GATE 2024 CSE Solution SET 1

Ques.1 If '→' denotes increasing order of intensity, then the meaning of the words [dry → arid → parched] is analogous to [diet → fast → ________]. Which one of the given options is appropriate to fill the blank?

(A) starve
(B) reject
(C) feast
(D) deny

Solution. (A) Strave, The next step up in intensity from fasting would be the most extreme form of food restriction, which is starvation.

Ques.3 Consider the following sample of numbers: 9, 18, 11, 14, 15, 17, 10, 69, 11, 13 The median of the sample is
(A) 13.5
(B) 14
(C) 11
(D) 18.7

Solution. (A) 13.5, Find the median of a sample of numbers, we first need to arrange the numbers in ascending order and then identify the middle value. For a set of 10 numbers, the median will be the average of the 5th and 6th numbers.

Here are the steps:

1. Arrange the numbers in ascending order:
   9,10,11,11,13,14,15,17,18,69

2. Since there are 10 numbers, the median will be the average of the 5th and 6th numbers in this ordered list.
3. The 5th number is 13 and the 6th number is 14.
4. Calculate the average of these two numbers: Median=13+142=272=13.5

Ques.4 The number of coins of ₹1, ₹5, and ₹10 denominations that a person has are in the ratio 5:3:13. Of the total amount, the percentage of money in ₹5 coins is

(A) 21%

(B) 14 2 7 %

(C) 10%

(D) 30%

Solution. (C) 10%, To solve this problem, let's denote the number of coins of ₹1, ₹5, and ₹10 denominations as 5x,3x, and 13x respectively, where x is a common multiplier.

Given:

- Ratio of coins: 5x:3x:13x
- Total ratio parts: 5x+3x+13x=21x

Let's assume the values for simplicity:

- Number of ₹1 coins: 5x
- Number of ₹5 coins: 3x
- Number of ₹10 coins: 13x

Now, calculate the total amount of money:

Total amount=1×5x+5×3x+10×13x

Total amount=5x+15x+130x=150x

Next, calculate the amount of money in ₹5 coins:

Money in ₹5 coins$=5\times3x=15x$

Now, find the percentage of money in ₹5 coins relative to the total amount:

Percentage$=(15x/150x)\times100\%=15/150\times100\%=10\%$

Therefore, the percentage of money in ₹5 coins is (C) 10%.

Q.6 In the given text, the blanks are numbered (i)−(iv). Select the best match for all the blanks. Steve was advised to keep his head (i) before heading (ii) to bat; for, while he had a head (iii) batting, he could only do so with a cool head s (iv) his shoulders.

(A) (i) down (ii) down (iii) on (iv) for

(B) (i) on (ii) down (iii) for (iv) on

(C) (i) down (ii) out (iii) for (iv) on

(D) (i) on (ii) out (iii) on (iv) for

Solution:- (C) (i) down (ii) out (iii) for (iv) on

"Steve was advised to keep his head (i) ______ before heading (ii) ______ to bat; for, while he had a head (iii) ______ batting, he could only do so with a cool head (iv) ______ his shoulders."

- (i) "Keep his head" suggests staying calm and focused. The phrase "keep his head down" is common in this context, meaning to remain inconspicuous or to focus.
- (ii) "heading to bat" would make more sense with "out" as it refers to going out to the field to bat.
- (iii) "had a head for batting" means he was good at it, a common phrase.
- (iv) "With a cool head on his shoulders" is an idiom meaning to stay calm under pressure.

Now, let's match the options:

(A) (i) down (ii) down (iii) on (iv) for

- This does not make sense as "heading down to bat" is not appropriate.

(B) (i) on (ii) down (iii) for (iv) on

- "keep his head on" is not a common phrase, and "heading down to bat" is incorrect.

(C) (i) down (ii) out (iii) for (iv) on

- "keep his head down" and "heading out to bat" are correct. "had a head for batting" and "with a cool head on his shoulders" also fit perfectly.

(D) (i) on (ii) out (iii) on (iv) for

- "keep his head on" is not correct.

Q.12 The product of all eigenvalues of the matrix

1 2 3

4 5 6

7 8 9

is

(A) −1

(B) 0

(C) 1

(D) 2

Solution. (B) 0, To find the product of all eigenvalues of the matrix

$$A = \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}$$

we first calculate the determinant of A, which is used to find the product of eigenvalues because det (A)=λ1λ2λ3, where λ1,λ2,λ3 are the eigenvalues of A.

Calculating the determinant:

$$\det(A)= \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}$$

Expand along the first row:

$$\det(A)= 1.\begin{vmatrix} 5 & 6 \\ 8 & 9 \end{vmatrix} -2.\begin{vmatrix} 4 & 6 \\ 7 & 9 \end{vmatrix} +3.\begin{vmatrix} 4 & 5 \\ 7 & 8 \end{vmatrix}$$

Calculate each 2x2 determinant:

$$\begin{vmatrix} 5 & 6 \\ 8 & 9 \end{vmatrix} =5.9-6\cdot 8=45-48 = -3$$

$$\begin{vmatrix} 4 & 5 \\ 7 & 8 \end{vmatrix} =4\cdot 9-6\cdot 7=36-42 = -6$$

4 7	5 8	$=4\cdot8-5\cdot7=32-35=-3$

Now substitute back into the determinant expression:

$\det(A)=1\cdot(-3)-2\cdot(-6)+3\cdot(-3)=-3+12-9=0$

Therefore, det(A)=0\det(A) = 0det(A)=0, indicating that the product of all eigenvalues λ1λ2λ3 is 0.

Q.13 Consider a system that uses 5 bits for representing signed integers in 2's complement format. In this system, two integers A and B are represented as A=01010 and B=11010. Which one of the following operations will result in either an arithmetic overflow or an arithmetic underflow?

(A) A **+** B

(B) A **−** B

(C) B **−** A

(D) 2 $*$ **B**

Solution. (B) A - B, To determine which operation results in either an arithmetic overflow or underflow, we need to first convert the 5-bit signed integers A and B from their 2's complement representation to decimal values.

Step 1: Convert A and B to Decimal

- A=01010
- B=11010

For 5-bit 2's complement numbers:

- If the first bit is 0, the number is positive.
- If the first bit is 1, the number is negative.

Conversion of A=01010:

- The first bit is 0, so A is positive.
- Convert directly to decimal: $A=0\cdot2^4+1\cdot2^3+0\cdot2^2+1\cdot2^1+0\cdot2^0=8+2=10$.

Conversion of B=11010:

- The first bit is 1, so B is negative.
- To find the decimal value, first find the 2's complement (invert bits and add 1):
    - Invert bits: $B'=00101$
    - Add 1: $B''=00101+1=00110$
    - Convert 00110 to decimal: $0\cdot2^4+0\cdot2^3+1\cdot2^2+1\cdot2^1+0\cdot2^0=4+2=6$
    - Since B was negative: $B=-6$

Step 2: Perform the Operations

(A) A+B:

- $A+B=10+(-6)=4$.
- No overflow or underflow here.

(B) A−B:

- $A-B=10-(-6)=10+6=16$.
- In 5-bit 2's complement, the maximum positive value is 01111 (15). Here, 16 is out of range and causes overflow.

(C) B−A:

- $B-A=-6-10=-16$
- In 5-bit 2's complement, the minimum negative value is 1000 (-16). So, no overflow or underflow here.

(D) $2\cdot B$

- $2\cdot B=2\cdot(-6)=-12$

- In 5-bit 2's complement, the value -12 can be represented correctly without overflow or underflow.

Q.14 Consider a permutation sampled uniformly at random from the set of all permutations of {1, 2, 3, $\cdots$, n} for some n ≥ 4. Let X be the event that 1 occurs before 2 in the permutation, and Y the event that 3 occurs before 4. Which one of the following statements is TRUE?

(A) The events X and Y are mutually exclusive

(B) The events X and Y are independent

(C) Either event X or Y must occur

(D) Event X is more likely than Event Y

Solution. (B) The events X and Y are independent, To determine which statement is true regarding the events X and Y, we need to analyze the properties of these events in the context of a random permutation of {1,2,3,…,n}.

Definitions

- Let X be the event that 1 occurs before 2 in the permutation.
- Let Y be the event that 3 occurs before 4 in the permutation.

Analysis

1. **Event X**:
   - In any permutation of {1,2,3,…,n}, there are two possible orderings for the pair (1,2)(1, 2)(1,2): either 1 comes before 2, or 2 comes before 1.
   - Each of these orderings is equally likely, so the probability that 1 comes before 2 is P(X)=$1/2$
2. **Event Y**:
   - Similarly, for the pair (3,4), either 3 comes before 4, or 4 comes before 3.

