Logic Based Testing

1) Overview about Logic Based Testing:-

"Logic" is one of the most often used words in programmers vocabularies but one of their least used techniques.

This chapter concerns logic in its simplest form, boolean algebra and its applications to program and specification test and design. Boolean algebra is to logic, as arithmetic is to mathematics.

2) Decision Tables:-

A decision table is a tabular form that consists of a set of conditions and their respective actions. A decision table consists of four parts. They are as follows,

i) Condition stub
ii) Condition entry
iii) Action stub
iv) Action entry.

Condition Stub:- It consists of a list of causes. for a rule to be satisfied, conditions are marked with Yes/No. "Yes" indicates that the condition is satisfied and "No" specifies that the condition is not satisfied. The variable 'I' specifies the immaterial test case, which means that a particular condition doesn't have any role in that rule.

Action Stub:- It describes the possible actions that are to be executed and "NO" specifies that an action need not be executed.

Table1: An Example of a Decision Table

		RULE1	RULE2	RULE 3	RULE 4
CONDITION STUB	CONDITION1	YES	YES	NO	NO
	CONDITION2	YES	I	NO	I
	CONDITION3	NO	YES	NO	I
	CONDITION 4	NO	YES	NO	YES
ACTION STUB	ACTION 1	YES	YES	NO	NO
	ACTION 2	NO	NO	YES	NO
	ACTION 3	NO	NO	NO	YES
		ACTION ENTRY			

1.a) Action 1 will take place if conditions 1 and 2 are met and if conditions 3 and 4 are not met (rule 1), or if conditions 1, 3 and 4 are met (rule 2).

1.b) Action 1 will be taken if Predicates 1 and 2 are true and if Predicates 3 and 4 are false (rule 1), or if Predicates 1, 3 and 4 are true (rule 2).

2. Action 2 will be taken if the Predicates are all false (rule 3).

3. Action 3 will take place if Predicate 1 is false and Predicate 4 is true (rule 4).

	RULE 5	RULE 6	RULE 7	RULE 8
CONDITION1	I	NO	YES	YES
CONDITION2	I	YES	I	NO
CONDITION3	YES	I	NO	NO
CONDITION4	NO	NO	YES	I
DEFAULT ACTION	YES	YES	YES	YES

Table: the Default Rules for Table 1.

Applications of Decision Tables:-

1) Decision tables are used in business data Processing.

2) Decision table processors are generally used for generating a high-level code. The translator in the Processor is used to examine the decision table for consistency and completeness. These processors are implemented using boolean algebra.

3) Decision tables are used in many application areas like business analysis, Programming, testing, design of hardware.

Decision-Table Processors:-

Decision tables represent higher-level language, as they can be translated into Program code automatically. The translator of decision table is responsible for checking whether the source decision table is reliable and complete. It also checks whether any default rules are required and specifies them if needed.

The advantages of using decision table as source language are,

i) It provides integrity of clarity.

ii) It provides explicit relation to specification.

iii) It provides high-level of maintainability.

The drawback of using decision table as source language is that it provides inefficient program logic.

Decision Table Based Testing :-

Decision tables are used for designing test-case when specification are specified as decision table. On the other hand, decision tables are used as a basis for designing test-case when program's logic is implemented as decision table. In this situation the reliability and completeness of decision table are examined using decision table processors. The processor automatically converts the decision table into specialized language

where a translator is responsible for checking reliability and specifying the default rules if required.

The following are the situations where decision tables are restricted for designing test-case.

1) When specifications are specified or converted as decision table.

2) When the particular sequence in which conditions are estimated has no influence on the resulting action or definition.

3) When the particular sequence in which rules are estimated has no influence on resulting action.

4) If a test-case is true and leads to the execution of many actions, ~~is executed~~ the sequence in which actions are performed has no importance.

5) When a specific test case is true and its respective action is executed, it is not necessary to check the remaining test cases.

Expansion of Immaterial Cases:-

Immaterial entries (I) cause most decision-table contradictions. If a condition's truth value is immaterial in a ~~value~~ rule, satisfying the rule does not depend on the condition. It doesn't mean that the case is impossible. For example,

Rule 1: "If the persons are male and over 30, then they shall receive a 15% raise".

Rule 2: "But if the persons are female, then they shall receive a 10% raise"

The above rules state that age is material for a male's raise, but immaterial for deter-mining a female's raise. No one would suggest that females either under or over 30 are impossible. If there are n predicates there are 2^n cases to consider. Find the cases by expanding the immaterial cases.

