# 1 Introducción

1. Programa
2. Programación
3. Lenguaje de programación
   1. Bajo nivel
      1. Binario: opcode, direccion
      2. Ensamblador, ensamblador
   2. Alto nivel
      1. Compilado, compilador
      2. Interpretado, interprete
      3. Bytecodes, máquina virtual java
4. Fases creación programa
   1. Editor + seudo código
   2. Programa fuente
   3. Compilador
   4. Programa objeto
   5. Ligador
   6. Ejecutable
5. Lenguaje orientado a procedimientos o programas estructurados
6. Procedimiento
7. Lenguaje orientado a objetos o lenguaje de simulacion
8. objeto
9. Software aplicación
10. Software sistema
11. Sistema operativo
12. Multiusuario
13. multitarea
14. Ingeniería de software: crear sistemas legibles, eficientes, confiables y mantenibles.
15. Procedimiento de desarrollo de software (ciclo de vida de un software)
    1. Diseño y desarrollo (requerimientos del programa)
       1. Análisis: que debe de hacer (Entradas y Salidas), 10%
       2. Diseño: como lo debe de hacer (algoritmos), 20%
       3. Codificación 20%
          1. Secuencia
          2. Selección
          3. Iteración
          4. Invocación
       4. Pruebas 50%
    2. Documentación
       1. Descripción del programa
       2. Desarrollo y cambios del algoritmo
       3. Listado del programa bien comentado
       4. Muestras de las pruebas efectuadas
       5. Manual de usuario
    3. Mantenimiento
       1. respaldos
16. Algoritmos (diagrama de flujo)
    1. Terminal
    2. Entrada / salida
    3. Proceso
    4. Decisión
    5. Iteración
    6. Proceso predefinido (función)
17. Longitud de palabra
    1. Bit
    2. byte
18. Capacidades de una computadora
    1. Entradas
    2. Salida
    3. Almacenamiento (unidad de Memoria)
    4. Operaciones aritméticas y lógicas (ALU)
    5. Control (unidad de control)

# 2 Programación Orientada a Procedimientos

1. Programa modular
2. Modulo
3. Función
4. Identificadores
5. Palabras clave
6. Función main()
7. Objeto cout
8. Archivos de encabezado #
9. Comentarios
   1. Una línea //
   2. Bloque /\*\*/
10. Tipos de datos
    1. De clase
    2. Primitivos
       1. Enteros int, short int, long int, unsigned int, unsigned short int, unsigned long int
       2. Puntos decimal double, float
       3. Carácter char, unsigned char
       4. Boleanos bool
11. Carácter de escape \n \t
12. Sizeof()
13. Notación exponencial
14. Operaciones aritméticas
    1. +
    2. –
    3. \*
    4. /
    5. %
15. Expresiones
16. Negación -a
17. Precedencia de operadores
    1. – der a izq
    2. \*/% izq a der
    3. + - izq a der
18. Variables, dirección de memoria &variable
19. Instrucciones de declaración
20. Instrucciones de asignación
21. Declaraciones multiples

# 3 Asignación, formateo y entrada interactiva

1. Asignación
2. Coereción
3. Variaciones de asignación (+=, -=, \*=, /=, %=)
4. Acumulación (suma = suma + n)
5. Conteo (i++, ++i, i--,--i)
6. Formato de salida (iomanip)
   1. Setw(n)
   2. Setprecision(n)
   3. Setfill(‘x’)
   4. Scientific
   5. Dec
   6. Hex
   7. oct
   8. Fixed
   9. right
   10. left
   11. flush
7. funciones matematicas (cmath, math.h)
   1. abs(n)
   2. pow(a,n)
   3. sqrt(a)
   4. sin(a) sin(30 \* 3.1416 / 180) Los argumentos se expresan en radianes
   5. cos(a)
   6. tan(a)
   7. log(a)
   8. log10(a)
   9. exp(a)
8. moldes(tipoDeDato(exprecion))
9. entrada de datos (cin>>)
10. validación de entradas del usuario
11. constantes (const tipoDeDato NOMBRE = valor)
12. errores de programación
    1. verificación de escritorio
    2. bug
    3. depuración
    4. depurador

## 4 Estructuras de selección

1. flujo de control (secuenciales, selección, iteración e invocación)
2. estructura if-else
3. expresiones condicionales o relacionales (<,>,>=,<=,==,!=)
4. operadores lógicos
   * and &&
   * or ||
   * not !
5. Presedencia
   * !,-,++,-- derecha a izquierda
   * \*,/,% izquierda a derecha
   * +.- izquierda a derecha
   * <,<=,>,>= izquierda a derecha
   * ==, != izquierda a derecha
   * && izquierda a derecha
   * || izquierda a derecha
   * =, +=, -=, \*=, /= derecha a izquierda
6. Presicion numérica
   * Abd(operando1-operando2)<0.00000001
7. sintaxis if-else
   * if(condición){
   * bloque verdadero;
   * } else{
   * Bloque falso;
   * }
8. alcance de un bloque
9. instrucciones if anidadas if-else-if
   * if(condición){
   * bloque verdadero;
   * } else if(condicion){
   * Bloque verdadero;
   * } else {
   * Bloque falso
   * }
10. sintaxis swhitch
    * switch(variable){
    * case a: instrucciones; break;
    * case b: instrucciones; break;
    * case c: instrucciones; break;
    * default: instrucciones;
    * }

# 5 instructiones de repeticion

1. instrucciones de repeticion
   * while
   * for
   * do while
2. ciclos de prueba preliminar: while y for
3. ciclos de prueba posterior: do while
4. ciclos de cuenta fija: dar seguimiento al número de repeticiones
5. ciclos de cuenta variable: variable interactiva a cada paso del ciclo
6. ciclo infinito
7. sintaxis while
   * int contador=1;
   * while(condicion){
   * instrucciones;
   * contador++
   * }
8. ciclos while interactivos
9. centinela
10. instrucciones break y continue
11. instrucción nula;
12. sintaxis for()
    * for(lista de inicialización; expresión; lista de alteración){
    * instrucciones;
    * }
13. Tecnicas de programación con ciclos
    * Entrada interactiva (cin)
    * Selección dentro de un ciclo (if)
    * Evaluaciones de funciones de una variable f(x)=
    * Control interactivo de un ciclo (variables externas de inicialización; finalización; alteración)
14. Ciclos anidados
15. Sintaxis do while
16. Verificación de valides

# 11 Arreglos

1. Variables atómicas
2. Concepto de un arreglo
3. Sintaxis de un arreglo
4. Elementos de un arreglo
5. Índices en un arreglo, variable indexada
6. Secuenciación de un arreglo estructura for
7. Entrada y salida de valores cout cin
8. Verificación de límites del tamaño de un arreglo
9. Inicialización de arreglos
   1. vector[tamaño];
   2. vector[]={a,b…,x,y,z};