Computer Architecture and Organization

GATEBOOK

June 2018

1 Machine instructions and addressing modes

1. Which of the following is/are a matter of computer architecture rather than that of computer organization:
   (A) Main Memory type
   (B) Cache size
   (C) Instruction size
   (D) fan-in size of logic Gates
   (E) Addition of two integers

2. Which of the following about various modes of addressing mechanisms for operands in machine instructions is true:
   (A) It may increase the number of instructions
   (B) It helps to gain higher space density
   (C) It is time-efficient
   (D) All of the above
   (E) None of the above

3. A hypothetical machine has a machine-instruction as,
   Add R1, #2019
   Which of the following is best description of this instruction?
   (A) Adds the octal value 2019 to the content of R1 and stores the result in R1.
   (B) Adds 2019 to the content of R1 and brings the memory word from this location to R1.
   (C) Finds the memory location 2019 and adds that content to that of R1.
   (D) Adds the decimal value 2019 to the content of R1 and stores the result in R1.

4. The addressing mode which makes use of one in-direction pointers is:
   (A) Relative addressing mode
   (B) Index addressing mode
   (C) Doubly indirect addressing mode

(D) Direct addressing mode
(E) Indirect addressing mode

5. Consider an instruction for a hypothetical machine which supports all types of addressing modes, as shown below, with a blank space which might be filled to complete the instruction:
$ADD - \#2019$
Assuming the machine always seek for the smallest possible instruction, which of the following statements about the instruction above is true in order to make it complete:
(A) A register-name must be placed to complete the instruction
(B) The instruction is complete and adds zero to 2019 and store it in accumulator.
(C) The instruction is complete and the value 2019 is added to the value at top of the stack and the result is pushed onto the stack
(D) The instruction needs a memory location of another operand for the addition operation and further to store the result in the same location.

6. Consider following statements about autoincrement/autodecrement addressing modes:
(i) In autoincrement addressing mode address of the operand is computed first followed by increment of the content of the register
(ii) In autodecrement addressing mode address of the operand is computed first followed by decrement of the content of the register.
(iii) In autoincrement addressing mode address of the operand is computed after the increment of the content of the register
(iv) In autodecrement addressing mode address of the operand is computed after the decrement of the content of the register.
Which of the above statements is/are TRUE?
(A) (i) and (ii) only
(B) (i) and (iii) only
(C) (ii) only
(D) (iii) and (iv) only
(E) (i) and (iv) only

7. Which of the following is/are true of the auto-increment addressing mode?
(i) It is useful in creating self-relocating code
(ii) If it is included in an Instruction Set Architecture, then an additional ALU is required for effective address calculation
(iii) The amount of increment depends on the size of the data item accessed
(A) (i) only
(B) (ii) only
(C) (iii) only

(D) (ii) and (iii) only

8. Consider a hypothetical processor with an instruction:
LOAD R1, ((R2+K))
,which during execution reads a 32-bit word from memory and stores it in a 32-bit register R1. The effective address of the operand is obtained by loading the content of the memory address which is obtained by addition of a constant K and the contents of register R2. Which of the following best reflects the addressing mode implemented by this instruction for operand in memory?
(A) Absolute Addressing
(B) Register Scaled Addressing
(C) Register Indirect Scaled Addressing
(D) Base Indexed Indirect Addressing

9. Which of the following addressing modes is a best way to provide position-independence to the code:
(A) Based indexed addressing
(B) Relative addressing
(C) Absolute addressing
(D) Register Indirect addressing

10. In the absolute addressing mode:
(A) the operand is inside the instruction
(B) the address of the operand in inside the instruction
(C) the register containing the address of the operand is specified inside the instruction
(D) the location of the operand is implicit

11. Consider a 32-bit system with an instruction of size of four words. The first half of the instruction contains the OPCODE and second half contains an address-field. The instruction is stored in memory with base-address B. The address-field contains a value A. The operand is stored at memory location P. The addressing mode employed is Relative-addressing mode. What is the relation between P and B?
(A) P=(B+1)+M[A+3]
(B) P=(B+2)+M[A+3]
(C) P=(B+3)+M[A+3]
(D) P=(B+4)+M[A]

12. The memory locations 1000, 1001 and 1020 have data values 18, 1 and 16 respectively before the following program is executed.
MOVI R_s, 1 ; Move immediate