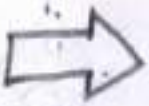
This is done by converting each 'I' entry into a pair of

entries, one with a 'YES' and the other with a 'NO'. Each 'I' entry in a rule doubles the number of cases in the expansion of that rule. Rule 2 in Table 1 contains one 'I' entry and therefore expands into two equivalent subrules. Rule 4 contains two 'I' entries and therefore expands into four subrules. The expansion of rules 2 and 4 are shown in below table:

	RULE 2		RULE 4			
	Rule 2.1	Rule 2.2	Rule 4.1	Rule 4.2	Rule 4.3	Rule 4.4
CONDITION 1	YES	YES	NO	NO	NO	NO
CONDITION 2	YES	NO	YES	YES	NO	NO
CONDITION 3	YES	YES	YES	NO	NO	YES
CONDITION 4	YES	YES	YES	YES	YES	YES

Table: Expansion of Immaterial cases for Rules 2 and 4

	RULE 1	RULE 2
CONDITION1	YES	YES
CONDITION2	I	NO
CONDITION3	YES	I
CONDITION4	NO	NO
ACTION 1	YES	NO
ACTION 2	NO	YES



RULE 1·1	RULE 1·2	RULE 2·1	RULE 2·2
YES	YES	YES	YES
YES	NO	NO	NO
YES	YES	YES	NO
NO	NO	NO	NO
YES	YES	NO	NO
NO	NO	YES	YES

Table: The Expansion of an Inconsistent Specification

Decision Tables and Structure:-

Decision tables can also be used to examine a program's structure. Below figure shows a program segment that consists of a decision tree. These decisions, in various combinations, can lead to actions 1, 2, or 3.

If the decision appears on a path, put in 'YES' or 'NO' as appropriate. If the decision does not appear on the path, put in an 'I'.

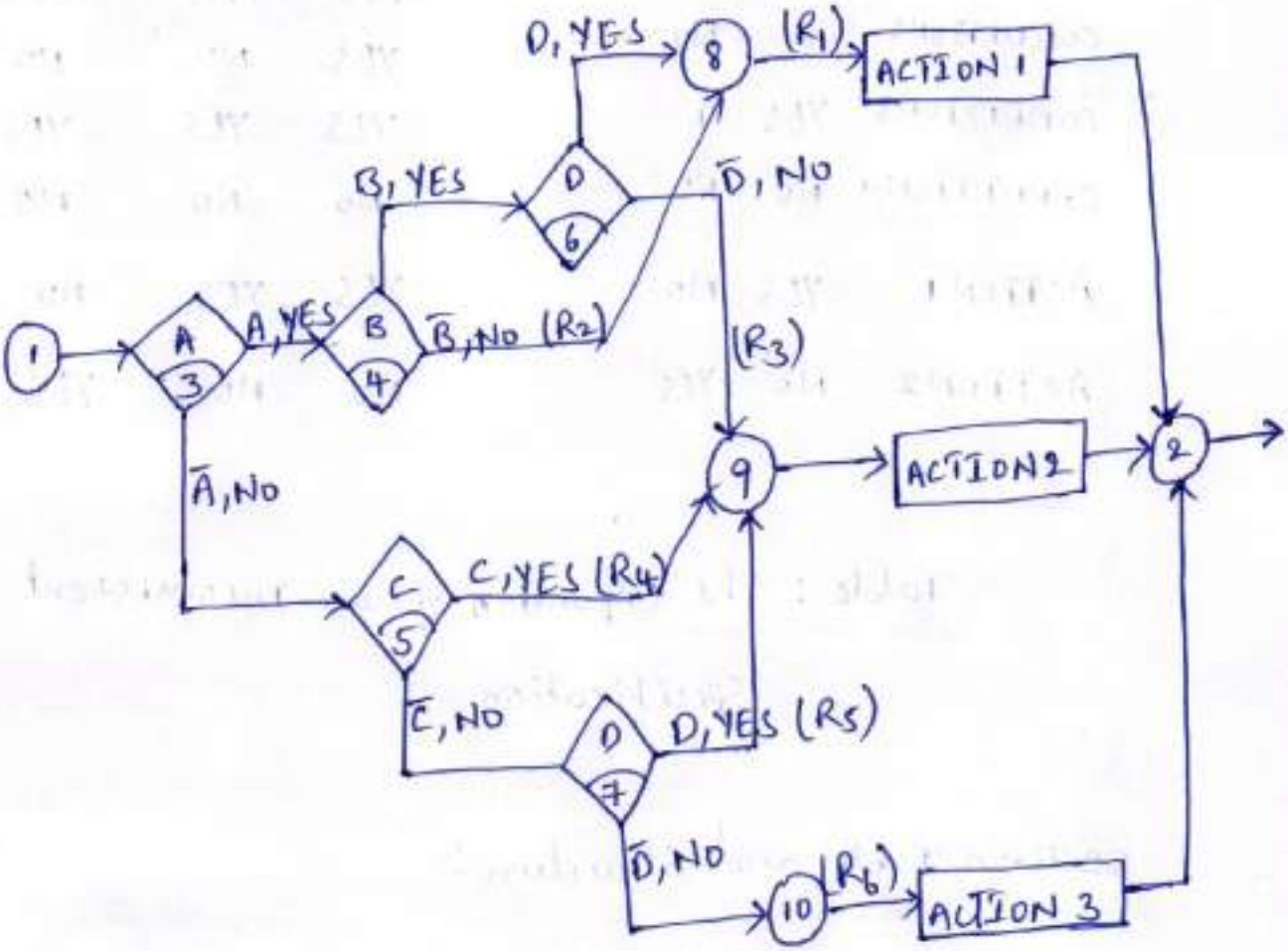


figure: A sample Program.