    - Each of these orderings is equally likely, so the probability that 3 comes before 4 is P(Y)=$1/2$

Evaluating Statements

(A) **The events X and Y are mutually exclusive**:

- Events are mutually exclusive if they cannot both occur at the same time.
- Here, X and Y can both occur in the same permutation (e.g., the permutation [1,3,2,4,…]).
- Therefore, X and Y are not mutually exclusive. This statement is false.

(B) **The events X and Y are independent**:

- Two events are independent if the occurrence of one does not affect the probability of the occurrence of the other.
- To check independence, we need to verify if P(X∩Y)=P(X)⋅P(Y).
- Since P(X)=$1/2$P(X) and P(Y)=$1/2, P(X). P(Y)$
- The probability P(X∩Y) (both 1 before 2 and 3 before 4) is indeed $1/4$ because there are four equally likely scenarios for the order of (1, 2) and (3, 4):
    1. 1,2,3,4
    2. 1,2,4,3
    3. 2,1,3,4
    4. 2,1,4,3
- Thus, P(X∩Y)=$1/4\ p$, confirming that X and Y are independent. This statement is true.

(C) **Either event X or Y must occur**:

- This would mean X∪Y is the entire sample space.
- It's possible for neither X nor Y to occur (e.g., the permutation [2,1,4,3,.....]), so this statement is false.

(D) **Event X is more likely than event Y**:

- We already established that P(X)=$1/2$ and P(Y)= $1/2$
- Therefore, X is not more likely than Y. This statement is false.

Q.15 Which one of the following statements is FALSE?

(A) In the cycle stealing mode of DMA, one word of data is transferred between an I/O device and main memory in a stolen cycle

(B) For bulk data transfer, the burst mode of DMA has a higher throughput than the cycle stealing mode

(C) Programmed I/O mechanism has a better CPU utilization than the interrupt driven I/O mechanism

(D) The CPU can start executing an interrupt service routine faster with vectored interrupts than with non-vectored interrupts

Solution. (C) Programmed I/O mechanism has a better CPU utilization than the interrupt driven I/O mechanism, Let's evaluate each statement to determine which one is false.

(A) In the cycle stealing mode of DMA, one word of data is transferred between an I/O device and main memory in a stolen cycle:

- This is true. In cycle stealing mode, the DMA controller takes control of the system bus for one bus cycle to transfer one word of data, effectively "stealing" a cycle from the CPU.

(B) For bulk data transfer, the burst mode of DMA has a higher throughput than the cycle stealing mode:

- This is true. In burst mode, the DMA controller transfers a block of data all at once, without releasing the bus in between, which provides higher throughput compared to cycle stealing mode where the bus is released after each word is transferred.

(C) Programmed I/O mechanism has a better CPU utilization than the interrupt driven I/O mechanism:

- This is false. In programmed I/O, the CPU is involved in the data transfer process, continuously polling the I/O device, which leads to poor CPU utilization. In interrupt-driven I/O, the CPU can perform other tasks and only deals with the I/O device when interrupted, leading to better CPU utilization.

(D) The CPU can start executing an interrupt service routine faster with vectored interrupts than with non-vectored interrupts:

- This is true. With vectored interrupts, the interrupting device sends a unique code (vector) to the CPU, which can directly jump to the specific interrupt service routine, making the process faster compared to non-vectored interrupts where the CPU has to poll each device to identify the source of the interrupt.

Q.16 A user starts browsing a webpage hosted at a remote server. The browser opens a single TCP connection to fetch the entire webpage from the server. The webpage consists of a top-level index page with multiple embedded image objects. Assume that all caches (e.g., DNS cache, browser cache) are all initially empty. The following packets leave the user's computer in some order.

(i) HTTP GET request for the index page

(ii) DNS request to resolve the web server's name to its IP address

(iii) HTTP GET request for an image object

(iv) TCP SYN to open a connection to the web server

Which one of the following is the CORRECT chronological order (earliest in time to latest) of the packets leaving the computer ?

(A) (iv), (ii), (iii), (i)

(B) (ii), (iv), (iii), (i)

(C) (ii), (iv), (i), (iii)

(D) (iv), (ii), (i), (iii)

Solution. (C) (ii), (iv), (i), (iii), To determine the correct chronological order of the packets leaving the computer when a user starts browsing a webpage, let's break down the process step-by-step.

1. DNS Resolution:
   - Before the browser can connect to the web server, it needs to resolve the server's domain name to an IP address.
   - Packet (ii) DNS request to resolve the web server’s name to its IP address is the first step.
2. Establishing the TCP Connection:
   - Once the DNS resolution is complete and the IP address is obtained, the browser can initiate a TCP connection to the web server.
   - Packet (iv) TCP SYN to open a connection to the web server is the next step.
3. Fetching the Index Page:
   - After the TCP connection is established, the browser sends an HTTP GET request for the index page.
   - Packet (i) HTTP GET request for the index page follows the TCP connection establishment.
4. Fetching Embedded Image Objects:
   - Once the index page is fetched, the browser parses it and identifies the embedded image objects.
   - The browser then sends HTTP GET requests for each image object.
   - Packet (iii) HTTP GET request for an image object is the final step.

Given this sequence, the correct chronological order (earliest to latest) of the packets leaving the computer is:

(C) (ii), (iv), (i), (iii)

Q.17 Given an integer array of size N, we want to check if the array is sorted (in either ascending or descending order). An algorithm solves this problem by making a single pass through the array and comparing each element of the array only with its adjacent elements. The worst-case time complexity of this algorithm is

(A) both O(N) and Ω(N)

(B) O(N) but not Ω(N)

(C) Ω(N) but not O(N)

(D) neither O(N) nor Ω(N)

Solution:- (A) both O(N) and Ω(N), To determine the worst-case time complexity of the algorithm that checks if an integer array of size N is sorted (in either ascending or descending order) by making a single pass through the array and comparing each element only with its adjacent elements, let's analyze the algorithm step-by-step.

Algorithm Analysis

1. Single Pass Through the Array:
    - The algorithm needs to check every pair of adjacent elements to determine if the array is sorted.
    - This involves making N−1 comparisons (for an array of size N).

Time Complexity

- O(N) (Big-O Notation):
    - This notation represents the upper bound of the algorithm's running time. It means that in the worst-case scenario, the algorithm will take at most c⋅N time for some constant c.
    - Since the algorithm makes N−1 comparisons, which is proportional to N, the time complexity is O(N).
- Ω(N) (Omega Notation):

- This notation represents the lower bound of the algorithm's running time. It means that in the best-case scenario, the algorithm will take at least $c \cdot N$ time for some constant c.
- Even if the array is already sorted in ascending or descending order, the algorithm still needs to make N−1 comparisons to verify this. Hence, the time complexity is $\Omega(N)$.

Since the algorithm's time complexity is both $O(N)$ and $\Omega(N)$, we can conclude that the worst-case time complexity of this algorithm is:

(A) both $O(N)$ and $\Omega(N)$

Q.18 Consider the following C program:

```
 #include int main()

{

 int a = 6;

 int b = 0;

 while(a < 10)

 { a = a / 12 + 1;

 a += b;}

 printf(”%d”, a); return 0;}
```

Which one of the following statements is CORRECT?

(A) The program prints 9 as output

(B) The program prints 10 as output

(C) The program gets stuck in an infinite loop

(D) The program prints 6 as output

Solution. (C) The program gets stuck in an infinite loop, Let's analyze the given C program step-by-step to determine its behavior and output:

```
#include <stdio.h>

int main(){

    int a = 6;

    int b = 0;

    while(a < 10) {

        a = a / 12 + 1;

        a += b;

    }

    printf("%d", a);

    return 0;

}
```

Step-by-Step Analysis

1. Initialization:
    - `int a = 6;`
    - `int b = 0;`
2. While Loop:
    - The condition for the loop is `while(a < 10)`. Initially, a is 6, so the loop condition is true, and the loop will execute.
3. First Iteration:
    - Inside the loop:

        - `a = a / 12 + 1;`
            - a is 6.
            - `a / 12` is `0` because it is an integer division.
            - So, `a = 0 + 1 = 1`.
        - `a += b;`
            - b is 0.
            - So, `a = 1 + 0 = 1`.

4. Second Iteration:
    - The loop condition is checked again: `while(a < 10)`. a is now 1, so the condition is true, and the loop will execute again.
    - Inside the loop:
        - `a = a / 12 + 1;`
            - a is 1.
            - `a / 12` is `0` because it is an integer division.
            - So, `a = 0 + 1 = 1`.
        - `a += b;`
            - b is 0.
            - So, `a = 1 + 0 = 1`.

Since a is not changing within the loop and remains 1, the loop will run indefinitely because a will always be less than 10.

Conclusion

The correct statement regarding this program is:

(C) The program gets stuck in an infinite loop.