LOAD R_d, 1000(R_s) ; Load from memory
ADDI R_d, 1000 ; Add immediate
STOREI 0(R_d), 20 ; Store immediate
Which of the statements below is TRUE after the program is executed ?
(A) Memory location 1000 has value 20
(B) Memory location 1020 has value 20
(C) Memory location 1021 has value 20
(D) Memory location 1001 has value 20

13. Which of the following is/are true about the auto-increment addressing mode?
(I) It is useful in creating self-relocating code.
(II) If it is included in an Instruction Set Architecture, then an additional ALU is required for effective address calculation.
(III) The amount of increment depends on the size of the data item accessed.
(A) I only
(B) II only
(C) III Only
(D) II and III only

14. Which of the following must be true for the RFE (Return from Exception) instruction on a general purpose processor?
(I) It must be a trap instruction
(II) It should be a privileged instruction
(III) Exception may take place during execution of an RFE instruction
(A) I only
(B) II only
(C) I and II only
(D) I, II and III only

15. The most appropriate matching for the following pairs:

X: Indirect addressing	1: Loops
Y: Immediate addressing	2: Pointers
Z: Auto decrement addressing	3: Constants

is:
(A) X - 3, Y - 2, Z - 1
(B) X - 1, Y - 3, Z - 2
(C) X - 2, Y - 3, Z - 1
(D) X - 3, Y - 1, Z - 2

16. Which is the most appropriate match for the items in the first column with the items in the second column:

X. Indirect Addressing	I. Array implementation
Y. Indexed addressing	II. Writing relocatable code
Z. Base Register Addressing	III. Passing array as parameter

(A) (X, III) (Y, I) (Z, II)
(B) (X, II) (Y, III) (Z, I)
(C) (X, III) (Y, II) (Z, I)
(D) (X, I) (Y, III) (Z, II)

17. A CPU has 64-bit instructions. A program starts at address 0x10FEB2019. Which one of the following is a legal program counter (all values in hexadecimal)?
(A) 0x11011415B
(B) 0x11011415D
(C) 0x110114142
(D) 0x110114159

18. For computers based on three-address instruction formats, each address field can be used to specify which of the following:
S1: A memory operand
S2: A processor register
S3: An implied accumulator register
(A) S1 and S2
(B) S2 and S3
(C) Either S1 or S2 but not S3
(D) All of S1, S2 and S3

19. Consider following statements about relative addressing mode as follows:
(i) It enables increased instruction size
(ii) It enables easy relocation of data
(iii) It enables faster address calculations than absolute addressing
(iv) It allows indexing of array elements with same instruction
Which of the above statements is/are not FALSE?
(A) (i) only
(B) (i) and (iii) only
(C) (ii) and (iv) only
(D) (ii), (iii) and (iv) only

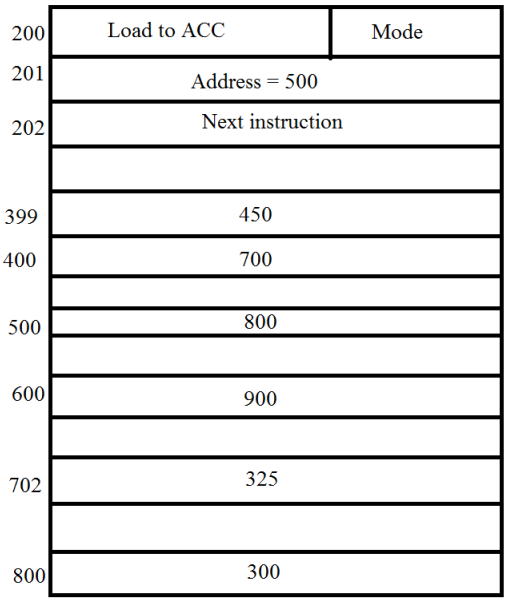
20. A processor has 70 distinct instructions and 46 general purpose registers. A 32-bit instruction word has an opcode, two register operands and an immediate operand. The number of bits available for the immediate operand field is Z. Compute the smallest positive value of the difference $2^p - Z$, where p is some non-negative integer.