	RULE 1	RULE 2	RULE 3	RULE 4	RULE 5	RULE 6
CONDITION A	YES	YES	YES	NO	NO	NO
CONDITION B	YES	NO	YES	I	I	I
CONDITION C	I	I	I	YES	NO	NO
CONDITION D	YES	I	NO	I	YES	NO
ACTION 1	YES	YES	NO	NO	NO	NO
ACTION 2	NO	NO	YES	YES	YES	NO
ACTION 3	NO	NO	NO	NO	NO	YES

Table 2: The Decision Table corresponding to above figure.

	R1	RULE 2	R3	RULE 4	R5	R6
CONDITION A	Y Y	Y Y Y Y	Y Y	N N N N	N N	N N
CONDITION B	Y Y	N N N N	Y Y	Y Y N N	N Y	Y N
CONDITION C	Y N	N N Y Y	Y N	Y Y Y Y	N N	N N
CONDITION D	Y Y	Y N N Y	N N	N Y Y N	Y Y	N N

Table : The Expansion of Table 2.

3) Path Expressions:-

i) Predicates and Binary truth values:-

Predicate is defined as a process which gives truth values as its output. Predicates are not confined to only arithmetic relations but also depend on relational operators such as $(>, >=, <, <=, =, \neq)$ is a subset of is a subgraph of, is above, is below.

Predicates is not only implemented using the basic arithmetic relation but also an equality predicate.

This equality predicate is generally used in looping structure that is responsible for scanning the entire set.

ii) Case statements and multivalued logics:-

Predicates are not restricted to binary truth values which are True/false. Predicates can be either be multiway Predicates or multivalued logic.

There are many situations where something can go wrong with predicate.

a) The wrong relational operator is used e.g. > instead of <=

b) The predicate expression of a compound predicate is incorrect ex: A+B instead AB.

c) The wrong operands are used: ex: A > x instead of A > z

d) The processing leading to the predicate (along the predicates interpretation path) is faulty.

Boolean Algebra Rules:-

The different operators that are used while solving any boolean algebra are,

i) Multiplication (x): This Symbol Specifies the AND operation. A statement 'PxQ' is said to be true if both statements P and Q are true, otherwise it is false.

PxQ Truth Table

P	Q	PxQ
T	T	T
T	f	f
f	T	f
f	f	f

ii) Addition (+): This symbol specifies the "OR" operation. A statement 'P+Q' is said to be true if either P is true or both are true.

P+q Truth Table

p	q	p+q
T	T	T
T	f	T
f	T	T
f	f	f

Negation (-): It specifies the complement operation. It is referred as "NOT" operator. It is represented as $\bar{p}$ which means that statement is true if the statement p is false.

$\bar{P}$ Truth Table

p	$\bar{p}$
T	f
f	T

Laws of Boolean Algebra:-

Law1 (Idempotency Law): P+P = P [If both statements are true then the resultant statement is true].

$\bar{P} + \bar{P} = \bar{P}$ [If both statements are false, then the resultant statement is false].

Law 2 (Null Law): $P + 1 = 1$. Here '1' is universal true. According to '+' operation, if one statement is true, then resultant is always true.

Law 3: $P + 0 = P$. Here '0' is a null value.

Law 4 (Commutative Law): $P + Q = Q + P$

Law 5: $A + \bar{A} = 1$. The resultant statement is universally true, if either A (or) its negation is true.

Law 6 (Idempotency Law): $P \times P = P$, $\bar{P} \times \bar{P} = \bar{P}$

Law 7: $P \times 1 = P$ [If a true statement is multiplied with universal true statement, then the resultant statement is also true].

Law 8 (Null Law): $P \times 0 = 0$ [If a true statement is multiplied using null value, then the resultant statement is also a null value].

Law 9 (Commutative law of multiplication):

$P \times q = q \times p$

Law 10: $P \times \bar{P} = 0$. It is not possible for a single statement to be true and false simultaneously.

Law 11: $\bar{\bar{P}} = P$. This Law represents that the negation of negation statement is the statement itself.

Law 12 (Involution): $\bar{0} = 1$

Law 13: $\bar{1} = 0$

Law 14 (Demorgan's law): $\overline{P+q} = \bar{P}\bar{q}$, $\overline{Pq} = \bar{P} + \bar{q}$

Law 15 (Distributive Law): $P(q+r) = Pq + Pr$

Law 16 (Associative Law of multiplication):

$(P \times q)r = P(q \times r)$

Law 17 (Associative Law Addition):

$(P+q)+r = P+(q+r)$

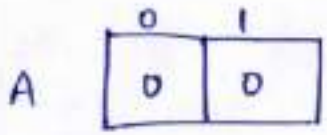
Law 18 (Absorptive Law): $P + \bar{P}q = P + q$

Law 19: $P + Pq = P$

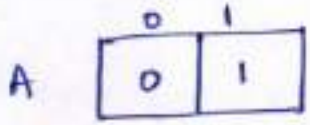
4) KV charts (Karnaugh Veitch)

Simple Forms:- All the boolean functions of a single variable and their equivalent representation as a KV chart. A '1' means the variables value is '1' or TRUE. A '0' means that the variable value is '0' or FALSE.