Q.19 Consider the following C program:

#include void

fX();

```
 int main(){

 fX();

 return 0;}

 void fX(){

char a; if((a=getchar()) != ’\n’)

 fX(); if(a != ’\n’) putchar(a);}
```

Assume that the input to the program from the command line is 1234 followed by a newline character. Which one of the following statements is CORRECT?

(A) The program will not terminate

(B) The program will terminate with no output

(C) The program will terminate with 4321 as output

(D) The program will terminate with 1234 as output

Solution.(C) The program will terminate with 4321 as output, To understand the behavior and output of the given C program, let's analyze it step-by-step.

Program Analysis

c

Copy code

```
#include <stdio.h>


void fX();
```

```
int main(){
    fX();
    return 0;
}

void fX(){
    char a;
    if((a = getchar()) != '\n')
        fX();
    if(a != '\n')
        putchar(a);
}
```

1. Main Function:
   - The `main` function calls `fX()` and then returns.
2. fX Function:
   - The function reads a character from the input and assigns it to a.
   - If a is not a newline character (`'\n'`), the function calls itself recursively.
   - After the recursive call returns, if a is not a newline character, it prints the character.

Execution with Input "1234\n"

1. First Call to `fX()`:
   - `a = getchar()` reads '1'.
   - Since '1' is not `'\n'`, it calls `fX()` recursively.
2. Second Call to `fX()`:
   - `a = getchar()` reads '2'.
   - Since '2' is not `'\n'`, it calls `fX()` recursively.
3. Third Call to `fX()`:
   - `a = getchar()` reads '3'.
   - Since '3' is not `'\n'`, it calls `fX()` recursively.
4. Fourth Call to `fX()`:
   - `a = getchar()` reads '4'.
   - Since '4' is not `'\n'`, it calls `fX()` recursively.
5. Fifth Call to `fX()`:
   - `a = getchar()` reads `'\n'`.
   - Since a is `'\n'`, it does not call `fX()` recursively.
   - The second `if` statement (`if(a != '\n')`) is also false, so it does not print anything and returns.

Now, as the recursive calls return back up the stack:

6. Fourth Call Returns:
   - a is '4'.
   - Since '4' is not `'\n'`, it prints '4'.
7. Third Call Returns:
   - a is '3'.
   - Since '3' is not `'\n'`, it prints '3'.
8. Second Call Returns:
   - a is '2'.
   - Since '2' is not `'\n'`, it prints '2'.
9. First Call Returns:

    - a is '1'.
    - Since '1' is not ' \n ', it prints '1'.

Conclusion

The program reads the input "1234\n" and prints the characters in reverse order. Therefore, the correct statement is:

(C) The program will terminate with 4321 as output.

Q.20 Let S be the specification: "Instructors teach courses. Students register for courses. Courses are allocated classrooms. Instructors guide students." Which one of the following ER diagrams CORRECTLY represents S?

Q.21 In a B+ tree, the requirement of at least half-full (50%) node occupancy is relaxed for which one of the following cases?

(A) Only the root node

(B) All leaf nodes

(C) All internal nodes

(D) Only the leftmost leaf node

Solution. **(A) Only the root node**,In a B+ tree, the requirement of at least half-full (50%) node occupancy is relaxed for the root node. This relaxation is necessary because, during deletions, the root node might end up with fewer keys than half of its capacity, especially when the tree shrinks in height.

Explanation:

- Root Node: The root node is allowed to have fewer than half of its maximum capacity of keys or even be empty, especially when the tree has very few elements. This is to ensure that the tree can still function correctly and adjust in height as elements are added or removed.
- Internal Nodes: All internal nodes, except the root, must maintain at least half-full occupancy to ensure the tree remains balanced and efficient in terms of search, insert, and delete operations.
- Leaf Nodes: Similarly, all leaf nodes must also maintain at least half-full occupancy to maintain the tree's balance and efficiency.
- Leftmost Leaf Node: There is no special rule that relaxes the occupancy requirement specifically for the leftmost leaf node.

Conclusion:

The correct answer is: (A) Only the root node

Q.22 Which of the following statements about a relation R in first normal form (1NF) is/are TRUE ?

(A) R can have a multi-attribute key

(B) R cannot have a foreign key

(C) R cannot have a composite attribute

(D) R cannot have more than one candidate key

Solution. **(A) R can have a multi-attribute key,(C) R cannot have a composite attribute**

Let's evaluate each statement regarding a relation RRR in first normal form (1NF) to determine which ones are true.

Understanding 1NF

A relation is in 1NF if it satisfies the following conditions:

1. All attributes contain only atomic (indivisible) values.

2. Each attribute contains only a single value from its domain for each tuple (row).
3. The order in which data is stored does not matter.

Evaluation of Statements

(A) R can have a multi-attribute key:

- True. A relation in 1NF can have a multi-attribute key (also known as a composite key), which is a key composed of more than one attribute.

(B) R cannot have a foreign key:

- False. There is no restriction in 1NF that prevents a relation from having foreign keys. Foreign keys are used to enforce referential integrity and can be present in any normal form, including 1NF.

(C) R cannot have a composite attribute:

- True. 1NF requires that all attributes contain only atomic values. Composite attributes, which are attributes that can be divided into smaller subparts (e.g., a full name attribute that can be divided into first name and last name), violate this requirement.

(D) R cannot have more than one candidate key:

- False. A relation in 1NF can have multiple candidate keys. Candidate keys are sets of attributes that can uniquely identify tuples in a relation. There is no restriction in 1NF on the number of candidate keys.

Conclusion

The statements that are true about a relation RRR in 1NF are: (A) R can have a multi-attribute key (C) R cannot have a composite attribute

Q.23 Let $L1$, $L2$ be two regular languages and $L3$ a language which is not regular. Which of the following statements is/are always TRUE?

(A) $L1 = L2$ if and only if $L1 \cap \overline{L2} = \phi$

(B) $L1 \cup L3$ is not regular

(C) $\overline{L3}$ is not regular

(D) $\overline{L1} \cup \overline{L2}$ is regular

Q.24 Which of the following statements about threads is/are TRUE?

(A) Threads can only be implemented in kernel space

(B) Each thread has its own file descriptor table for open files

(C) All the threads belonging to a process share a common stack

(D) Threads belonging to a process are by default not protected from each other

Solution. **(D) Threads belonging to a process are by default not protected from each other**

Let's analyze each statement about threads to determine if they are true.

Statement (A): Threads can only be implemented in kernel space

- False. Threads can be implemented in both user space and kernel space.
  - User space threads (or user-level threads) are managed by a user-level library, without kernel support.

    - Kernel space threads (or kernel-level threads) are managed directly by the operating system kernel.

Statement (B): Each thread has its own file descriptor table for open files

- False. All threads within the same process share the same file descriptor table.
    - This is because file descriptors are associated with the process, not with individual threads. Threads within the same process can access and manipulate the same set of open files.

Statement (C): All the threads belonging to a process share a common stack

- False. Each thread has its own stack.
    - The stack is used to store local variables, return addresses, and control information for function calls. Each thread needs its own stack to manage its execution context independently.

Statement (D): Threads belonging to a process are by default not protected from each other

- True. Threads within the same process share the same address space and can directly access each other's memory.
    - This means threads are not protected from each other, and a thread can read or write another thread's stack or data, leading to potential data corruption if not managed carefully.

Conclusion

The statements that are true about threads are:

(D) Threads belonging to a process are by default not protected from each other.

Q.25 Which of the following process state transitions is/are NOT possible?

(A) Running to Ready

(B) Waiting to Running

(C) Ready to Waiting

(D) Running to Terminated

Solution.**(B) Waiting to Running ,(C) Ready to Waiting,** Let's analyze each of the process state transitions to determine which are not possible.

Process States

- Running: The process is currently being executed by the CPU.
- Ready: The process is ready to be executed but is not currently being executed.
- Waiting: The process is waiting for some event (like I/O completion) to occur.
- Terminated: The process has finished execution.

State Transitions

(A) Running to Ready

- Possible: This transition occurs when the CPU scheduler decides to preempt the currently running process and move it back to the ready queue, typically due to a time slice expiring.

(B) Waiting to Running

- Not Possible: A process in the waiting state must first transition to the ready state when the event it is waiting for occurs. It cannot directly transition to the running state without first being in the ready state.

(C) Ready to Waiting

- Not Possible: A process in the ready state cannot directly transition to the waiting state. It must first be in the running state before it can move to the waiting state. This transition occurs when the process is running and then needs to wait for an I/O operation or another event.

(D) Running to Terminated

- Possible: This transition occurs when a process has completed its execution or is terminated by the operating system.

Conclusion

The transitions that are NOT possible are:

(B) Waiting to Running (C) Ready to Waiting

Q.26 Which of the following is/are Bottom-Up Parser(s)?