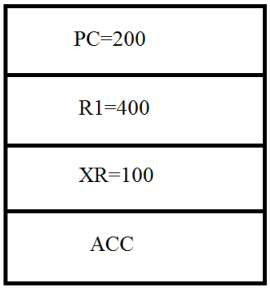
21. A processor that has carry, overflow and sign flag bits as part of its program status word (PSW) performs addition of the following two 2's complement numbers 01001101 and 11101001. After the execution of this addition operation, the status of the carry, overflow and sign flags, respectively will be:
(A) 1, 1, 0
(B) 1, 0, 0
(C) 0, 1, 0
(D) 1, 0, 1

22. A machine has a 32-bit architecture, with two-word long instructions. It has 4200 General Purpose Registers and 1048 Special Purpose Registers, each of which is 32 bits long. It needs to support 1880 instructions, which have an immediate operand in addition to two General Purpose Register operands and one Special Purpose Register operand. Assuming that the immediate operand is represented in 2's complement integer format, the maximum positive value of the immediate operand is:
(A) 65635
(B) 65536
(C) 32765
(D) 32767

23. In an 11-bit computer instruction format, the size of address field is 4-bits. The computer uses expanding opcode technique and has 5 two-address instructions and 32 one-address instructions. The number of zero-address instructions it can support is:
(A) 128
(B) 132
(C) 256
(D) 512

24. Consider the memory layout for some hypothetical system as follows:





Note that PC is program counter, R1 is processor register, XR is index register and AC is accumulator in the figure shown below:
Compute the effective addresses of operand for following addressing modes:
(i) Immediate addressing mode
(ii) Direct addressing mode
(iii) Indirect addressing mode
(iv) Relative addressing mode
(v) Indexed addressing mode
(vi) Register addressing mode
(vii) Register Indirect addressing mode
(viii) Autoincrement addressing mode
(ix) Autodecrement addressing mode

25. Consider a three word machine instruction, which is already in memory, given as:
$[< MUL >< R0 >< S = B >< D = A >]\ A,\ B$
The first operand (destination) "A" uses indexed-indirect addressing mode with R0 as the index register where A and R0 are added to get the intermediate address. The second operand (source) "B" uses double indirect addressing mode which uses 2 levels of indirections. A and B are memory addresses residing at the second and the third words, respectively. The first word of the instruction specifies the opcode, the index register designation and the source and destination addressing modes. During execution of MUL instruction, the two operands are multiplied and the result is stored in the destination (first operand). The number of memory cycles

needed during the execution cycle of the instruction is
(A) 3
(B) 4
(C) 5
(D) 6

26. Consider the MIPS code fragment shown below.

Line	Assembly Language Instruction	Machine Language Instruction(s)	Address
1	li $t6, 0xfffffec	0x3c01????	0x0040ffd8
		0x342e????	0x0040ffdc
2	move $t0, $a0	0x00044021	0x0040ffe0
3	move $t1, $a1	0x00054821	0x0040ffe4
4	li $t4, 0	0x340c0000	0x0040ffe8
5	add $t4, $t4, 1	0x218c????	0x0040ffec
6	lw $t3, 0($t0)	0x8d0b0000	0x0040fff0
7	slt $t5, $t3, $zero	0x0160682a	0x0040fff4
8	LblTwo: beq $t3, $zero, LblOne	0x1160????	0x0040fff8
9	add $11, $11, $12	0x????????	0x0040fffc
10	sw $t3, -4($t1)	0xad2b????	0x00410000
11	add $t1, $t1, 4	0x21290004	0x00410004
12	j LblTwo	0x????????	0x00410008
13	LblOne:		0x0041000c

Parts of several machine language instructions have been replaced by '?' sign. For each of those instructions provide the following information:
(i) Address (assume that the address of the first instruction is 0x0040ffd8)
(ii) Name of each field and its value in either decimal or hexadecimal representation
(iii) Addressing mode of each operand
(iv) Complete machine language instruction in hexadecimal representation
For example, for the first instruction:
(i) The address is 0x0040ffd8.
(ii) The fields are opcode = 15, rs = 0, rt = 1, and immediate = 0x0fff.
(iii) The addressing modes of 0,1, and 0x0fff are register, register, and immediate, respectively.
(iv) The complete machine language instruction is 0x3c010fff.