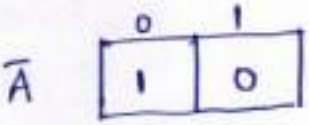
KV chats for functions of a single variable



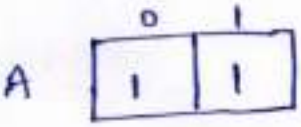
The function is never 'true'



The function is true when A is true

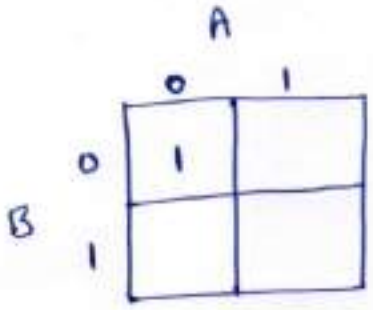


The function is true when A is false

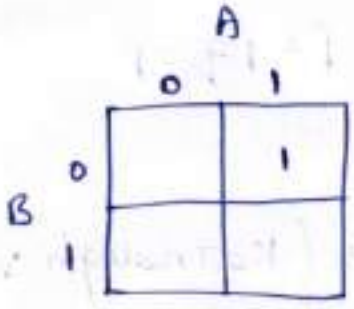


The function is always true

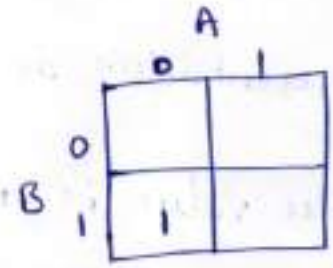
Functions of two variables



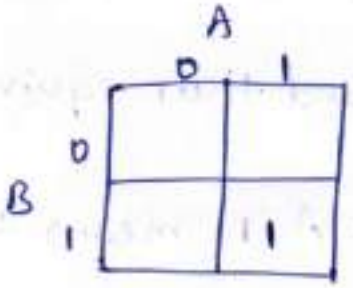
$\bar{A}\bar{B}$ - NAND



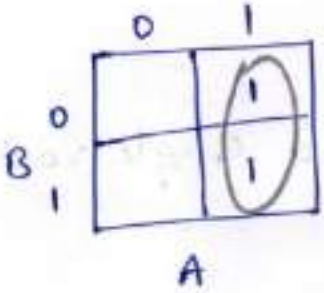
$A\bar{B}$ - A and not B

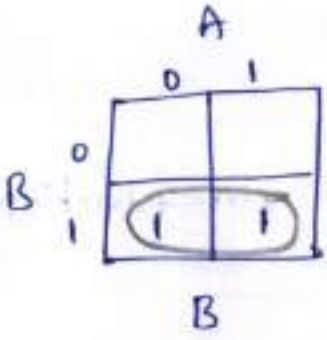


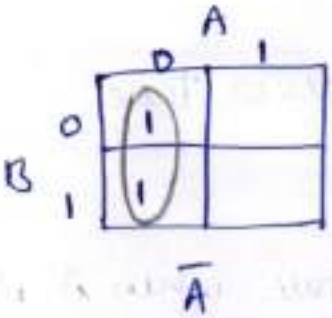
$\bar{A}B$ - B and not A

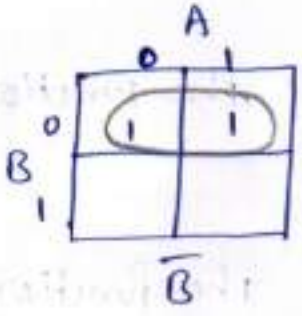


AB - A and B

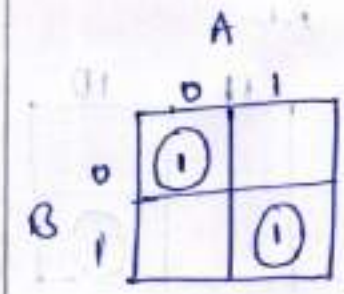




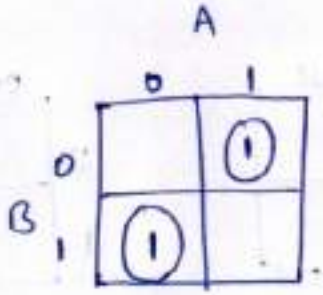




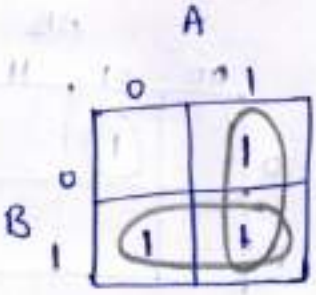
More functions of two variables



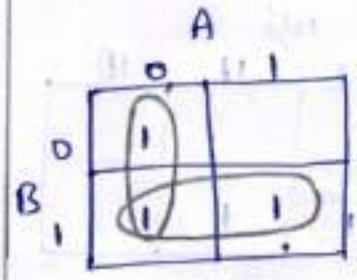
$\bar{A}\bar{B} + AB$



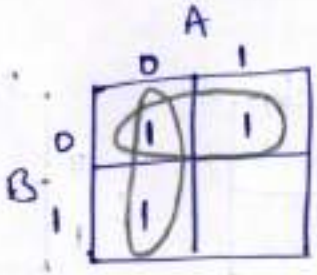
$A\bar{B} + \bar{A}B$



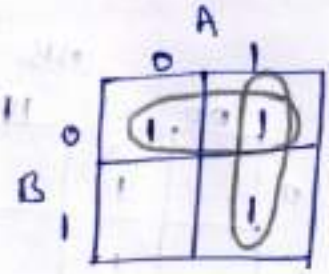
$A + B$



$\bar{A} + B$

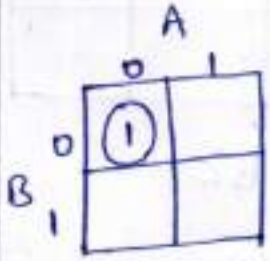


$\bar{A} + \bar{B}$

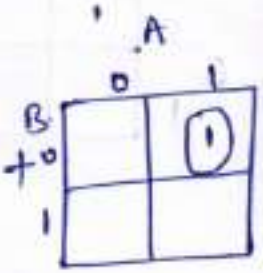


$\bar{B} + A$

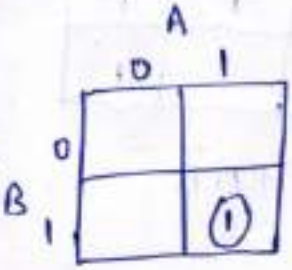
Ex:- $\bar{A}\bar{B} + A\bar{B} + AB = \bar{B} + A$



