Unidad 1: Logica Proposicional.

Logica: Es la disciplina de la ciencia que estudia los metodos de razonmaiento.

Falacias Logicas: Son argumentos que parecen ser verdaderos pero que no son. Ejemplo: "Si es Bahiense entonces es Argentino". "No es Bahiense". "No es Argentino".

Proposicion: Es una oracion que puede tener valor de verdad, verdadero o falso.

Conectivos Logicos:

 - ¬ Negacion.

 - ∧ Conjuncion.

 - ∨ Disyuncion.

 - → Implicacion.

 - ↔ Equivalencia.

Sintaxis: La Sintaxis determina si una expresion esta gramaticalmente correcta o no.

Semantica: La Semantica determina el significado de una expresion.

Simbolos:

 - De verdad: V, F

 - Conectivos: ¬, ∧, ∨, →, ↔

 - Variables: P, Q, R, etc.

 - Simbolos Puntuacion: (,)

Formula Bien Formada FBF:

 - Simbolo de Verdad:

 - Letra Proposicional

 - Negacion de una FBF

 - Conjuncion de dos FBF

 - Disyuncion de dos FBF

 - Implicacion de dos FBF

 - FBF Rodeada de Parentesis

Forma Normal de Backus-Naur (BNF): Notacion utilizada para describir lenguajes formales.

Escritura de BNF: <simbolo> ::= <expresion>. Donde <simbolo> es un no terminal y <expresion> una secuencia de simboloes o secuencias separadas por |.

La gramatica para las FBFs: <fbf> ::= V | F | P | Q | R | ¬<fbf> | (<fbf>∧<fbf>) | (<fbf>∨<fbf>) | (<fbf>→<fbf>) | (<fbf>)

Tablas de Verdad: Son utilizadas para mostrar los valores de verdad que toman las fbfs. Toda fbf puede tener asociada una tabla de verdad, pero se deben construir respetando la jerarquia de precedencia de los operadores: ¬, ∧, ∨, →.

Tautologia: Es una fbf que es verdadera para todos los valores de verdad.

Contradiccion: Es una fbf que es falsa para todos los valores de verdad.

Contigencia: Es una fbf que es falsa y verdadera para algunos valores de verdad.

Equivalencia: Dos FBFs son equivalentes si tienen la misma tabla de verdad.

Funciones Booleanes: Son funciones que toman como parametros variables proposicionales.

Forma Normal Disyuntiva: Son FBFs de la forma, disyuncion de conjunciones. Ejemplo: (P∧Q)∨(P∧R)

Forma Normal Conjuntiva: Son FBFs de la forma, conjuncion de disyunciones. Ejemplo: (P∨Q)∧(P∨R)

Forma Normal Disyuntiva Completa: Son FNDs que en cada termino aparecen todas las variables proposicionales.

Forma Normal Conjuntiva Completa: Son FNCs que en cada termino aparecen todas las variables proposicionales.

Conjunto Completo de Conectivos: Un conjunto de conectivos logicos se dice completo, si es posible expresar cualquiera funcion de verdad con ellos. Ejemplo {¬, ∧, ∨}.

Literal: Un literal es una letra proposional o su negacion. Ejemplo: P, ¬P.

Conjuncion Fundamental: Es un literal o una conjuncion de literales. Ejemplo: P, (P∧Q), (P∧Q∧¬R).

Disyuncion Fundamental: Es un literal o una disyuncion de literales. Ejemplo: P, (P∨Q), (P∨Q∨¬R).

Conjetura: Es un argumento apoyado sobre obvservaciones.

Tecnicas para probar si una Conjetura es V o F:

 - Por contrajemplo.

 - Prueba Exhaustiva.

 - Prueba Directa.

 - Contrapositiva.

 - Contradiccion.

 - Induccion.

Por Contraejemplo: Sirven par aprobar que algo es falso.

Prueba Exhaustiva: Sirve para probar que algo es verdadero o falso, cuando se tiene una coleccion finita dee casos.

Prueba Directa: Sirve para probar una conjetura, mediante la forma P → Q, si P es verdadero entonces Q tambien. Se asume la hipotesis P y se deduce Q.

Contrapositiva: Sirve para probar cuando P → Q resulta dificil, emplendo ¬Q → ¬P.