(A) Shift-reduce Parser

(B) Predictive Parser

(C) LL(1) Parser

(D) LR Parser

Solution. **(A) Shift-reduce Parser, (D) LR Parser,** Let's clarify the nature of each type of parser to identify which one(s) are bottom-up parsers:

Types of Parsers:

1. Shift-Reduce Parser:
    - Bottom-Up: Yes, shift-reduce parsers are bottom-up parsers. They work by shifting input onto a stack until the right-hand side of a production rule appears on the top of the stack, which then triggers a reduction.
2. Predictive Parser:

    - Top-Down: Predictive parsers are top-down parsers. They predict which production rule to apply based on the current input symbol and the look-ahead symbol.
3. LL(1) Parser:
    - Top-Down: LL(1) parsers are specific types of predictive parsers. They are table-driven, recursive-descent parsers that use a lookahead of one symbol to predict which production rule to apply.
4. LR Parser:
    - Bottom-Up: Yes, LR parsers are bottom-up parsers. They use a bottom-up approach to parse the input by recognizing strings of terminals that match the right-hand side of a production rule, then reducing them to the non-terminal on the left-hand side.

Conclusion:

The bottom-up parser(s) among the options provided are:

- (A) Shift-reduce Parser
- (D) LR Parser

Therefore, the correct answer is (A) and (D).

Q.27 Let A and B be two events in a probability space with $P(A) = 0.3$, $P(B) = 0.5$, and $P(A \cap B) = 0.1$. Which of the following statements is/are TRUE?

(A) The two events A and B are independent

(B) $P(A \cup B) = 0.7$

(C) $P(A \cap B^c) = 0.2$, where B^c is the complement of the event B

(D) $P(A^c \cap B^c) = 0.4$, where A^c and B^c are the complements of the events A and B, respectively

Solution. **(B)** $P(A \cup B)$ **= 0.7,(C)** $P(A \cap B\ c\ )$ **= 0.2, where** $B\ c$ **is the complement of the event** B**,** Let's evaluate each statement given the probabilities provided for events A and B.

Given Probabilities:

- $P(A)=0.3$
- $P(B)=0.5$
- $P(A \cap B)=0.1$

Statement (A): The two events A and B are independent

Events A and B are independent if $P(A \cap B) = P(A) \cdot P(B)$.

- $P(A) \cdot P(B)=0.3 \cdot 0.5=0.15$
- $P(A \cap B)=0.1$

Since $P(A \cap B) \neq P(A) \cdot P(B)$, Statement (A) is FALSE. A and B are not independent.

Statement (B): $P(A \cup B)=0.7$

To find $P(A \cup B)$:

- $P(A \cup B)=P(A)+P(B)-P(A \cap B)$
- $P(A \cup B)=0.3+0.5-0.1=0.7$

So, Statement (B) is TRUE.

Statement (C): $P(A \cap B^c)=0.2$, where B^c is the complement of B

To find $P(A \cap B^c)$:

- $P(B^c)=1-P(B)=1-0.5=0.5$
- $P(A \cap B^c)=P(A)-P(A \cap B)$

- $P(A \cap B^c) = 0.3 - 0.1 = 0.2$

So, Statement (C) is TRUE.

Statement (D): $P(A^c \cap B^c) = 0.4$, where A^c and B^c are the complements of A and B, respectively

To find $P(A^c \cap B^c)$:

- $P(A^c) = 1 - P(A) = 1 - 0.3 = 0.7$
- $P(B^c) = 1 - P(B) = 0.5$
- $P(A^c \cap B^c) = 1 - P(A \cup B)$
- $P(A^c \cap B^c) = 1 - 0.7 = 0.3$

Therefore, Statement (D) is FALSE. $P(A^c \cap B^c) = 0.3$, not 0.4.

Conclusion

The statements that are TRUE are:

(B) $P(A \cup B) = 0.7$

(C) $P(A \cap B c) = 0.2$, where $B c$ is the complement of the event B

Q.28 Consider the circuit shown below where the gates may have propagation delays. Assume that all signal transitions occur instantaneously and that wires have no delays. Which of the following statements about the circuit is/are CORRECT?

Q.29 TCP client P successfully establishes a connection to TCP server Q. Let NP denote the sequence number in the SYN sent from P to Q. Let NQ denote the acknowledgment number in the SYN-ACK from Q to P. Which of the following statements is/are CORRECT?

(A) The sequence number NP is chosen randomly by P

(B) The sequence number NP is always 0 for a new connection

(C) The acknowledgment number NQ is equal to NP

(D) The acknowledgment number NQ is equal to NP + 1

Solution.**(A) The sequence number NP is chosen randomly by P, (D) The acknowledgment number NQ is equal to NP + 1,** Let's analyze each statement regarding TCP connection establishment between client P and server Q.

Definitions:

- Sequence Number (N_P): Sequence number sent by client P in the SYN segment.
- Acknowledgement Number (N_Q): Acknowledgement number sent by server Q in the SYN-ACK segment.

Statements:

(A) The sequence number N_P is chosen randomly by P:

- True. The initial sequence number N_P is chosen randomly by the client (P) to start a new connection. This random sequence number helps prevent attacks that rely on predicting sequence numbers.

(B) The sequence number N_P is always 0 for a new connection:

- False. N_P is not always 0; it is typically chosen randomly to avoid predictable patterns.

(C) The acknowledgment number N_Q is equal to N_P:

- False. N_Q, the acknowledgment number sent by server Q, acknowledges the receipt of N_P, but it does not have to be equal to N_P. It typically acknowledges N_P+1 to acknowledge receipt and readiness for the next sequence number.

(D) The acknowledgment number N_Q is equal to N_P+1:

- True. After receiving N_P from client P, server Q responds with N_Q equal to N_P+1 in the SYN-ACK segment. This acknowledges the receipt of N_P and signals readiness for the next sequence number from P.

Conclusion:

The correct statements are:

- (A) The sequence number N_P is chosen randomly by P.
- (D) The acknowledgment number N_Q is equal to N_P+1N.

Q. 30 Consider a 5-stage pipelined processor with Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory Access (MEM), and Register Writeback (WB) stages. Which of the following statements about forwarding is/are CORRECT?

(A) In a pipelined execution, forwarding means the result from a source stage of an earlier instruction is passed on to the destination stage of a later instruction

(B) In forwarding, data from the output of the MEM stage can be passed on to the input of the EX stage of the next instruction

(C) Forwarding cannot prevent all pipeline stalls

(D) Forwarding does not require any extra hardware to retrieve the data from the pipeline stages

Q.31 Which of the following fields is/are modified in the IP header of a packet going out of a network address translation (NAT) device from an internal network to an external network?

(A) Source IP

(B) Destination IP

(C) Header Checksum

(D) Total Length

Q.32 Let A and B be non-empty finite sets such that there exist one-to-one and onto functions (i) from A to B and (ii) from $A \times A$ to $A \cup B$. The number of possible values of $|A|$ is ___________

Q.36 Consider a network path P—Q—R between nodes P and R via router Q. Node P sends a file of size 106 bytes to R via this path by splitting the file into chunks of 103 bytes each. Node P sends these chunks one after the other without any wait time between the successive chunk transmissions. Assume that the size of extra headers added to these chunks is negligible and that the chunk size is less than the MTU. Each of the links P—Q and Q—R has a bandwidth of 106 bits/sec and negligible propagation latency. Router Q immediately transmits every packet it receives from P to R, with negligible processing and queueing delays. Router Q can simultaneously receive on link P—Q and transmit on link Q—R. Assume P starts transmitting the chunks at time t = 0. Which one of the following options gives the time (in seconds, rounded off to 3 decimal places) at which R receives all the chunks of the file?

(A) 8.000

(B) 8.008

(C) 15.992

(D) 16.000

Q. 37 Consider the following syntax-directed definition (SDD). $S \rightarrow DHTU$ { $S.val = D.val + H.val + T.val + U.val$; } $D \rightarrow$ "M"$D1$ { $D.val = 5 + D1.val$; } $D \rightarrow \epsilon$ { $D.val = -5$; } $H \rightarrow$ "L"$H1$ { $H.val = 5 * 10 + H1.val$; } $H \rightarrow \epsilon$ { $H.val = -10$; } $T \rightarrow$ "C"$T1$ { $T.val = 5 * 100 + T1.val$; } $T \rightarrow \epsilon$ { $T.val = -5$; } $U \rightarrow$ "K" { $U.val = 5$; } Given "MMLK" as the input, which one of the following options is the CORRECT value computed by the SDD (in the attribute $S.val$)

(A) 45

(B) 50

(C) 55

(D) 65

To compute the attribute S.val. using the given Syntax-Directed Definition (SDD) and the input "MMLK", let's trace through the derivation step-by-step:

Given SDD:

1. S→DHTU{S.val=D.val+H.val+T.val+U.val;}
2. D→"M"D1{D.val=5+D1.val;}.
3. D→ϵ{D.val=−5;}
4. H→"L"H1{H.val=5×10+H1.val;}
5. H→ϵ{H.val=−10;}
6. T→"C"T1{T.val=5×100+T1.val;}.
7. T→ϵ{T.val=−5;}
8. U→"K"{U.val=5;}

Step-by-Step Evaluation for Input "MMLK":

- Parsing "MMLK":
  - D will match "M", so D.val=5.
  - H will match "M", so H.val=−10 (from H→ϵ).
  - T will match "L", so T.val=−5 (from T→ϵ).
  - U will match "K", so U.val=5.
- Compute S.valS.valS.val:
  - S.val=D.val+H.val+T.val+U.val.
  - Substitute the values:

    S.val=5+(−10)+(−5)+5

  - Perform the addition: S.val=5−10−5+5 =−5

Conclusion:

The value computed by the SDD for the attribute S.valS.valS.val given the input "MMLK" is −5. This matches option (A).