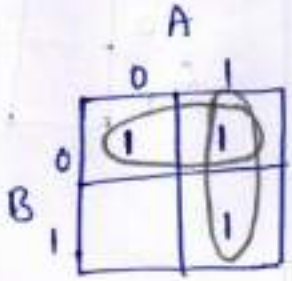
$\bar{A}\bar{B}$ +



$A\bar{B}$ +

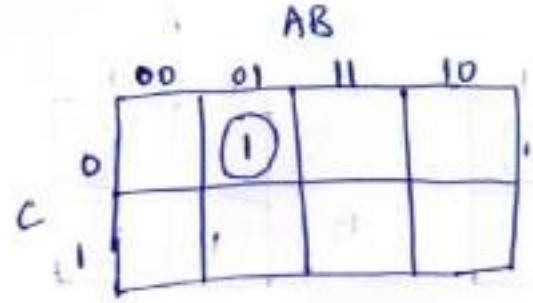


AB =

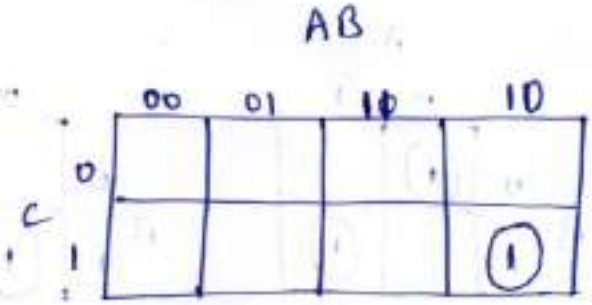


$\bar{B} + A$

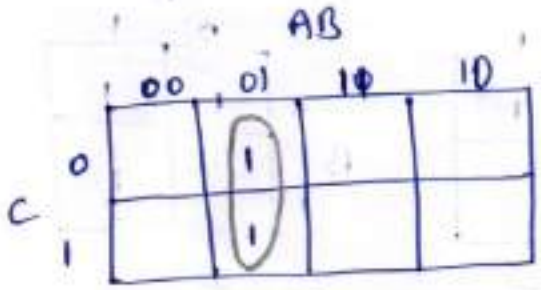
Three Variables:-



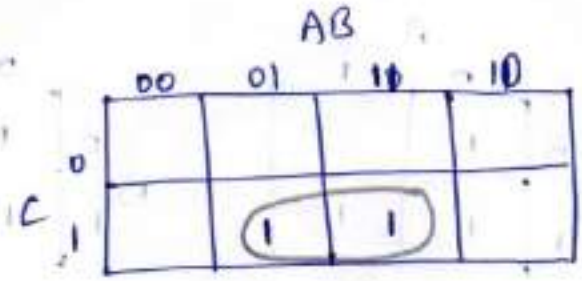
$\bar{A}B\bar{C}$



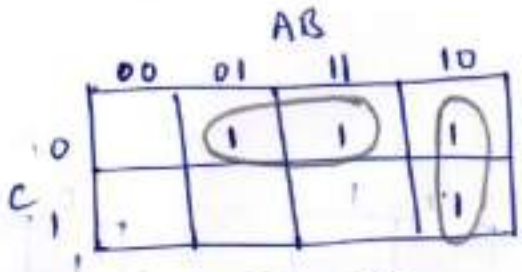
$A\bar{B}C$