Contradiccion: Estas pruebas se obtienen asumiendo que la conjetura es falsa y mostrando que esa suposicion implica una proipiedad conocida es falsa. Para probar P → Q, asumo P ∧ ¬Q y llego a una contradiccion. Se las conoce como Prueba por el Absurdo.

Regla de Inferencia: Son patrones sintacticos que establecen que a partir de un conjunto de premisas, podemos derivar a una conclusion.

P1

P2

...

PN

---

.:.C

P1 ∧ P2 ∧ ... ∧ PN → C. Cuando esto es cierto es una tautologia.

Las reglas de inferencia deben presevar la verdad.

Modus Ponenes.

A ∧ (A → B) | B.

Modus Tollens.

¬B ∧ (A → B) | ¬A.

Regla de Conjuncion.

A ∧ B | A ∧ B.

Regla de Simplificacion.

A ∧ B | A

Regla de Adicion.

A | A ∨ B

Sislogismo Disyuntivo.

(A ∨ B) ∧ ¬A | B

Sislogismo Hipotetico.

(A → B) ∧ (B → C) | A → C

Axioma: Es una FBF que queremos usar como base a partir de la cual podremos razonar.

Un axioma es usualmente una FBF que conocemos como verdadera.

Cuando la logica es aplicada a cierto tema, un axioma puede ser algo como "queremos que sea verdadero".

En una teoria formal, los axiomas y las reglas de inferencia se combinan para formar pruebas.

Prueba: Es una secuencia finita de FBF con la propiedad de que cada FBF en la secuencia es: un axioma o una otra FBF inferida de la secuencia anterior.

La ultima FBF de la secuencia es llamada Teorema.

Una Teoria Formal es Sensata: Si todo teorema de la teoria es una tautologia.

Una Teoria Formal es Completa: Si todas las tautologias son teoremas.

Una Teoria Formal es Consistente: Si no es posible probar una formula y su negacion.

Unidad 2:

Conjunto: Es una coleccion de elemetos u objectos.

Elemento: es un miembro de un conjunto.

Conjunto vacio: {}

Conjunto con un elemento: {1}

Conjunto con dos elementos: {1,2}

Pertenencia: Notamos que un elemento x pertenece a un conjunto A con: x ∈ A.

Union: A ∪ B = {x | x ∈ A ∪ B}

Interseccion: A ∩ B = {x | x ∈ A ∩ B}

Diferencia: A \ B = {x | x ∈ A \ B}

Complemento: A' = {x | x !∈ A}

Podemos definir un conjunto por extension o por compresion.

Un conjunto que no tiene elementos se dice vacio.

La cardinalidad de un conjunto es el numero de elementos que tiene.

La cardinalidad de un conjunto A es: |A|

Conjunto Potencia: Es el conjunto cuyos elementos son todos los subconjuntos de A.

Si para todo x,y ∈ A, x ° y ∈ A, entonces A se dice cerrado bajo la operacion °.

Si para todo x,y ∈ A, x ° y eciste y es unico, entonces se dice que ° esta bien definida.

La operacion binaria ° es una operacion binaria sobre un conjunto A.

Una relacion binaria sobre A es un subconjunto de A x A.

Tipos de relaciones:

 - Uno a Uno: Si cada primer componente y cada segundo componente aparece solo 1 vez en la relacion.

 - Uno a Muchos: Si el mismo primer component aparece con mas de 1 segundo componente.

 - Muchos a Uno: Si un segundo component aparece con mas de un primer componente.

 - Muchos a Muchos: Si al menos 1 primer componente aparece con mas de 1 segundo componente y al menos 1 segundo component aparece con mas de 1 primer componente.

Propiedades de las Relaciones Binarias:

 - Reflexiva: Para todo x(x ∈ A -> (x,x) ∈ P)

 - Simetrica: Para todo x,y(x ∈ ∧ y ∈ A ∧ (x,y) ∈ P -> (y,x) ∈ P)

 - Transitiva: Para todo x,y,z(x ∈ A ∧ y ∈ A ∧ z ∈ A ∧ (x,y) ∈ P ∧ (y,z) ∈ P -> (x,z) ∈ P)

 - Antisimetrica: Para todo x,y(x ∈ A ∧ y ∈ A ∧ (x,y) ∈ P ∧ (y,x) ∈ P -> x = y)

 - Irreflexiva: Para todo x(x ∈ A -> (x,x) !∈ P)

Clausuras de Relaciones: Una relacion binaria P\* sobre el conjunto A es la clausura de una relacion P sobre A respecto a la propiedad T si:

 - P\* tiene la propiedad T.