27. Consider a processor with byte-addressable memory. Assume that all registers, including Program Counter (PC) and Program Status Word (PSW), are of size 2 bytes. A stack in the main memory is implemented from memory location $(0100)_{16}$ and it grows upward. The stack pointer (SP) points to the top element of the stack. The current value of SP is $(016E)_{16}$. The *CALL* instruction is of two words, the first word is the opcode and the second word is the starting address of the subroutine (one word = 2 bytes).
The CALL instruction is implemented as follows:

• Store the current value of PC in the stack.
• Store the value of PSW register in the stack.
• Load the starting address of the subroutine in PC.
The content of PC just before the fetch of a CALL instruction is $(5FA0)_{16}$. After execution of the CALL instruction, the value of the stack pointer is:
(A) $(016A)_{16}$
(B) $(016C)_{16}$
(C) $(0170)_{16}$
(D) $(0172)_{16}$

28. Consider a processor with 150 registers and having 76 different types of instructions. Each instruction has five distinct fields, namely, opcode, two source register identifiers, one destination register identifier, and a thirteen-bit immediate value. Each instruction must be stored in memory in a byte-aligned fashion. If a program has 200 instructions, the amount of memory (in bytes) consumed by the program text is:
(A) 1000
(B) 1100
(C) 1200
(D) 1300

29. A processor has 16 integer registers (R0, R1, ... , R15) and 64 floating point registers (F0, F1, ... , F63). It uses a 2-byte instruction format. There are four categories of instructions: Type-1, Type-2, Type-3, and Type 4. Type-1 category consists of four instructions, each with 3 integer register operands (3Rs). Type-2 category consists of eight instructions, each with 2 floating point register operands (2Fs). Type-3 category consists of fourteen instructions, each with one integer register operand and one floating point register operand (1R+1F). Type-4 category consists of N instructions, each with a floating point register operand (1F). The maximum value of N is:
(A) 96
(B) 64
(C) 256
(D) 32

30. A certain processor supports only the immediate and the direct addressing modes. Which of the following programming language features cannot be implemented on this processor?
(i) Pointers
(ii) Arrays
(iii) Records
(iv) Recursive procedures with local variables
(A) (i) only

(B) (i) and (ii) only
(C) (iii) and (iv)
(D) All four

31. Consider the following assembly codes:
(P1) :
```
BYTE_VALUE DB 150          // A byte value is defined
WORD_VALUE DW 300          // A word value is defined
ADD BYTE_VALUE, 65         // An immediate operand 65 is added
MOV AX, 45H          // Immediate constant 45H is transferred to AX
```

(P2) :
```
MY_TABLE TIMES 10 DW 0          // Allocates 10 words (2 bytes) each
initialized to 0
MOV EBX, [MY_TABLE]          // Effective Address of MY_TABLE in
EBX
MOV [EBX], 110          // MY_TABLE[0] = 110
ADD EBX, 2          // EBX = EBX +2
MOV [EBX], 123          // MY_TABLE[1] = 123
```

Which of the following option is correct?
(A) P1 uses immediate Addressing, and P2 uses Indirect Memory Addressing mode.
(B) P1 uses immediate Addressing, and P2 uses Direct Memory Addressing mode.
(C) P1 uses Direct Memory Addressing, and P2 uses Direct Memory Addressing mode.
(D) None of these

32. If we use internal data forwarding to speed up the performance of a CPU (R1, R2 and R3 are registers and M[100] is a memory reference), then the sequence of operations:
$R_1 \rightarrow M[100]$
$M[100] \rightarrow R_2$
$M[100] \rightarrow R_3$

can be replaced by

(A)
$R_1 \rightarrow R_3$
$R_2 \rightarrow M[100]$

(B)
$M[100] \rightarrow R_2$
$R_1 \rightarrow R_2$

$R_1 \rightarrow R_3$

(C)
$R_1 \rightarrow M[100]$
$R_2 \rightarrow R_3$

(D)
$R_1 \rightarrow R_2$
$R_1 \rightarrow R_3$
$R_1 \rightarrow M[100]$

33. Assume that $EA = (X)+$ is the effective address equal to the contents of location X, with X incremented by one word length after the effective address is calculated; $EA = -(X)$ is the effective address equal to the contents of location X, with X decremented by one word length before the effective address is calculated; $EA = (X)-$ is the effective address equal to the contents of location X, with X decremented by one word length after the effective address is calculated. The format of the instruction is $(opcode, source, destination)$, which means $(destination \leftarrow source\ op\ destination)$. Using X as a stack pointer, which of the following instructions can pop the top two elements from the stack, perform the addition operation and push the result back to the stack.
(A) $ADD\ (X)-, (X)$
(B) $ADD\ (X), (X)-$
(C) $ADD\ -(X), (X)+$
(D) $ADD\ -(X), (X)+$

