, 0xA, 0xB
0xC,  0xD, 0xC, 0xD
0xA,  0xB, 0xA, 0xB
0xC,  0xD, 0xC, 0xD];
```

Now fill-in-the blanks for the code shown below on canvas.

```
char *tile(char*b, int n, int rep) {
  int w =_______Q1_____________________;
  char *t_img = malloc(__________Q2______);
  for (int j = 0; j < w; j++) {
        for (int i = 0; i < w; i++) {
            int x = ___Q3____;
            int y = ___Q4____;
            int loc = _Q5____;
            t_img[loc] = b[x + y*n];
            }
    }
    _______Q6 (could be multiple lines)______;
    return t_img;
}
```

B. Easy. RISCV Blackbox. [7 Points]

Assume we have two arrays input and output.

```
int *input = malloc(8*sizeof(int));
int *result = malloc(8, sizeof(int));
for (int i = 0; i < 8; i++) {
    input[i] = i;
}
```

Study the following RISC-V code shown below and answer the questions.
You can assume a0:input a1:result a2:8

```
main:
  .....
  addi a2, zero, 8
#  Q7 What registers need to be stored onto the stack?
  jal ra, BLACKBOX   # a0 holds input, a1 holds result a2 holds 8.
  # CHECK  finished calling BLACKBOX...
.... # Other code and function calls.
exit:
  addi a0, x0, 10
  add  a1, x0, x0
  ecall            # Terminate ecall

BLACKBOX:
# Q8 What registers need to be stored onto the stack?
  mv s0, zero
  mv s1,a1
  mv t0, zero
loop:
  beq t0, a2, done
  lw t1, 0(a0)
  add s0, s0, t1
  slli t2, t0, 2
  add t2, t2, s1
  sw s0, 0(t2)
  addi t0, t0, 1
  addi a0, a0, 4
  jal x0, loop
done:
  mv a0, s0
# Q9 TODO: epilogue.  What registers need to be restored?
jr ra
```

7. What is the minimum set of registers need to be stored onto the stack at this point: line 4. ? [1]

8. What is the minmum set of registers need to be stored onto the stack at this point: line 14. ? [1]

9. What is the minmum set of registers need to be restored from the stack at this point: line 30 ? [1]

10. Assume you have the prologue and epilogue correctly coded. You set a breakpoint at `line 6: CHECK". What does result contain when your program pauses at the breakpoint? [4]

C. RISC-V Instructions Encoding [5 points]

Consider the standard RISC-V encoding below. Standard 32 bit instructions. Answer questions below