 - P subconjunto de P\*

 - P\* es un subconjunto de cualquiera otra relacion sobre A que incluye P y la propiedad T.

Grafo: es un conjunto no vacio de nodoes y un conjunto de arcos.

Un grafo es un par ordenado (N,A), donde N = conjunto no vacio de nodos y A = conjunto de arcos.

Grafo Dirigido: Es un par ordenados (N,A) donde N = conjunto no vacio de nodos y A = conjunto de arcos. Donde cada arco es un par ordenado (x,y) donde x es el inicio e y el final.

Grafo Etiquetado: Es un grafo cuyos nodos tienen asociada informacion tal como nombres de ciudades.

Grafo Ponderador: Es un grafo donde cada arco tiene asociado un valor numerico o peso.

Multigrafo: Es un grafo con arcos multiples, es decir 2 nodos pueden estar conectado por mas de 1 arco.

Matriz de Adyacencia: Es una matriz de NxN donde N es la cantidad de nodos y el contenido de la matriz es la relacion de que si un nodo esta conectado con otro nodo.

Dado un Grafo Dirigido G con N nodos y sin arcos paralelos donde N es el conjunto de nodos, podemos definir una relacion binaria P tal que:

 Si n\_i, n\_j ∈ N, entonces n\_iPn\_j <-> existe un arco en G de n\_i a n\_j.

Dada una relacion binara P sobre un conjunto S de n elementos, podemos construir un Grafo Dirigido G.

Dado un grafo podemos utilizar su matriz de adyacencia para calcular caminos de diferentes longitudes.

Si el grafo es ponderador, la matriz de adyacencia va a tener 0 en la diagonal, y los pesos de los arcos donde exista un arco e infinito donde no exista.

Una relacion binaria que cumpla con: Reflexiva, Simetrica y Transitiva se dice de Equivalencia. Ejemplo: Sobre N, tal que x + y es par.

Una particion P de un conjunto A es una coleccion de subconjuntos A\_i ⊇ A, no vacios, disjuntos 2 a 2 y su union da como resultado A.

Sea P una relacion de equivalencia sobre A y a ∈ A, entoncees [a] es el conjunto de elementos de A con los que a se relacion y es llamado clase de equivalencia de a: [a] = {b ∈ A | aPb}

El conjunto cociente A/P de A por una relacion de equivalencia P es el conjunto de las clases de equivalencia segun la relacion P.

Una relacion binaria sobre A que es Reflexiva, Antisimetrica y Transitiva se dice Relacion de Orden Parcial. Ejemplo: En N x <= y.

Orden Estricto: x < y <-> x <=y y x!=y.

Orden No Estricto: x<=y <-> x < y o x=y.

El conjunto arbitrario parcialmente ordenado se denota (S,<=).

Sea (S,<=) un conjunto ordenado. Se dicee que el elemento b ∈ S cubre al elemento a ∈ S si:

 - a < b.

 - No existe ninung elemento x ∈ S tal que a < x < b.

Los diagramas de Hasse son diagramos usado para representar visualmente conjuntos parcialmente ordenados con un numero finito de elementos.

Dos elementos x e y se dicen comparables si x<=y o y<=x.

Dos elementos x e y no comparables se dicen incomparables.

Una cadena es un conjunto parcialmente ordenado donde todo par de elementos es comparable.

Una anticadena es un conjunto parcialmente ordenado donde todo par de elementos no es comparable.

Un elemento m de un conjunto ordenado (S,<=) se dice elemento maximalde S si:

 - Si x ∈ S tal que m<=x entonces x=m.

Puedne haber 0, 1 o mas elementos maximales.

Un elemento minimal de S se dice elemento minimalde S si:

 - Si x ∈ S es tal que x<=m entonces x = m.

Puedne haber 0, 1 o mas elementos minimales.

Sea (S,<=) un conjunto ordenado. Se dice que un elemento 0 ∈ S es primer elemento de S si:

 - 0 <= x para todo x ∈ S. Es decir todos los elementos por encima de 0 estan conectados con 0.