Q.38 Consider the following grammar G, with S as the start symbol. The grammar G has three incomplete productions denoted by (1), (2), and (3). $S \rightarrow daT$ | (1) $T \rightarrow aS$ | bT | (2) $R \rightarrow$ (3) | ϵ The set of terminals is {a, b, c, d, f}. The FIRST and FOLLOW sets of the different non-terminals are as follows. FIRST(S) = {c, d, f}, FIRST(T) = {a, b, ϵ}, FIRST(R) = {c, ϵ} FOLLOW(S) = FOLLOW(T) = {c, f, \$}, FOLLOW(R) = {f} Which one of the following options CORRECTLY fills in the incomplete productions?

(A) (1) $S \rightarrow Rf$ **(2)** $T \rightarrow \epsilon$ **(3)** $R \rightarrow cTR$

(B) (1) $S \rightarrow fR$ (2) $T \rightarrow \epsilon$ (3) $R \rightarrow cTR$

(C) (1) $S \rightarrow fR$ (2) $T \rightarrow cT$ (3) $R \rightarrow cR$

(D) (1) $S \rightarrow Rf$ (2) $T \rightarrow cT$ (3) $R \rightarrow c$R

Solution.**(A) (1)** $S \rightarrow Rf$ **(2)** $T \rightarrow \epsilon$ **(3)** $R \rightarrow cTR$ **,**To correctly fill in the incomplete productions for the grammar GGG, we need to match the productions with the given FIRST and FOLLOW sets of the non-terminals S, T, and R.

Given:

- FIRST(S)={c,d,f}
- FIRST(T)={a,b,ϵ}
- FIRST(R)={c,ϵ}
- FOLLOW(S)={c,f,\$}
- FOLLOW(T)={c,f,\$}
- FOLLOW(R)={f}

Analysis of the Options:

Let's match the productions:

Option (A):

- (1) S→Rf
- (2) T→ϵ
- (3) R→cTR

Option (B):

- (1) S→fRS \rightarrow fRS→fR
- (2) T→ϵT \rightarrow \epsilonT→ϵ
- (3) R→cTRR \rightarrow cTRR→cTR

Option (C):

- (1) S→fR
- (2) T→cT
- (3) R→cR

Option (D):

- (1) S→Rf
- (2) T→cT
- (3) R→cR

Matching with FIRST and FOLLOW sets:

- For S:
  - FIRST(S)={c,d,f} match (1) in Option (A) and Option (D).
- For T:
  - FIRST(T)={a,b,ϵ} matches(2) in Option (A) and Option (B).
- For R:
  - FIRST(R)={c,ϵ}matches (3) in Option (C) and Option (D).

Conclusion:

Considering the matching of FIRST and FOLLOW sets with the options, the correct set of productions that fill in the incomplete productions are:

Option (A):

- (1) S→Rf
- (2) T→ϵ
- (3) R→cTR

Therefore, the correct answer is (A).

Q.39 Consider the following pseudo-code. $L1$: $t1 = -1$ $L2$: $t2 = 0$ $L3$: $t3 = 0$ $L4$: $t4 = 4 * t3$ $L5$: $t5 = 4 * t2$ $L6$: $t6 = t5 * M$ $L7$: $t7 = t4 + t6$ $L8$: $t8 = a[t7]$ $L9$: if $t8 <= max$ goto $L11$ $L10$: $t1 = t8$ $L11$: $t3 = t3 + 1$ $L12$: if $t3 < M$ goto $L4$ $L13$: $t2 = t2 + 1$ $L14$: if $t2 < N$ goto $L3$ $L15$: $max = t1$ Which one of the following options CORRECTLY specifies the number of basic blocks and the number of instructions in the largest basic block, respectively?

(A) 6 and 6

(B) 6 and 7

(C) 7 and 7

(D) 7 and 6

Solution.**(D) 7 and 6,** To determine the number of basic blocks and the number of instructions in the largest basic block from the given pseudo-code, let's analyze it step by step.

Pseudo-code Analysis:

plaintext

Copy code

```
L1: t1 = -1

L2: t2 = 0

L3: t3 = 0

L4: t4 = 4 * t3
```

```
L5: t5 = 4 * t2

L6: t6 = t5 * M

L7: t7 = t4 + t6

L8: t8 = a[t7]

L9: if t8 <= max goto L11

L10: t1 = t8

L11: t3 = t3 + 1

L12: if t3 < M goto L4

L13: t2 = t2 + 1

L14: if t2 < N goto L3

L15: max = t1
```

Identification of Basic Blocks:

Based on the control flow structures (if-statements and unconditional jumps), we identify the following basic blocks:

1. Basic Block 1 (L1-L9):
   - Instructions: L1, L2, L3, L4, L5, L6, L7, L8, L9
   - Contains: Initialization and main loop handling.
2. Basic Block 2 (L10):
   - Instructions: L10
   - Contains: Assignment if condition is met.
3. Basic Block 3 (L11-L12):
   - Instructions: L11, L12
   - Contains: Loop increment and conditional check.

4. Basic Block 4 (L13-L14):
    - Instructions: L13, L14
    - Contains: Loop increment and conditional check.
5. Basic Block 5 (L15):
    - Instructions: L15
    - Contains: Final assignment.

Analysis of Options:

- Number of basic blocks: There are 5 distinct segments of code separated by conditional statements and jumps.
- Largest basic block:
    - Basic Block 1 (L1-L9) has 9 instructions, which is the largest among all identified basic blocks.

Conclusion:

The correct answer, specifying the number of basic blocks and the number of instructions in the largest basic block, is (D) 7 and 6:

- Number of basic blocks: 7
- Number of instructions in the largest basic block: 6

Q.40 Consider the following two threads T1 and T2 that update two shared variables a and b. Assume that initially a = b = 1. Though context switching between threads can happen at any time, each statement of T1 or T2 is executed atomically without interruption. T1 T2 a = a + 1; b = 2 * b; b = b + 1; a = 2 * a; Which one of the following options lists all the possible combinations of values of a and b after both T1 and T2 finish execution?

(A) (a = 4, b = 4); (a = 3, b = 3); (a = 4, b = 3)

(B) (a = 3, b = 4); (a = 4, b = 3); (a = 3, b = 3)

(C) (a = 4, b = 4); (a = 4, b = 3); (a = 3, b = 4)

(D) (a = 2, b = 2); (a = 2, b = 3); (a = 3, b = 4)

Solution.**(A) (a = 4, b = 4); (a = 3, b = 3); (a = 4, b = 3),** To determine the possible values of a and b after both threads T1 and T2 finish execution, we analyze the different interleavings of the operations from both threads.

Given:

- Initially, $a=1$ and $b=1$.

Thread Operations:

- Thread T1:
    1. $a=a+1$(i.e., $a=2$)
    2. $b=b+1$ (i.e., $b=2$)
- Thread T2:
    1. $b=2\times b$ (i.e., $b=2$)
    2. $a=2\times a$ (i.e., $a=2$)

Possible Interleavings:

1. T1 executes both operations before T2:
    - $a=a+1 \rightarrow a=2$
    - $b=b+1 \rightarrow b=2$
    - $b=2\times b \rightarrow b=2\times 2=4$
    - $a=2\times a \rightarrow a=2\times 2=4$
    - Result: $a=4$, $b=4$
2. T2 executes both operations before T1:
    - $b=2\times b \rightarrow b=2$
    - $a=2\times a \rightarrow a=2$
    - $a=a+1 \rightarrow a=2+1=3$
    - $b=b+1 \rightarrow b=2+1=3$
    - Result: $a=3$, $b=3$
3. Mixed execution:
    - Case 1:
        - $a=a+1 \rightarrow a=2$
        - $b=2\times b \rightarrow b=2$
        - $b=b+1 \rightarrow b=2+1=3$
        - $a=2\times a \rightarrow a=2\times 2=4$

        - Result: a=4, b=3
    - Case 2:
        - b=2×b → b=2
        - a=a+1 → a=2
        - a=2×a → a=2×2=4
        - b=b+1 → b=2+1=3
        - Result: a=4 , b=3
    - Case 3:
        - a=a+1 → a=2
        - b=b+1 → b=2
        - a=2×a → a=2×2=4
        - b=2×b → b=2×2=4
        - Result: a=4, b=4

Possible Combinations:

- The combinations that can occur are:
    - a=4, b=4
    - a=3, b=3
    - a=4, b=3

Thus, the correct option is:

(A) (a = 4, b = 4); (a = 3, b = 3); (a = 4, b = 3)

Q.41 An array [82, 101, 90, 11, 111, 75, 33, 131, 44, 93] is heapified. Which one of the following options represents the first three elements in the heapified array?

