43. The Design Principle "Anticipate Obsolescence" can be achieved by
(a) using undocumented features of software libraries
(b) using reusable software from smaller companies
(c) using standard languages and technologies that are supported by multiple vendors
(d) using deprecated classes

44. The Design Principle "Design for Portability" can be achieved by
(a) avoiding the use of facilities that are specific to one particular environment
(b) anticipating obsolescence
(c) using only big-endian format
(d) using only 2-byte integers

45. Providing a command-line version, in addition to usual graphical user-interface, achieves the goal of
(a) Design defensively
(b) Design for testability
(c) Design for portability
(d) Anticipate obsolescence

GROUP-B

46. A university is admitting students in a professional course, subject to the following **conditions**:

(a) Marks in Java ≥ 70
(b) Marks in C++ ≥ 60
(c) Marks in OOAD ≥ 60
(d) Total in all three subject ≥ 220 **OR**
Total in Java and C++ ≥ 150

If the aggregate marks of an eligible candidate is more than 240, he/she will be eligible for the **scholarship course**, otherwise he/she will be eligible for a **normal course**. The program reads the marks in the three subjects and generates the following **eligibility (outputs)**:

(i) Not eligible
(ii) Eligible for **scholarship course**
(iii) Eligible for **normal course**

(a) Construct a matrix "**Conditions**" of conditions (in rows) vs. rules (in columns). An entry in the matrix will be true(T), or false(F), or immaterial/don't-care (I). Thus, a column is really a vector of **conditions**, the index to the vector being the condition-ID. The rules are the eligibility outputs (i)..(iii) listed above. Several columns may correspond to the same eligibility output because a different set of conditions can lead to the same output. For compactness, the rules may be labelled R1, R2, R3, etc. in the matrix.

(b) Construct another matrix **"Actions"**, just below **"Conditions"**. Its rows are the outputs in (i)..(iii) above, and its columns are identical to that of **"Conditions"**. An entry in this matrix will be full (X) or empty(blank). In other words, a column j represents a particular combination of **conditions**, and a full(X) entry in **Actions**[i,j] specifies that the eligibility in row-i will be applicable for that combination of conditions.

(c) Design **test cases**, one for each column in (a) above. Write your answer in the form of a table with columns: Serial No, Java-marks, C++ marks, OOAD marks, Aggregate-marks, **Eligibility (output)**.

9 + 4 + 7 = 20

END

13 → Kol to NJP

14 → NJP to Kalim to Lava. (IN)

15 → Lava Seight (IN) + Rishop Seight + Lava IN

16 → Rishop Seight + Lava (IN) → Lava to Kalim + Kalim Seight (I

17 → Lava to Kalim + Kalim Seight (IN) → Kalim to Danjee (IN)

18 → Kalim to Danjee (IN) → Danjee Seight (IN)

19 → Danjee Seight 7Pt. + Back to NJP