Sea (S,<=) un conjunto ordenado. Se dice que un elemento 1 ∈ S es el ultimo elemento de S si:

 - x <= 1 para todo x ∈ S. Es decir todos los elementos por debajo de 1 estan conectados con 1.

Sea (S,<=) un conjunto ordenado. Dados a,b ∈ S de dice que el elemento c ∈ S es el infimo de a y b y se nota c = a ∧ b si :

 - c <= a y c<=b.

Sea (S,<=) un conjunto ordenado. Dados a,b ∈ S de dice que el elemento c ∈ S es el supremo de a y b y se nota c = a ∧ b si :

 - c >= a y c>=b.

Todo conjunto ordenado finito tiene por lo menos un elemento minimal(maximal).

Si un conjunto ordenado tiene primer(ultimo) elemento este es unico.

En un conjunto ordenado si existe el infimo(supremo) de 2 elementos, es unico.

Un conjunto ordenado (S,<=) se dice reticulado inferior si todo par de elementos tiene infimo.

Un conjunto ordenado (S,<=) se dice reticulado superior si todo par de elementos tiene supremo.

Un conjunto ordenado (S,<=) se dice reticulado si es reticulado inferior y superior.

El ordenamiento topologico es un proceso para encontrar un orden total K a partir de un orden parcial P sobre un conjunto finito.

Unidad 3: Lenguajes Formales y Automatas Finitos.

Alfabeto: Σ es un conjunto finito no vacio de simbolos.

Cadena: Una cadena sobre Σ es una secuencia finita de simbolos del alfabeto Σ.

Σ\*: Es el conjunto de todas las cadenas posibles sobre Σ.

Lenguaje: Un lenguaje sobre Σ es un subconjunto de Σ\*.

Gramatica: Una gramatica para un lenguajee puede ser descripta definiendo un proceso generativo.

Gramatica estructura de frase: Es una gramatica de tipo 0 definida a traves de una 4 tupla: G=(Vn, Vt, S, P):

 - Vn: Conjunto de simboloes no terminales.

 - Vt: Conjunto de simboloes terminales.

 - S: Simbolo inical de la gramatica.

 - P: Conjunto finito de producciones de la forma a -> b, donde a es una cadena sobre Vn ⋃ Vt con al menos un simbolo de Vn y b es una cadena sobre Vn ⋃ Vt.

Decimos que w1 deriva directamente w2 y notamos w1 => w2 si:

 - a -> b es una producion de P.

 - w1 contiene una instancia de a.

 - w2 es obtenida a partir de w1, reemplazando esa instancia de a con b.

Dada una gramatica G, el lenguaje L generado por G denotado L(G) es el conjunto L = {w ∈ Vt\* | S=>+ w}.

Tipos de gramaticas:

 - Tipo 0: Estructura de Frase. Sin restriccionees.

 - Tipo 1: Sensibles al contexto. Para cada produccion a -> b. |a| <= |b|. Expecto la producion S -> λ.

 - Tipo 2: Libre de Contexto. Para cada produccion a -> b. a ∈ Vn y |a| <= |b|. Expecto la de S -> λ.

 - Tipo 3: Regualar. Para cada produccion a -> b, a ∈ Vn y b es de la forma t o tW, donde t ∈ Vt y W ∈ Vn. Expecto S->λ.

Un lenguaje se dice de tipo N si puede ser generado por una gramatica de tipo N.

El dispositivo computacional mas general se llama maquina de Turing.

Los lenguaes reconocidos por maquinas de Turing son los lenguajes de tipo 0.

Operaciones entre lenguajes.

Sean L1 y L2 dos lenguajes sobre un alfabeto Σ.

 - Union: L1 ∪ L2 = {w | w ∈ L1 ∨ w ∈ L2}.

 - Interseccion: L1 ∩ L2 = {w | w ∈ L1 ∧ w ∈ L2}.

 - Complemento: L1' = {w | w ∈ Σ\* y w ∉ L1}.

 - Estrella de Kleen: L1\* = {w1w2w3...wn | wi ∈ L1, n>=0}.

Las gramaticas regulares sirven para generar:

 - Cualquier lenguaje finito.

