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.

Questions Sheet.

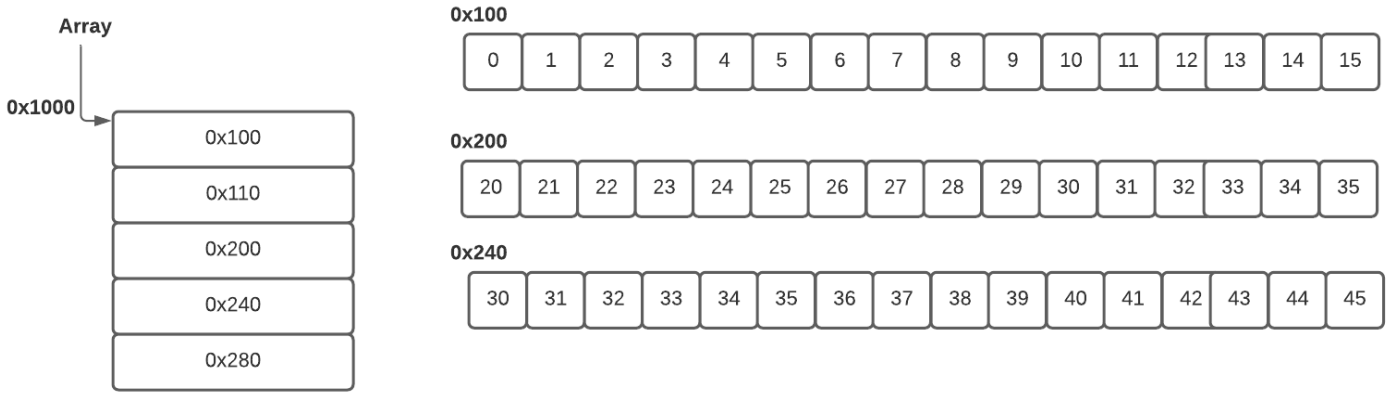
Read all of the following information before starting the exam:

- For each question fill out the appropriate choice or write text on 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.
- 1 pt Qs (0 or 1). 2 or 3pt Qs (if no explaination only 1 pt.)
- 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). 1 or 2 sentences atmost.
- 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. Arrays. 6 points

Q1-6

Given the multi-dimensional array of type int, fill in the table below. Assume pointers and ints are of size 4 bytes.
If value is unknown, write unknown.



{: .table-striped .table-bordered}

Access	Address	Value
1. Array[2][0]	0x200	20
2. Array[1][-1]	0x10C	3
3. Array[2][20]	0x250	34
4. Array[3]	0x100C	0x240
5. Array[4][-16]	0x240	30
6. Array[3][16]	0x280	Unknown

B. Hard. RISCV Blackbox. 6 Points

Assume we have two arrays input and output. Answer questions below

```
int input[6]  = {0x0, 0x5, 0x3, 0x4, 0x2, 0x1}
int result[6] = {0,0,0,0,0,0};
```

You can assume a0:input a1:result a2:8

```
main:
.....
  # a0=input, a1=output a2=6
  # Point 1
  jal ra, BLACKBOX
  # CHECK  finished calling BLACKBOX...
exit:
   ....

BLACKBOX:
  # Point 2. What registers are saved on stack?
  mv s0,a0    # s0=a0
  mv s1,a1    # s1=a1
  mv t0, zero # t0=0
loop:
  beq  t0, a2, done
  lw   t1, 0(s0)
  slli t2, t1,2
  add  t3,t2,a0
  lw  t1,0(t3)
  sw   t1,0(s1)
  addi s0,s0,4
  addi s1,s1,4
  addi a2,a2,-1
  j loop
done:
  # Point 3. What registers are restored from stack?
  jr ra
```

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

t0,t1,t2,t3 or t0-t3 and a0-a1. See ... at the end of main. It could be running any code including those that use t registers. Hence we have to save them. We have to save a registers for similar reason. Blackbox will only save t registers.