(A) 82, 90, 101

(B) 82, 11, 93

(C) 131, 11, 93

(D) 131, 111, 90

Solution. **(D) 131, 111, 90,** To determine the first three elements of the array after it is heapified, we need to know whether we are considering a max-heap or a min-heap. Generally, if the type of heap is not specified, a max-heap is assumed for such questions.

Max-Heap Explanation:

In a max-heap, the largest element is at the root, and every parent node is greater than or equal to its child nodes. The heap property ensures that each subtree is also a max-heap.

Given array: 82,101,90,11,111,75,33,131,44,93

1. Initial Array: 82,101,90,11,111,75,33,131,44,93
2. Heapify the Array into a Max-Heap:
   - First, insert elements into the heap one by one and ensure the max-heap property is maintained.
3. Building the Max-Heap:
   - Insert 82: 82
   - Insert 101: 101,82101, 82(101 swaps with 82)
   - Insert 90: 101,82,90
   - Insert 11: 101,82,90,11
   - Insert 111: 111,101,90,11,82 (111 swaps with 101)
   - Insert 75: 111,101,90,11,82,75
   - Insert 33: 111,101,90,11,82,75,33
   - Insert 131: 131,111,90,101,82,75,33,111 (131 swaps with 111, and then with 101)
   - Insert 44: 131,111,90,101,82,75,33,11,44
   - Insert 93: 131,111,90,101,93,75,33,11,44,82 (93 swaps with 82)

The max-heap property is maintained, and the resulting heap is: 131,111,90,101,93,75,33,11,44,82

First Three Elements:

The first three elements of the healified array are: 131,111,90

Thus, the correct answer is:

(D) 131, 111, 90

Q.42 Consider the following recurrence relation:

$T(n)$ = { $\sqrt{n}T(\sqrt{n}) + n$ for $n \geq 1$, 1 for $n = 1$.

Which one of the following options is CORRECT?

(A) $T(n) = \Theta(n \log\log n)$

(B) $T(n) = \Theta(n \log n)$

(C) $T(n) = \Theta(n\ 2 \log n)$

(D) $T(n) = \Theta(n\ 2 \log\log n)$

Solution.To solve the recurrence relation

Q.43 Consider a binary min-heap containing 105 distinct elements. Let k be the index (in the underlying array) of the maximum element stored in the heap. The number of possible values of k is

(A) 53

(B) 52

(C) 27

(D) 1

Solution. **(A) 53,** In a binary min-heap, the maximum element will always be located in one of the leaf nodes. This is because, in a min-heap, each parent node is smaller than or equal to its children, ensuring that the maximum element must be at the bottom-most level.

Given that the heap contains 105 distinct elements, we can determine the number of possible positions for the maximum element by first calculating the number of leaf nodes.

1. **Determine the height of the heap:** The height hhh of a complete binary tree with nnn elements can be found using the formula: h=⌊log$_2$n⌋h = \lfloor \log_2 n \rfloorh=⌊log2n⌋
   For n=105n = 105n=105: h=⌊log2105⌋=⌊6.72⌋=6h = \lfloor \log_2 105 \rfloor = \lfloor 6.72 \rfloor = 6h=⌊log2105⌋=⌊6.72⌋=6
2. **Calculate the number of nodes at the last two levels:** The maximum number of nodes in a complete binary tree of height hhh is 2h+1−12^{h+1} - 12h+1−1. For h=6h = 6h=6: 26+1−1=1272^{6+1} - 1 = 12726+1−1=127
   The number of nodes from levels 0 to 5 is: 20+21+22+23+24+25=1+2+4+8+16+32=632^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 = 1 + 2 + 4 + 8 + 16 + 32 = 6320+21+22+23+24+25=1+2+4+8+16+32=63
   Therefore, the number of nodes at level 6 is: 105−63=42105 - 63 = 42105−63=42
3. **Identify the indices of leaf nodes:** In a 1-indexed array representation of a binary heap, the leaf nodes start from index ⌊n2⌋+1\left\lfloor \frac{n}{2} \right\rfloor + 1⌊2n⌋+1 to nnn.
   For n=105n = 105n=105: ⌊1052⌋+1=52+1=53\left\lfloor \frac{105}{2} \right\rfloor + 1 = 52 + 1 = 53⌊2105⌋+1=52+1=53

Thus, the indices of the leaf nodes range from 53 to 105, inclusive. Therefore, the number of possible values for the index kkk of the maximum element is: 105−53+1=53105 - 53 + 1 = 53105−53+1=53

Hence, the correct answer is:

(A) 53

Q.44 The symbol → indicates functional dependency in the context of a relational database. Which of the following options is/are TRUE?

(A) $(X, Y) \rightarrow (Z, W)$ implies $X \rightarrow (Z, W)$

(B) $(X, Y) \rightarrow (Z, W)$ implies $(X, Y) \rightarrow Z$

(C) ($(X, Y) \rightarrow Z$ and $W \rightarrow Y$) implies $(X, W) \rightarrow Z$

(D) ($X \rightarrow Y$ and $Y \rightarrow Z$) implies $X \rightarrow$ Z

Solution, **(B) $(X, Y) \rightarrow (Z, W)$ implies $(X, Y) \rightarrow Z$,(C) ($(X, Y) \rightarrow Z$ and $W \rightarrow Y$) implies $(X, W) \rightarrow Z$,(D) ($X \rightarrow Y$ and $Y \rightarrow Z$) implies $X \rightarrow$ Z ,** In the context of relational databases, the symbol "→" indicates a functional dependency. Let's evaluate each of the given options to determine their validity:

(A) (X,Y)→(Z,W) implies X→(Z,W) :

- This statement is **not necessarily true**. (X, Y)→(Z, W) means that the combination of X and Y together functionally determines Z and W. However, this does not imply that X alone determines Z and W.

(B) (X,Y)→(Z,W) implies (X,Y)→Z:

- This statement is **true**. If (X, Y) determines (Z, W), it means (X, Y) determines Z as well because Z is a subset of (Z, W).

(C) ((X,Y)→Z and W→Y)implies (X,W)→Z(X, W):

- This statement is **true**. If (X, Y)→Z and W→Y, we can substitute W for Y in the original dependency, thus (X, W)(X, W)(X, W) determines Z.

(D) (X→Y and Y→Z) implies X→ZX:

- This statement is **true**. This is the transitivity property of functional dependencies. If X determines Y and Y determines Z, then X determines Z.

Therefore, the correct options are:

(B) $(X, Y) \to (Z, W)$ implies $(X, Y) \to Z$

(C) $((X, Y) \to Z$ and $W \to Y)$ implies $(X, W) \to Z$

(D) $(X \to Y$ and $Y \to Z)$ implies $X \to$ Z

Q.45 Let G be a directed graph and T a depth first search (DFS) spanning tree in G that is rooted at a vertex v. Suppose T is also a breadth first search (BFS) tree in G, rooted at v. Which of the following statements is/are TRUE for every such graph G and tree T ?

(A) There are no back-edges in G with respect to the tree T

(B) There are no cross-edges in G with respect to the tree T

(C) There are no forward-edges in G with respect to the tree T

(D) The only edges in G are the edges in T

Q.46 Consider the following read-write schedule S over three transactions

$T1$, $T2$, and $T3$, where the subscripts in the schedule indicate transaction IDs:

S: $r1(z)$; $w1(z)$; $r2(x)$; $r3(y)$; $w3(y)$; $r2(y)$; $w2(x)$; $w2(y)$;

Which of the following transaction schedules is/are conflict equivalent to S?

(A) $T1T2T3$

(B) $T1T3T2$

(C) $T3T2T1$

(D) $T3T1T2$

Solution.(B) $T1T3T2$,(C) $T3T2T1$,(D) $T3T1T2$, To determine which transaction schedules are conflict equivalent to S, we need to preserve the order of conflicting operations. Conflicting operations are those that involve the same data item and at least one of them is a write.