 - Lenguajes finitos que presentan ciertas regualaridades que pueden ser expresadas de manera sencilla.

Las gramaticas libres de contexto tienen ciertas caracteristicas:

 - Son sencillas, reemplazan un simbolo a la vez.

Automatas finito: Un automata finito es un modelo que captura las caracteristicas de una computadora.

Tipos de AF:

 - Automates Finitos Reconocedores: Son capaces de reconocer lengaujes regualres.

 - Automatas Finitos Traductores: Toman una entrada y la traducen en una salida.

Tipos de Autoamtas Finitos Traductores:

 - Moore: Las salidas van asociadas a los estados.

 - Mealy: Las salidas van asociadas a las transiciones.

Un automata finito reconocedor es una quintupla:

 M=(S,Σ,δ,q0,F):

 - S: Conjunto de estados.

 - Σ: Conjunto de simbolos.

 - δ: Funcion de transiciones.

 - q0: Estado inicial.

 - F: Conjunto de estados finales.

Un lenguaje reconocido por un AFR se denota L(M) y se define como:

 L(M) = {m | δ\*(s0, w) e F}

Un automata finito traductor de Moore se define como una sextupla:

 M=(S,Σ,δ,q0,F,q):

 - S: Conjunto de estados.

 - Σ: Conjunto de simbolos.

 - δ: Funcion de transiciones.

 - q0: Estado inicial.

 - F: Conjunto de estados finales.

 - q: Estado de salida.

Un automata finito traductor de Mealy se define como una sextupla:

 M=(S,Σ,δ,q0,F,q):

 - S: Conjunto de estados.

 - Σ: Conjunto de simbolos.

 - δ: Funcion de transiciones.

 - q0: Estado inicial.

 - F: Conjunto de estados finales.

 - q: Estado de salida.

Un automata finito reconocedor no determinsita es una quintupla:

 M = (S,Σ,δ,q0,F), donde el unico cambio esta en la funcion de transicion, basicamente un estado puede tener mas de 1 transiciones con e l mismo simbolo.

Un automata finito reconocedor no determinista con transiciones lambda,

es lo mismo que el AFRND pero es posible crear transiciones lambda.

El proceso para minimizar un AFR:

 - Se remueven los estados inalcanzables.

 - Se realizan los k equivalentes.

Expresion Regular: Una expresion regular sobre una alfabeto Σ son cadenas del alfabeto A = Σ U {(,),\*,+,kleen, λ, vacio}.

λ Es una Expresion Regular.

Vacio Es una Expresion Regular.

Todo simbolo a ∈ Σ es una Expresion Regular.

Si a y b son expresiones regualres entonces, (a), a\*b, a +b, akkleen son expresiones regualres.

Teorema de Kleene: Un lenguaje L puede ser expresado por una expresion regualar si y solo si es reconocido por un AF.

Unidad 4: Modelos de Computacion.

Primer principio de induccion.

Sea P(n) una propiedad definida para el entero n y supongamos:

 1) P(1) CASO BASE.

 2) P(n) para todo entero n. HIPOTESIS INDUCTIVA.

 3) P(n+1) PASO INDUCTIVO.

Los metodos de induccion pueder ser aplicados tanto a numeros naturales, como a estructuras como arboles, matrices.

La propiedad necesaria es que exista un principio de buen ordenamiento: debe existir una nocion de tamaño de tal manera que todos los objectos tengan un tamaño finito y cada conjunto de objetos deber tener un objeto menor.

Arboles binarios. Cada nodo puede tener como maximo 2 hijos.

Se dice que un arbol binario esta lleno si todos los nodos tiene 0 o 2 hijos.

Se dice que un arbol binario esta completo: si cada nivel del arbol binario esta completamente lleno, excepto posiblement el ultimo nivel.

Se dice que un arbol binario es perfecto si cada nodo tiene 0 o 2 hijos y todas las hojas tiene la misma profundidad o nivel.

Segundo principio de induccion. Sea P(n) una propiedad para el entero n y supongamos:

 1) P(1) CASO BASE.

 2) Si P(r) es verdadera para todo entero positivo r, 1 <=r<=k

 3) P(n+1) PASO INDUCTIVO.

Un algoritmo es correcto si para cualquiera entrada legal termina y produce la salida deseada.