34. Consider the following assembly language program for a hypothetical processor. A, B, and C are 8 bit registers. The meanings of various instructions are shown as comments.

```
        MOV B, #8          ; B←8
        MOV C, #8          ; C←8
Z:
        CMP C, #0          ; compare C with 0
        JZ X               ; jump to X if zero flag is set
        SUB C, #1          ; C←C−1
        LRC A, #1          ; Left rotate A through carry by one bit.
                           ; Thus if the initial values of A and the carry
                           ; flag are a7..a0 and c0 respectively, their values
                           ; after the execution of this instruction will
                           ; be a6..a0c0 and a7 respectively.
        JC Y               ; jump to Y if carry flag is set
        JMP Z              ; jump to Z
Y:
        SUB B, #1          ; B ← B-1
```

```
        JMP Z           ; jump to Z
X:
```

If the initial value of register A is A_0 the value of register B after the program execution will be:
(A) the number of 0 bits in A_0
(B) the number of 1 bits in A_0
(C) A_0
(D) 8

35. In above question, which of the following instructions when inserted at location X will ensure that the value of the register A after program execution is as same as that at the end of the fourth iteration of the loop?
(A) RRC A, #4
(B) NOP
(C) LRC A, #4
(D) ADD A,#4

36. Consider following assembly language program for a hypothetical processor with 32 bit accumulator register.
Note that *xchg* is a swap operation and *arr* is an array.

```
arr:[1,2,3,4]
mov eax,arr
xchg eax,[arr+4]
xchg eax,[arr+8]
xchg eax,[arr+8]
xchg eax,[arr+12]
xchg arr,eax
```

What is the value of array *arr* after the completion of above code?
(A) 1,2,3,4
(B) 1,4,2,3
(C) 3,2,1,2
(D) 1,1,2,3
(E) 4,1,3,2

37. Consider following assembly language program for a hypothetical processor with 32 bit accumulator register.

```
        mov eax,1
        xor ebx,ebx
        xor edx,edx
L1:
```

```
        add eax,ebx         ; eax += ebx
        mov ebx,edx
        mov edx,eax
loop L1
        ret
```

What is the above assembly language program doing if the *ecx* is set to input value $n = (9)_{10}$?
(A) Computes addition of two numbers
(B) Outputs the Fibonacci series till 9
(C) Computes the factorial of 9
(D) Computes subtraction of two numbers
(E) Computes the ninth Fibonacci number.

38. Consider the following program segment for a hypothetical CPU having three user registers R_1, R_2 and R_3.

Instruction	Operation	Instruction Size (in words)
$MOV\ R_1,\ 5000$	$R_1 \leftarrow Memory[5000]$	2
$MOV\ R_2,\ (R_1)$	$R_2 \leftarrow Memory[(R_1)]$	2
$ADD\ R_2, R_3$	$R_2 \leftarrow R_2 + R_3$	1
$MOV\ 6000, R_2$	$Memory[6000] \leftarrow R_2$	2
$HALT$	$Machine\ halts$	1

Consider that the memory is byte addressable with size 32 bits, and the program has been loaded starting from memory location 1000 (decimal). If an interrupt occurs while the CPU has been halted after executing the HALT instruction, the return address (in decimal) saved in the stack will be:
(A) 1032
(B) 1020
(C) 1024
(D) 1028

39. In above question, let the clock cycles required for various operation be as follows:
Register to/from Memory transfer: 5 clock cycles.
Add with both operands in register: 1 clock cycle
Instruction fetch and decode: 3 clock cycles per word
The total number of clock cycles required to execute the above program is:
(A) 29
(B) 30
(C) 24
(D) 40

40. Consider following assembly language program for a hypothetical processor with 32 bit accumulator register.

```
main:
        mov ecx, 8
        mov eax, ecx
loop:
        cmp ecx, 1
        jle end
        sub ecx, 1
        mul ecx
        mul #0.5
        jmp loop
end:
```

What is the value of *eax* after the code is completed ?
(A) 315
(B) 252
(C) 5040
(D) 720
(E) 630