8. What is the minmum set of registers need to be stored onto the stack at this point: Point 2. ? [1]

s0,s1,ra. 12 bytes

9. What is the minmum set of registers need to be restored from the stack at this point: Point 3 ? [1]

s0,s1,ra. 12 bytes

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? [3]

result = {0,1,4,2,3,5}

C. RISC-V Instructions Encoding [5 points]

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

```
1  loop:
2    bgt t0, x0, end
3    lw  s0, 0(a0)
4    addi s0, s0, 1
5    sw  s0, 0(a0)
6    addi a0, a0, 4
7    addi t0, t0, -1
8    jal x0, loop
9  end:
10   addi a0,a0,10
11   ecall
```

11. For the instruction line 2: `bgt t0, x0, end` . What is the immediate [1]

28 or 0x1C

12. Line 2:What is actual opcode, rs1 and rs2 (not pseudo-names) ? [1]

`Hint: bgt with x0 is a pseudo-instruction. Convert to actual instruction before finding opcode`

0000000 - 0x5 or 00101 (rs2)
- 0x0 or 00000 (x0) rs1
- Opcode: 1100011 (0x63)

13. Line 2:What is funct7 and funct3 ? [1]

100 (f3) f7 (N/A)

Imm: 011100

14. What is the immediate field of line 8: `jal x0,loop` ? [1]

-24

15. What is the instruction corresponding to `0xFE9FF06F` ? [1]

jal x0, loop

D. Easy. RISC-V Custom Opcodes. 4 points

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 17 different operations: ADD, MUL, XOR, OR, NOT, SUB,

ACC, LD, SW, LUI, ADDI, MULI, XORI, SUBI, JAL, BEQ, and BLT. He decides that each instruction should be 17 bits wide.

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

- R-type: rs2,rs1,[rd=rs1],opcode
  - (rd = rs1 and hence can be excluded in the instruction e.g., add x6,x6,x5)
- I-type and Loads: imm,rs1,[rd=rs1],opcode
  - (rd = rs1 and hence can be excluded e.g., addi x6,x6,5)
- S-type: imm,rs2,rs1,opcode
- B-type: imm,[rs2=zero]rs1,opcode
  - (rs2 can be excluded since it is hardcode to zero.Only comparisons against the zero registers e.g., beq zero,x6,label)
- U-type: imm,rd,opcode
- UJ-type: imm,rd,opcode

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

5. Since 17 different operations

17. If the opcode bits were 5. what is the maximum number of registers. [1]

We only need to store 2 registers since rs1=rd always.
17 - 5 bits = 12 bits (2 registers). 6 bits per register. Maximum of 64 registers.

18. What is the smallest range of immediate that an I instruction can use ? Opcode bits is same as Q16. Assume that register width is same as Q17. [1]

17 - (5+6) = 6 bits. -32 - 31

19. What is the offset in terms of bytes for a jal instruction. Assume instruction start in 4 byte aligned offsets. Opcode bits is same as Q16. Assume that register width is same as Q17. [1]

17 - (6 + 5) = -32 - 31 offset instructions.
4 byte boundary. -128 - 124

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 12 bits as show below.
Exponent is biased similar to conventional floating point.

Sign	Exponent	Mantissa
1 bit	6 bits.	5 bits.

20. What is the bias for the exponent ? [1]

31

21. What is the smallest non-zero positive value that can be represented? (Normalized form) [1]

2^{-30}

22. How do you represent the number 3.5 ? [1]

0x418

2*1.75

0 100000 11000

0 Exp 0b100000 Mantissa 11000

23. How do you represent -2^{-25} [1]

0x8c0

1 000110 00000

24. How many numbers can this 12 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]

1 - 0x3e0
2 - 0x400 (32 numbers)
4 - 0x420 (32 numbers)
8 - 0x440 (32 numbers)

96 numbers.