Las pruebas de correctidud no pueden ser automatizadas.

Correctitud parcial:

 Entrada legal => Algoritmo => Salida deseada

 Si este punto Entonces esta es la salida deseada.

 es alcanzado.

Correctitud Total:

 Entrada Legal => Algoritmo => Salida deseada

 Este punto Esta es la salida deseaseada.

 es alcanzado

Para probar la correctituda parcial asociamos una serie ade aserciones a puntos especificos del algoritmo.

Existen Precondiciones: Son aserciones que deben ser validas antes de que se ejecute el algoritmo.

Existen Postcondiciones: Son aserciones que deben ser validas despues de que se ejecute el algoritmo.

La tripla de Hoares: {Q}P{R}. Donde Q es la precondicion del progrma y R la postcondicion.

Ejemplo:

 {Q}

 S0

 {R1}

 S1 Para probar la correctitud de P se debe

 {R2} probar que para toda entrada X:

 ... Q(x) => R(x,P(x))

 Sn-1

 {R}

Para probar la correcitud de un programa que involucra un ciclo, es necesario probar que la invariante de ciclo es valida.

Se prueba que la invariante es valida.

Se prueba que al final el ciclo se verifica.

Lema: Sea n > 1 y sea f: Nn -> Nn definida como sigue: f(x) = (ax+b) mod n.

Entonces f es una biyeccion <-> el mcd(a,n) = 1, es decir si a y n son coprimos. Cuando esto se cumple, existe la inversa y se calcular de la forma.

f^-1(x) = (kx+c) mod n. Donde c es un entero tal f(c) = 0 y k es un enteor tal que 1 = ak + nm, para algun entero m.

Si una funcion es de la forma f(x) = (x+a)mod n, esta se dice funcion de cifrado aditiva, pues lo que hace es toma un valor x y lo cifra agregando un valor fijo a. Es decir corre la posicion.

Si una funcion es d e la forma f(x) = (ax+b) mod n, esta se dice funcion de cifrado afin, pues es mutliplicativa y aditiva al mismo tiempo.

Una funcion no tiene puntos fijos <-> mcd(a-1,n) no divide a b.

Una funcion hash es una funcion que mapea un conjunto S de claves a un conjunto finito de indices 0,1,....,n-1 de una tabla.

Esta tabla se llama tabla de hash.

Que ocurre si tenemos colisiones. Podemos usar la tecnica de sondeo lineal.

Sondeo Lineal: (k+1) mod n, (k+2) mod n, ..., (k+n) mod n

Consiste en tomar la posicion de la colision k e ir sumando de a 1 hasta que encontrmoes una posicion libre.

Sondeo Lineal con Huecos: (k+g) mod n, (k+2g) mod n, ..., (k+ng) mod n

Consiste en tomar la posicion de la colision k e y ir sumando multiplos deg hasta que se encuentre una posicion libre. Lo mas conveniente es tomar un g tal que mcd(g,n) = 1 y 1 <=g <=n.

Funciones Recursivas:

suc(x) = x + 1

suma(m,0) = m

suma(m, n+1) = suc(suma(m,n))

prod(m,0) = 0

prod(m,n+1) = suma(m,prod(m,n))

exp(m,0) = 1

exp(m,n+1) = prod(m, exp(m,n))

fib(0) = 0

fib(1) = 1

fib(n+2) = fib(n+1) + fib(n)

complemento(λ) = λ

complemento(aw) = bcomplemento(w)

complemento(bw) = acomplemento(w)

L = [] es la lista vacia.

L = x::T es la lista L con cabeza x y cola T

longitud([]) = 0

longitud(a::T) = 1 + longitud(T)

Sean A y B conjuntos, si existe una biyeccion entre A y B denotamos que |A| = |B|. En este caso los conjuntos tienen la misma cardinalidad.

Un conjunto se dice contablee si es finito o existe una biyeccion entre el conjunto y los naturales N. Si existe dicha biyeccion el conjunto es infinitamente contable.

De lo contrario es incontable.

Propiedades de conjuntos contables:

 - Todo subconjunto de N es contable.

 - S es contable <-> |S| <= |N|.

 - Cualquiera subconjunto de un conjunto contable ees contable.

 - Cualquier imagen de un conjunto contable es contable.