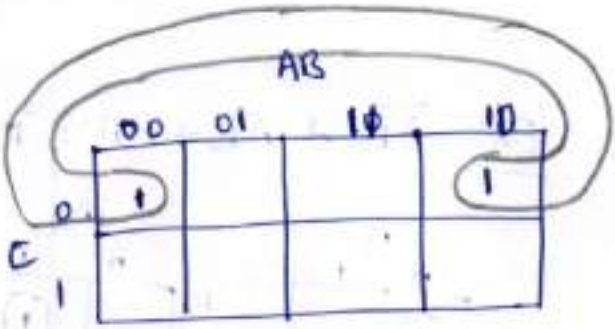
$\bar{A}B$



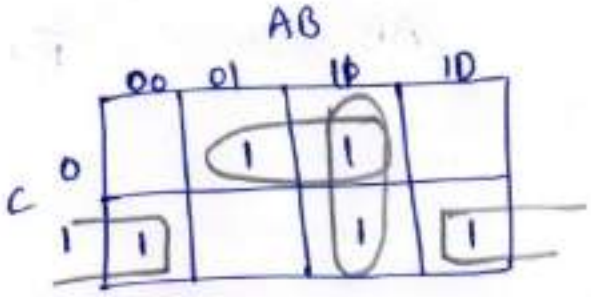
BC



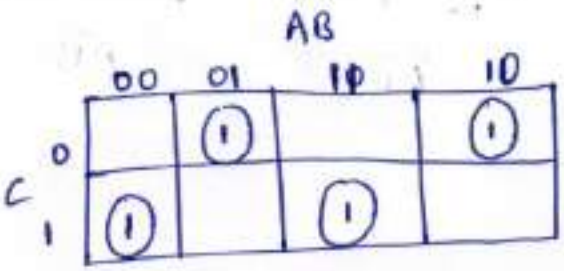
$B\bar{C} + A\bar{B}$



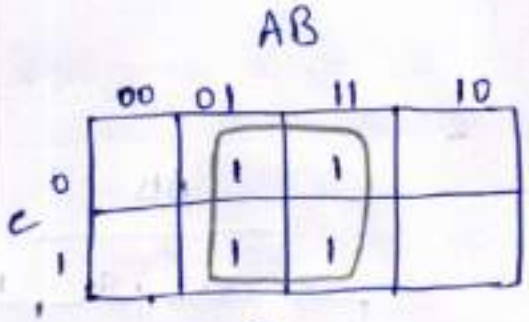
$\bar{B}\bar{C}$



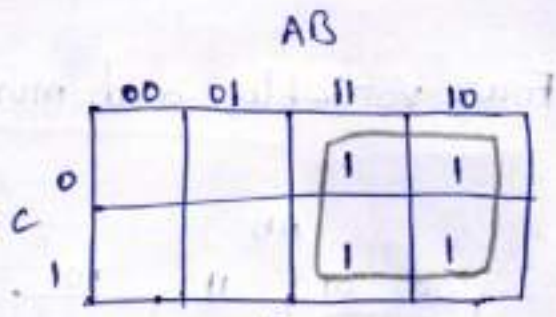
$B\bar{C} + AB + \bar{B}C$



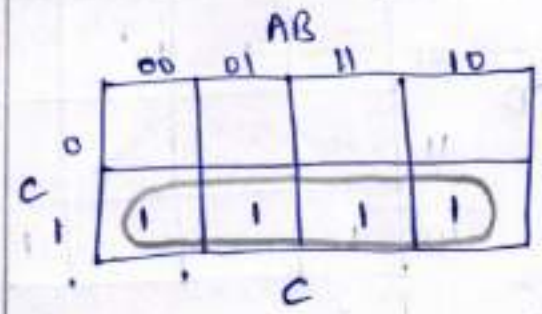
$\bar{A}\bar{B}C + \bar{A}B\bar{C} + ABC + A\bar{B}\bar{C}$