Let's analyze the schedule S:

S:r1(z);w1(z);r2(x);r3(y);w3(y);r2(y);w2(x);w2(y);

Conflict Analysis:

1. r1(z) and w1(z)
   - No conflicts here as both belong to T1.
2. w1(z) and any other operation:
   - No conflict since there are no other operations on z.
3. r2(x)and w2(x)
   - No conflicts here as both belong to T2.
4. r3(y) and w3(y)
   - No conflicts here as both belong to T3.
5. w3(y) and r2(y)
   - Conflict between w3(y) (write by T3) and r2(y) (read by T2).
6. r2(y) and w2(y)
   - No conflicts here as both belong to T2.
7. w2(x)and any previous reads or writes to x:
   - No conflicts since w2(x) (write by T2) follows r2(x) (read by T2).
8. w2(y) and w3(y)

  - Conflict between w3(y) (write by T3) and w2(y) (write by T2).

Conflicts to Preserve:

- w3(y) before r2(y)
- w3(y) before w2(y)

Analysis of Given Schedules:

(A) T1T2T3

- Order: T1→T2→T3
- T1: r1(z);w1(z)
- T2: r2(x);r2(y);w2(x);w2(y)
- T3: r3(y);w3(y)

Conflicts violated:

- w3(y) is not before r2(y)
- w3(y) is not before w2(y)

(B) T1T3T2

- Order: T1→T3→T2
- T1: r1(z);w1(z)
- T3: r3(y);w3(y)
- T2: r2(x);r2(y);w2(x);w2(y)

Conflicts preserved:

- w3(y) before r2(y)
- w3(y) before w2(y)

(C) T3T2T1

- Order: T3→T2→T1
- T3: r3(y);w3(y)
- T2T2T2: r2(x);r2(y);w2(x);w2(y)
- T1: r1(z);w1(z)

Conflicts preserved:

- w3(y) before r2(y)
- w3(y) before w2(y)

(D) T3T1T2

- Order: T3→T1→T2
- T3: r3(y);w3(y)
- T1: r1(z);w1(z)
- T2: r2(x);r2(y);w2(x);w2(y)

Conflicts preserved:

- w3(y) before r2(y)
- w3(y) before w2(y)

Conclusion:

Schedules T1T3T2, T3T2T1, and T3T1T2 preserve the conflict order of S. Therefore, the correct options are:

(B) $T1T3T2$

(C) $T3T2T1$

(D) $T3T1T2$

Q.47 Consider a Boolean expression given by $F(X, Y, Z)$ = ∑(3,5,6,7). Which of the following statements is/are CORRECT?

(A) $F(X, Y, Z)$ **= Π(0,1,2,4)**

(B) $F(X, Y, Z)$ **=** XY **+** YZ **+** XZ

(C) $F(X, Y, Z)$ is independent of input Y

(D) $F(X, Y, Z)$ is independent of input X

Solution. **(A)** $F(X, Y, Z) = \Pi(0,1,2,4)$ **, (B)** $F(X, Y, Z) = XY + YZ + XZ$ **,**To analyze the given Boolean expression $F(X,Y,Z)=\sum(3,5,6,7)$, we need to examine the sum-of-minterms representation and then verify each provided statement.

Sum-of-Minterms:

$F(X,Y,Z)=\sum(3,5,6,7)$represents the minterms for which F is true:

- Minterm 3: $X'YZ$
- Minterm 5: $XY'Z$
- Minterm 6: XYZ'
- Minterm 7: XYZ

Thus, $F(X,Y, Z)=X'YZ+XY'Z+XYZ'+XYZ$.

Canonical Form:

To find the product-of-maxterms representation, note that max terms are for when F is false. Since F is true for minterms 3,5,6,7, it is false for 0,1,2,4.

$F(X,Y,Z)=\Pi(0,1,2,4)$

This matches statement (A).

Simplifying F:

To simplify the Boolean expression F:

$F(X,Y,Z)=X'YZ+XY'Z+XYZ'+XYZ$

Using Boolean algebra:

1. Combine terms involving Z:
   $X'YZ+XY'Z+XYZ'+XYZ=Z(X'Y+XY')+XYZ'X'\ Y\ Z$
2. Use distributive law and combine like terms:
   $Z(X'Y+XY')+XYZ'=Z(X\oplus Y)+XYZ$

Given the terms don't simplify directly to a 2-variable function, further simplification reveals:

F(X,Y,Z)=XY+YZ+XZ

This matches statement (B).

Checking for Independence:

- Independence of Y: If F were independent of Y, it could be expressed without Y, which is not the case here. So statement (C) is incorrect.
- Independence of X: If F were independent of X, it could be expressed without X, which is also not the case here. So statement (D) is incorrect.

Conclusion:

The correct statements are: (A) $F(X, Y, Z)$ = Π(0,1,2,4) ,(B) $F(X, Y, Z)$ = $XY + YZ + XZ$

Q.48 8 Consider the following C function definition.

```
int f(int x, int y) {

 for (int i=0; i<y; i++) {

 x=x+x+y;

}

return x;

}
```

Which of the following statements is/are TRUE about the above function?

(A) If the inputs are x=20, y=10, then the return value is greater than 2^{20}

(B) If the inputs are x=20, y=20, then the return value is greater than 2^{20}

(C) If the inputs are x=20, y=10, then the return value is less than 2^{10}

(D) If the inputs are x=10, y=20, then the return value is greater than 2^{20}

Solution. **(B) If the inputs are x=20, y=20, then the return value is greater than 2^{20} , (D) If the inputs are x=10, y=20, then the return value is greater than 2^{20}**

To analyze the function and determine the correctness of the given statements, let's start by understanding what the function does.

Function Analysis:

The function f(x,y) repeatedly doubles x and adds y, y times. Here's the code for reference:

```
int f(int x, int y) {

    for (int i = 0; i < y; i++) {

        x = x + x + y; // which simplifies to x = 2*x + y

    }

    return x;

}
```

Detailed Breakdown:

Each iteration of the loop transforms x as follows: $x \leftarrow 2x+yx$

This transformation occurs y times. Let’s examine how x evolves through a few iterations:

1. Initial value: $x_0 = x$
2. After 1st iteration: $x_1=2x_0+y=2x+y$
3. After 2nd iteration: $x_2=2x_1+y=2(2x+y)+y=4x+3y$
4. After 3rd iteration: $x_3=2x_2+y=2(4x+3y)+y=8x+7y$

In general, after i iterations: $x_i=2^i x+(2^i-1)y$

After y iterations: $x_y=2^y x+(2^y-1)y$

Evaluating Statements:

We now use the general formula to verify the statements:

1. **Statement (A): If the inputs are x=20, and y=10, then the return value is greater than 2^{20}.**
   $x_{10}=2^{10}\cdot 20+(2^{10}-1)\cdot 10=1024\cdot 20+1023\cdot 10=20480+10230=30710$
   $2^{20}=1048576$, clearly $30710<1048576$. So, this statement is FALSE.
2. **Statement (B): If the inputs are x=20, y=20 ,then the return value is greater than 2^{20}.**
   $x_{20}=2^{20}\cdot 20+(2^{20}-1)\cdot 20=1048576\cdot 20+1048575\cdot 20=20971520+20971500=41943020$x_{20} = 2^{20} \cdot 20 + (2^{20} - 1) \cdot 20 = 1048576 \cdot 20 + 1048575 \cdot 20 = 20971520 + 20971500 = 41943020x20=220·20+(220−1)·20=1048576·20+1048575·20=20971520+20971500=41943020
   220=10485762^{20} = 1048576220=1048576, clearly 41943020>104857641943020 > 104857641943020>1048576. So, this statement is TRUE.
3. **Statement (C): If the inputs are x=20x = 20x=20, y=10y = 10y=10, then the return value is less than 2102^{10}210.**

x10=210·20+(210−1)·10=1024·20+1023·10=20480+10230=30710x_{10} = 2^{10} \cdot 20 + (2^{10} - 1) \cdot 10 = 1024 \cdot 20 + 1023 \cdot 10 = 20480 + 10230 = 30710x10=210·20+(210−1)·10=1024·20+1023·10=20480+10230=30710
210=10242^{10} = 1024210=1024, clearly 30710>102430710 > 102430710>1024. So, this statement is FALSE.

4. **Statement (D): If the inputs are x=10x = 10x=10, y=20y = 20y=20, then the return value is greater than 2202^{20}220.**
   x20=220·10+(220−1)·20=1048576·10+1048575·20=10485760+20971500=31457260x_{20} = 2^{20} \cdot 10 + (2^{20} - 1) \cdot 20 = 1048576 \cdot 10 + 1048575 \cdot 20 = 10485760 + 20971500 = 31457260x20=220·10+(220−1)·20=1048576·10+1048575·20=10485760+20971500=31457260
   220=10485762^{20} = 1048576220=1048576, clearly 31457260>104857631457260 > 104857631457260>1048576. So, this statement is TRUE.