El conjunto NxN es contable.

La union de conjuntos contables e s contable.

El conjunto de todas las cadenas posibles de un alfabeto finito es contable.

Los numeros racionale son contables.

Podemos utilizar la diagonalizacion para demostrar que un conjunto no es contable.

Unidad 5: Algebra Computacional.

Un algebra es una estructura consistentee de un conjunto no vacio junto con una o mas operaciones definidas sobre ese conjunto.

Ejemplos:

 - [R;+,-,\*]

 - [Q;+,-,\*]

Sea un cojunto A y sea \* una operacion binaria sobre S.

La operacion es asociativa si:

 - ∀(x)∀(y)∀(z)[x\*(y\*z) = (x\*y)\*z]

La operacion es conmutativa si:

 - ∀(x)∀(y)(x\*y=y\*x)

Tiene elemento identidad y es unico si:

 - ∃(i)∀(x)(x\*i=i\*x=x)

Si tiene elemento identidad i, entonces se dice que cada elemento de A tiene un inverso con respecto a i si:

 - ∀(x)∃(x^-1)(x\*x^-1=x^-1\*x=1)

Se dice que [A,\*] es un grupo si A es un conjunto no vacio y \* es una operacion binaria bien definidia sobre A tal que:

 - \* es asociativa.

 - ∃ un elemento identidad en A.

 - Cada elemento en A tiene inverso en A con respecto a \*.

Un grupo en el cual se cumple que la operacion binaria es conmutativa, se llama grupo abeliano.

Se dice que [A,\*] es un monoide si A es un conjunto no vacio y \* es una operacion binaria sobre A tal que:

 - \* es asociativa.

 - ∃ un elemento identidad en A.

Se dice que [A,\*] es un semigrupo si A es un conjunto no vacio y \* es una operacion binaria sobre A tal que \* es asociativa.

Un conjunto A con una operacion binaria satisface la ley de cancelacion a derecha si:

 - ∀(x)∀(y)∀(z)[x\*z = y\*z -> x =y]

Un conjunto A con una operacion binaria satisface la ley de cancelacion a izquierda si:

 - ∀(x)∀(y)∀(z)[z\*x = z\*y -> x =y]

Un anillo es un algebra [A;+,\*] donde:

 - [A;+] es un grupo conmutativo.

 - [A;\*] es un monide y la operacion \* es distributiva a izquierda y derecha sobre +.

Ejemplos: [Z;+,\*]

Un cuerpo es un anillo [A;+,\*] donde ademas se satisface que [A-{0};\*] es un grupo abeliano, donde 0 es la identidad para [A;+]. Ejemplo: [Q;+,\*]

Sea [G;\*] un grupo y A ⊆ G. Entonces [A;\*] es un subgrupo de [G;\*] si [A;\*] es un grupo.

Sea [G;\*] un grupo con identidad i y A ⊆ G; [A;\*] es un subgrupo de [G;\*] si:

 - A es cerrado bajo \*.

 - i ∈ A.

 - Todo x ∈ A tiene inverso en A.

Sean [S;\*] y [T;+] estructuras algebraicas.

Un mapeo f: S->T es un homomorfismo de [S;\*] a [T;+] si para todo x,y ∈ S tal que f(x\*y) = f(x) + f(y)

Sean [S;\*] y [T;+] estructuras algebraicas.

Un mapeo f: S->T es un isomorfismo de [S;\*] a [T;+] si:

 - para todo x,y ∈ S f(x\*y) = f(x) + f(y)

 - la funcion f es una biyeccion.

Un algebra de boolee es un conjunto B sobre el cual estan definidas dos operaciones binarias + y \*, una operacion unaria ' y se distinguen 2 elementos 0 y 1 y se verifican las siguientes propiedades.

x+y = y+x x\*y=y\*x conmutativa

(x+y)+z = x+(y+z) (x\*y)\*z=x\*(y\*z) asociativa

x+(y\*z) = (x+y)\*(x+z) x\*(y+z)=(x\*y)+(x\*z) distributiva

x+0 = x x\*1 = x identidad

x+x'=1 x\*x'=0 complemento

Denotamos una algebra de Boole [B;+,\*,',0,1]

Una compuerta logica es un dispositivo electronico que es la expresion fisica dee un operador booleana.