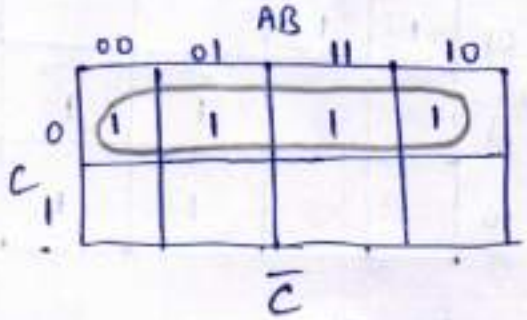
B



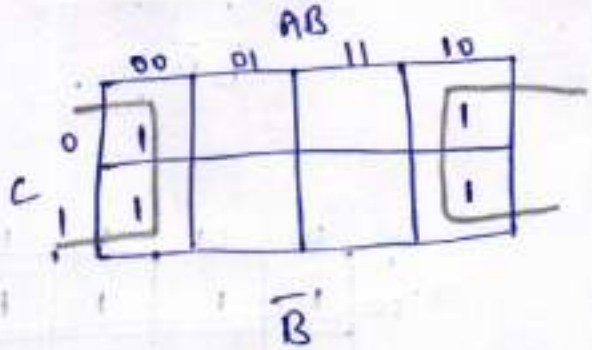
A



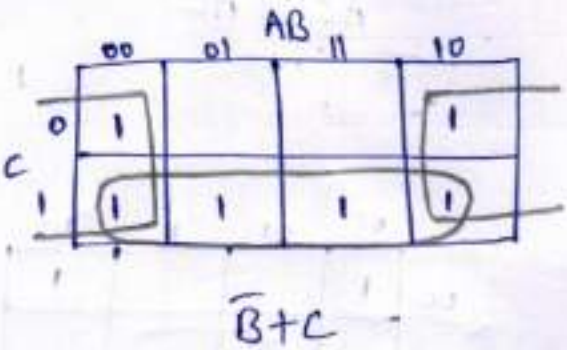
C



$\bar{C}$

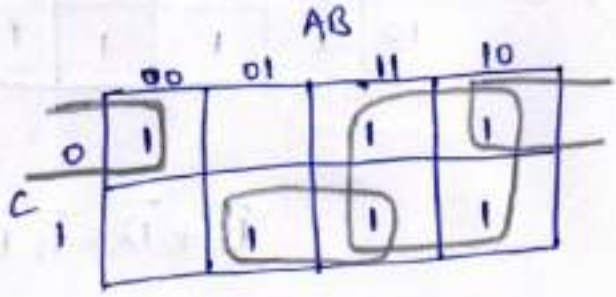


$\bar{B}$

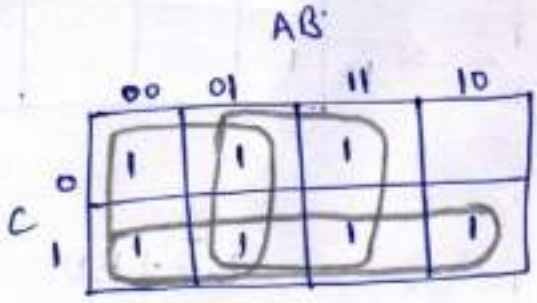


$\bar{B}+C$

$\bar{A}\bar{B} + A\bar{B}\,(\bar{A}+A)\bar{B}$

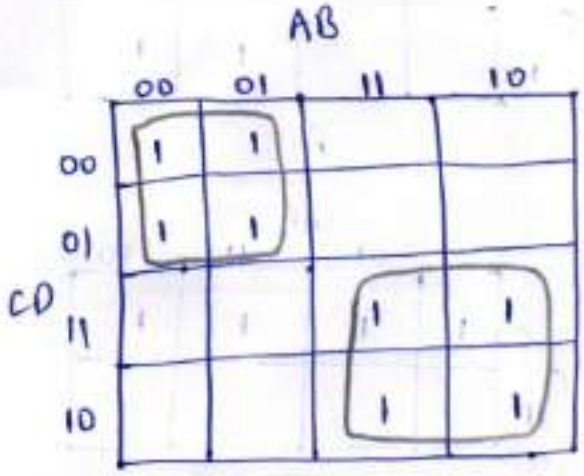


$A+BC+\bar{B}\bar{C}$

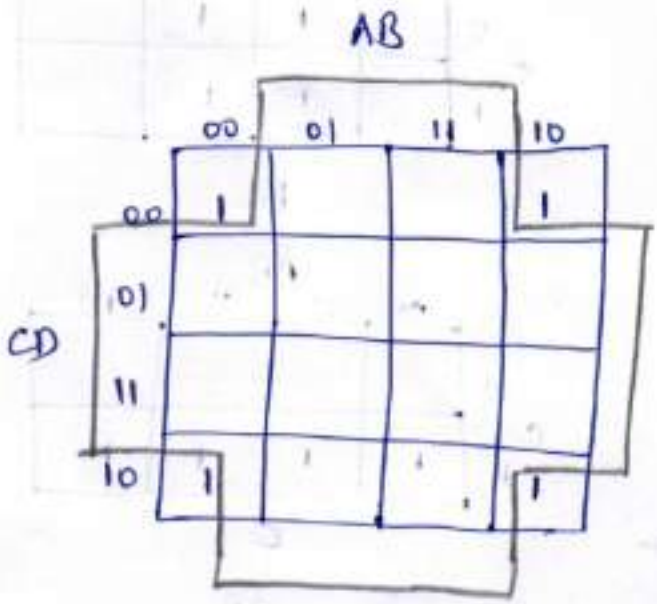


$\bar{A}+B+C$

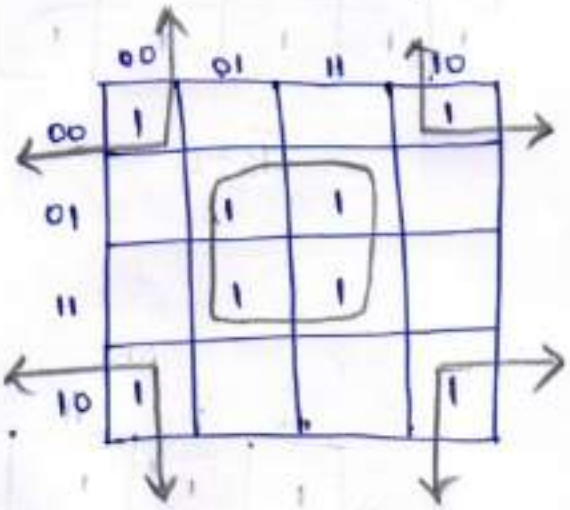
Four Variables and more:



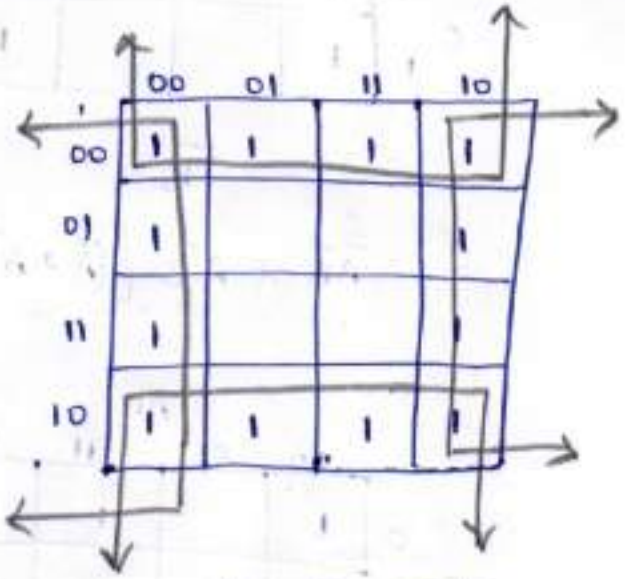