41. Consider the following program segment. Here R_1, R_2 and R_3 are the general purpose registers.

Instruction	**Operation**	**Instruction size (in words)**
$MOV\ R_1, (3000)$	$R_1 \leftarrow m[3000]$	2
LOOP:		
$MOV\ R_2, (R_3)$	$R_2 \leftarrow M[R_3]$	1
$ADD\ R_2, R_1$	$R_2 \leftarrow R_1 + R_2$	1
$MOV\ (R_3), R_2$	$M[R_3] \leftarrow R_2$	1
$INC\ R_3$	$R_3 \leftarrow R_3 + 1$	1
$DEC\ R_1$	$R_1 \leftarrow R_1 - 1$	1
$BNZ\ LOOP$	Branch on not zero	2
$HALT$	Stop	1

Assume that the content of memory location 3000 is 10 and the content of the register R_3 is 2000. The content of each of the memory locations from 2000 to 2010 is 100. The program is loaded from the memory location 1000. All the numbers are in decimal.
Assume that the memory is word addressable.
The number of memory references for accessing the data in executing the

program completely is:
(A) 10
(B) 11
(C) 20
(D) 21

42. In above question, assume that the memory is word addressable. After the execution of this program, the content of memory location 2010 is:
(A) 100
(B) 101
(C) 102
(D) 110

43. In above question, assume that the memory is byte addressable and the word size is 32 bits. If an interrupt occurs during the execution of the instruction "INC R3", what return address will be pushed on to the stack?
(A) 1005
(B) 1020
(C) 1024
(D) 1040

44. Consider the following assembly language program written for a hypothetical processor as follows:

Label	**Instruction**	**Comment**
	LXI H 2000H;	HL points at location 2000H
	MOV A, M[H];	Loads A with the content of M[H]
	MVI C, 08H;	Sets up counter for number of bits
	MVI B, 00H;	Sets counter to count number of ones
jump2:	RAL;	Rotates accumulator left through carry
	JNC jump1;	On no carry jumps to jump1:
	INC B;	Increases counter B by one
jump1:	DEC C;	Decreases counter C by one
	JNZ jump2;	When C is not zero jumps to jump2:
	HLT;	Terminates the program

What is the above program doing?
(A) Computes the decimal equivalent of a number stored in memory location 2000H.
(B) Multiplies the value stored in memory location 2000H to the content of accumulator and stores the result back to location 2000H.
(C) Counts number of 0's in a byte stored in memory location 2000H.

(D) Counts number of 1's in a byte stored in memory location 2000H.

45. Consider the following assembly language program,to set array-cells ,written for a hypothetical processor with 16 bit registers, as follows:

Instruction	**Comment**
MY_TABLE TIMES 10 DW 0	Allocates 10 words (2 bytes) each initialized to 0
MOV EBX, [MY_TABLE]	Loads Effective Address of MY_TABLE in EBX
MOV B, EBX	
L1: MOV [EBX], C	
ADD EBX, 2	
ADD C, 10	
CMP EBX, B+D	
BNZ L1	

Which of the following pairs could be a valid combination of the values of the tuple (C,D) such that all cells in array are set?
(A) (10,40)
(B) (20,10)
(C) (10,20)
(D) (20,40)

46. Consider the following one address instructions for a hypothetical processor implementing a function denoted by f:
Move x
Mult x
Store z
Move x
Mult 2
Add z
Add 3
Store y

Which of the following values represents the value of the function for input value = $(17)_{10}$:
(A) 230
(B) 232
(C) 334
(D) 326

47. Let $A = [a_1, a_2, ..., a_{100}]$ and $X = [x_1, x_2, ..., x_{100}]$ represent two arrays of integers which are stored in the main memory at location 1000_{10} and

2000_{10} respectively. The following program processes these arrays and produces an output as *output*. Direct addressing mode is employed to compute the effective address of the operand under consideration.

```
Move output, 0
Load Temp1, 1000
Load Temp2, 2000
Mult Temp1, Temp2
Add output, Temp1
...
Load Temp1, 1099
Load Temp2, 2099
Mult Temp1, Temp2
Add output, Temp1
```

What is the functionality of this code?
(A) It computes $\sum_{i=1}^{100} {a_i}^{x_i} + 1$
(B) It computes $\sum_{i=1}^{100} (a_i + x_i)$
(C) It computes $\sum_{i=1}^{100} (a_i * x_i)$
(D) It computes $\sum_{i=1}^{100} ({a_i}^2 + {x_i}^2) + 1$