F. Easy 2s complement [5]

25. Represent 0b10110100 as hexadecimal, unsigned decimal, and 2s complement decimal [1]

Hex: 0xB4
Unsigned: 180
2s Complement: -76

26. What is the number of bits needed to represent a 3 digit base-6 number ? [1]

8 bits. Max= 555_6 = 0 - 215 (216 numbers).

27. Lets use MSB (most-significant digit) for sign (1- postiive 0-ve) How many numbers can be represented ? [1]

36. (6*6)*2(1/2 -ve, 1/2 -ve).

28. What base 6 number XXX represents 0? (That is, your answer needs to have 3 base-6 characters.)? [1]

Hint

255_6

A 8-bit bias-encoded number presented in class has a bias of -127 so that roughly half the numbers are negative. but there's one more positive than negative number i.e., [-127 to +128]. Using an equivalent scheme for choosing the bias,

There is one more positive number
Hence answer is 255_6

000_6
100_6
200_6

300_6
400_6
500_6

G. Easy Lets C [7]

For this problem, assume all pointers and integers are 4 bytes and all characters are 1 byte. Consider the following C code (all the necessary #include directives are omitted). C structs are properly aligned in memory and all calls to malloc succeed.

```c
typedef struct entry {
  void *dat;
  struct entry *next;
  struct entry *prev;
} entry;
void add(entry *list, void *data) {
  entry *n = (entry *)malloc(sizeof(entry));
  n->data = data;
  n->next = list;
  n->prev = list->prev;
  list->prev->next = n;
  list->prev = n;
}
int main() {
  char *r = "CMPT 295";
  char s[] = "CMPT 295";
  entry node;
  node.next = &node;
  node.prv = &node;
  add(&node, r);
  add(&node, s);
  add(&node, &node);
  add(&node, calloc(sizeof(s) + 1, sizeof(char)));
}
```

For all of these questions, assume we are analyzing them right before main returns.

29. What type of address does node.next->next->data point to? [1] ?

- **Stack address** ✓
- Heap address
- Static address
- Code address

30. What type of address does &add point to? [1]

- Stack address
- Heap address
- Static address
- **Code address** ✓

31. What type of address does node.next->data point to? [1]

- Stack address
- Heap address
- **Static address** ✓
- Code address

32. What type of address does node.prev->prev->data points to? [1]

- **Stack address** ✓
- Heap address
- Static address
- Code address

32. What type of address does &node.prev->data points to? [1]

- Stack address
- **Heap address** ✓
- Static address
- Code address

34. How many bytes of memory are allocated but not free()d by this program, if any? [3]

57 bytes.

Each node will be sizeof(entry) == 12 bytes. We have allocated 4 nodes so 48 bytes.
We also made a calloc of 9 bytes (Since the compiler knows the length of the s array since it is stored on the stack which is 8 characters plus '\0` so 9 bytes long). This means that we have allocated: 57 bytes.

H. RISC-V Instruction II [6]

35. What does this sequence do. Explain ? [2]

int input[6] = {0,5,4,3,2,1}
a0=input

```
1 addi a1,zero,1
2 slli a1,a1,2
3 add  a2,a0,a1
4 sw   zero,0(a2)
```

Answer: input[1]=0
input = {0,0,4,3,2,1}

36. What does this sequence do. Explain ? [2]

int input[6] = {0,5,4,3,2,1}
a0=input

```
addi a1,zero,1
slli a1,a1,2
add  a2,a0,a1
lw   a3,0(a2)
slli a3,a3,2
add  a4,a0,a3
sw   zero,0(a4)
```

Answer: input[5]=0
input = {0,5,4,3,2,0}
Accesses array[array[1]]=0

37. What does this sequence do. Explain ? [2]

int input[6] = {0,5,4,3,2,1}
a0=input

```
addi a1,zero,2
slli a1,a1,2
add  a2,a0,a1
lw   a3,0(a2)
slli a3,a3,2
add  a4,a0,a3
addi a4,a4,4
sw   zero,0(a4)
```

Answer: input[5]=0
input = {0,5,4,3,2,0}
Indirect array access.
Accesses array[array[2]+1].