$\bar{A}\bar{C} + AC$



$\bar{B}\bar{D}$



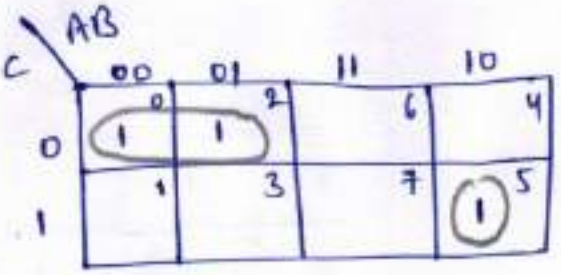
$\bar{B}\bar{D} + BD$



$\bar{B} + \bar{D} = \bar{B}\,\bar{D}$

Ex:- 1) Simplify the given function using K-maps

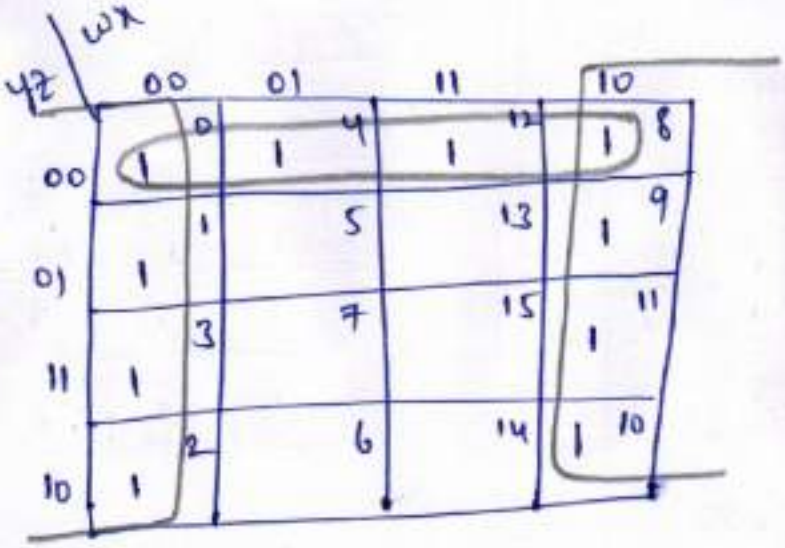
$f(A,B,C) = S(0,2,5)$



$F = \bar{A}\bar{C} + A\bar{B}C$

2) Find the simplified expression

$f(w,x,y,z) = \Sigma m(0,1,2,3,4,8,9,10,11,12)$



$F = \bar{w}\bar{x} + w\bar{x} + \bar{y}\bar{z}$ $(\because (w+\bar{w}) = 1)$

$= \bar{x}(w+\bar{w}) + \bar{y}\bar{z}$

$f = \bar{x} + \bar{y}\bar{z}$