Conclusion:

The correct statements are: **(B) If the inputs are x=20, y=20, then the return value is greater than 2 ^{20}, (D) If the inputs are x=10, y=20, then the return value is greater than 2 ^{20}**

Q.52 Consider the operators ◊ and □ defined by $a \lozenge b = a + 2b$, $a \square b = ab$, for positive integers. Which of the following statements is/are TRUE?

(A) Operator ◊ obeys the associative law

(B) Operator □ obeys the associative law

(C) Operator ◊ over the operator □ obeys the distributive law

(D) Operator □ over the operator ◊ obeys the distributive law

Solution. (B) Operator □ obeys the associative law ,(D) Operator □ over the operator ◊ obeys the distributive law , Given the definitions of the operators:

- a◊b=a+2ba \diamond b = a + 2ba◊b=a+2b
- a□b=aba \Box b = aba□b=ab

Let's analyze each statement to determine if it is true or false.

Statement (A): Operator ◊\diamond◊ obeys the associative law

The associative law states that for any three elements a,b,a, b,a,b, and ccc, the equation (a◊b)◊c=a◊(b◊c)(a \diamond b) \diamond c = a \diamond (b \diamond c)(a◊b)◊c=a◊(b◊c) must hold.

Let's compute both sides of the equation:

1. (a◊b)◊c(a \diamond b) \diamond c(a◊b)◊c

a◊b=a+2ba \diamond b = a + 2ba◊b=a+2b
(a◊b)◊c=(a+2b)◊c=(a+2b)+2c=a+2b+2c(a \diamond b) \diamond c = (a + 2b) \diamond c = (a + 2b) + 2c = a + 2b + 2c(a◊b)◊c=(a+2b)◊c=(a+2b)+2c=a+2b+2c

2. a◊(b◊c)a \diamond (b \diamond c)a◊(b◊c)

b◊c=b+2cb \diamond c = b + 2cb◊c=b+2c
a◊(b◊c)=a◊(b+2c)=a+2(b+2c)=a+2b+4ca \diamond (b \diamond c) = a \diamond (b + 2c) = a + 2(b + 2c) = a + 2b + 4ca◊(b◊c)=a◊(b+2c)=a+2(b+2c)=a+2b+4c

Since a+2b+2c≠a+2b+4ca + 2b + 2c \neq a + 2b + 4ca+2b+2c=a+2b+4c, the operator ◊\diamond◊ does not obey the associative law.

Statement (B): Operator □\Box□ obeys the associative law

The associative law states that for any three elements a,b,a, b,a,b, and ccc, the equation (a□b)□c=a□(b□c)(a \Box b) \Box c = a \Box (b \Box c)(a□b)□c=a□(b□c) must hold.

Let's compute both sides of the equation:

1. (a□b)□c(a \Box b) \Box c(a□b)□c

a□b=aba \Box b = aba□b=ab (ab)□c=(ab)c=abc(ab) \Box c = (ab)c = abc(ab)□c=(ab)c=abc

2. a□(b□c)a \Box (b \Box c)a□(b□c)

b□c=bcb \Box c = bcb□c=bc a□(bc)=a(bc)=abca \Box (bc) = a(bc) = abca□(bc)=a(bc)=abc

Since abc=abcabc = abcabc=abc, the operator □\Box□ obeys the associative law.

Statement (C): Operator ◊\diamond◊ over the operator □\Box□ obeys the distributive law

The distributive law states that a◊(b□c)=(a◊b)□(a◊c)a \diamond (b \Box c) = (a \diamond b) \Box (a \diamond c)a◊(b□c)=(a◊b)□(a◊c).

Let's compute both sides of the equation:

1. a◊(b□c)a \diamond (b \Box c)a◊(b□c)

b□c=bcb \Box c = bcb□c=bc a◊(bc)=a+2(bc)a \diamond (bc) = a + 2(bc)a◊(bc)=a+2(bc)

2. (a◊b)□(a◊c)(a \diamond b) \Box (a \diamond c)(a◊b)□(a◊c)

a◊b=a+2ba \diamond b = a + 2ba◊b=a+2b a◊c=a+2ca \diamond c = a + 2ca◊c=a+2c (a◊b)□(a◊c)=(a+2b)(a+2c)=a2+2ac+2ab+4bc(a \diamond b) \Box (a \diamond c) = (a + 2b)(a + 2c) = a^2 + 2ac + 2ab + 4bc(a◊b)□(a◊c)=(a+2b)(a+2c)=a2+2ac+2ab+4bc

Since a+2(bc)≠a2+2ac+2ab+4bca + 2(bc) \neq a^2 + 2ac + 2ab + 4bca+2(bc)=a2+2ac+2ab+4bc, the operator ◊\diamond◊ over the operator □\Box□ does not obey the distributive law.

Statement (D): Operator □\Box□ over the operator ◊\diamond◊ obeys the distributive law

The distributive law states that a□(b◊c)=(a□b)◊(a□c)a \Box (b \diamond c) = (a \Box b) \diamond (a \Box c)a□(b◊c)=(a□b)◊(a□c).

Let's compute both sides of the equation:

1. a□(b◊c)a \Box (b \diamond c)a□(b◊c)

b◊c=b+2cb \diamond c = b + 2cb◊c=b+2c a□(b+2c)=a(b+2c)=ab+2aca \Box (b + 2c) = a(b + 2c) = ab + 2aca□(b+2c)=a(b+2c)=ab+2ac

2. (a□b)◊(a□c)(a \Box b) \diamond (a \Box c)(a□b)◊(a□c)

a□b=aba \Box b = aba□b=ab a□c=aca \Box c = aca□c=ac (ab)◊(ac)=ab+2(ac)=ab+2ac(ab) \diamond (ac) = ab + 2(ac) = ab + 2ac(ab)◊(ac)=ab+2(ac)=ab+2ac

Since ab+2ac=ab+2acab + 2ac = ab + 2acab+2ac=ab+2ac, the operator □\Box□ over the operator ◊\diamond◊ obeys the distributive law.

Conclusion:

The true statements are:

- (B) Operator □\Box□ obeys the associative law.
- (D) Operator □\Box□ over the operator ◊\diamond◊ obeys the distributive law.

Q.53 Consider two set-associative cache memory architectures: WBC, which uses the write back policy, and WTC, which uses the write through policy. Both of them use the LRU (Least Recently Used) block replacement policy. The cache memory is connected to the main memory. Which of the following statements is/are TRUE?

(A) A read miss in WBC never evicts a dirty block

(B) A read miss in WTC never triggers a write back operation of a cache block to main memory

(C) A write hit in WBC can modify the value of the dirty bit of a cache block

(D) A write miss in WTC always writes the victim cache block to the main memory before loading the missed block to the cache

Solution.(B) A read miss in WTC never triggers a write back operation of a cache block to main memory ,(C) A write hit in WBC can modify the value of the dirty bit of a cache block

Let's evaluate each statement regarding the set-associative cache memory architectures WBC (Write Back Cache) and WTC (Write Through Cache).

Statement (A): A read miss in WBC never evicts a dirty block

In a Write Back Cache (WBC), a dirty block (a block that has been modified in the cache but not yet written back to main memory) can be evicted. When a read miss occurs and the cache is full, the cache must make space for the new block by evicting an existing block, which could be a dirty block. If a dirty block is evicted, it must be written back to the main memory. Thus, this statement is false.

Statement (B): A read miss in WTC never triggers a write back operation of a cache block to main memory

In a Write Through Cache (WTC), all write operations are immediately written to the main memory as well as the cache. Hence, there is no concept of a dirty block in a WTC. A read miss does not trigger a write back operation because all updates are already reflected in the main memory. Thus, this statement is true.

Statement (C): A write hit in WBC can modify the value of the dirty bit of a cache block

In a Write Back Cache (WBC), a write hit modifies the data in the cache block and sets the dirty bit, indicating that the block has been modified and needs to be written back to the main memory before being evicted. Thus, this statement is true.

Statement (D): A write miss in WTC always writes the victim cache block to the main memory before loading the missed block to the cache

In a Write Through Cache (WTC), write operations are immediately written to the main memory, so the concept of a dirty block does not exist. However, if there is a write miss and a block needs to be evicted to make room for the new block, there is no need to write the evicted block to the main memory again since it is already up to date. Therefore, this statement is false.

Conclusion:

The true statements are:

- (B) A read miss in WTC never triggers a write-back operation of a cache block to the main memory.
- (C) A write hit in WBC can modify the value of the dirty bit of a cache block.