```
1  main:
2          mv s1, a0
3          addi t2, t2 4
4  Start:
5          beq s1, x0, end
6    lw a0, 0(s1)
7          addi a0,a0,4
8          add s1, t2, s1
9    lw s1, 0(s1)
10   jal x0, Start
11 end:
12         addi a0,a0,10
13   ecall
```

11. For the instruction line 5: `beq s1, x0, end` . What is the immediate field [1]

12. What is actual opcode, rs1 and rs2 (not pseudo-names) ? [1]

13. What is funct7 and funct3 ? [1]

14. What is the instruction corresponding to `0x0004A503` ? [2]

D. RISC-V Custom Opcodes [6]

Prof. Shriraman is designing a new CPU with fewer operations. He decides to adapt and rethink the design of RISC-V instruction. He only needs to support 9 different operations: ADD, MUL, XOR, LD, SW, LUI, ADDI, MULI and BLT. He decides that each instruction should be 19 bits wide.

The fields in each instruction are listed below (no funct3 and funct7)

- R-type: rs2,rs1,rd,opcode
- I-type and Loads: imm,rs1,rd,opcode
- S-type: imm,rs2,rs1,opcode
- B-type: imm,rs2,rs1,opcode
- U-type: imm,rd,opcode
- UJ-type: imm,rd,opcode

15. What is the minimum bits would be required for the opcode field? [1]

16. If the opcode was encoded 5 bits and we would like to support the usual R-type instructions, 2 source and 1 destination. What is the maximum number of registers we can use [1]

17. Given the opcode is 5 bits wide. Each register field is 3 bits. What is the offset in terms of bytes that the branch instruction can jump. Note that instructions are 19 bits wide. [2]

18. What is the smallest negative offset that an I instruction can use ? Opcode bits is same as Q15. Assume that register width is same as Q16. [2]

E. Easy Floating Point [5 points]

The TAs get tired of having to convert floating-point values into 32 bits. As a result they propose the following smaller floating-point representation which is useful in a number of machine learning applications. It consists of a total of 10 bits as show below.
Exponent is biased similar to conventional floating point.

Sign	Exponent	Mantissa
1 bit	5 bits.	4 bits.

- Numbers are rounded to the closest representable value. Any numbers that have 2 equidistant representations are rounded down towards zero.

19. What is the smallest non-zero positive value that can be represented? [1]

20. How do you represent the number 4.5 ? [1]

21. How do you represent -2^{-9} [1]

22. How many numbers can this 10 bit floating point represent in the range 1 ≤ f < 8). Hint: Write does the floating point expressions for 1 and 8 and the answer should be apparent. [1]

23. Sort the following numbers 0x300, 0x100, 0x104, 0x328, 0x12c smallest to largest [1]

F. Unsigned/Signed Numbers [5]

Suppose that we define a new number format, **NOT**.
`Negative numbers are represented by the inverse` $\sim$
`of the binary representations of their corresponding positive numbers.` Like 2's complement most significant bit of NOT denotes the number's sign (0 for positive, 1 for negative). Answer the following questions.

NOT (~)

Input	Output
1	0
0	1

24. Represent -133 as a 8-bit NOT number. [1]

25. Represent -124 as 8-bit NOT number. [1]

26. What is the range of numbers represented by 8 bit NOT8. [1]

27. What is the representation that requires fewest number of bits needed to cover the given range 0 to 10. [1]

28. What is the representation that requires fewest number of bits needed to cover the given range -1 to 4. [1]

G. Bitgames. Write down single line C expressions that calculate the following. [4]

You can use the following operators &,|,~, ||, &&, !=, ==, <<, >>, ^,

29. NegativeFloat(unsigned f) - Return bit-level equivalent of expression -f.

f is passed as unsigned int, but

- they are to be interpreted as the bit-level representations of
- single precision float values.

30. is_float_power_of_2(unsigned f, int e, int m). You can assume f is normalized.

f is passed as unsigned int, but it is to be interpreted as the bit-level representation of a floating point of size 1+e+m bits

- e number of bits in the exponent
- m number of bits in mantissa
- 1 sign bit

H. Assembler Linker Compiler [6]

```
.data
str: string "The sum of 1..100 is %d \n"

.text
main:
  add sp,sp -4
  sw ra, 0 (sp)
  mv a1, zero
  li x9, 100
  j check

loop:
  mul s2,x9,x9
  add a1,a1,s2
  add x9,x9,-1

chk:
  bnez x9, loop
  la a0, str
  jal printf
  mv a0,zero
  lw ra, 0(sp)
  addi sp,sp,4
  ret
```

31. Write down pseudo instructions and registers ? [2]

32. What is the symbol and relocation table ? [2]

33. Replace the labels of PC-relative targets with their immediate values. What is the offset value of bnez at address 0x20? Write your answer in decimal. [2]

G. Moderate. RISC-V

34. What is the value of s2 at the end of the program ? Write in hex (e.g., 0xFFFF) [3]

Initial values: s0 = 0xC, s1 = 0xA, a0 = 3, a1 = 0, a2 = 2

```
.text
  la a3, start
start:
        beq a1,a0,End
  lw  a4,20(a3)
  slli a5,a2,12
  add a4,a4,a5
  sw a4, 20(a3)
  add s2,s1,s0
  addi a1,a1,1
  j start

End:
    addi a0,zero,10
   ecall
```