UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

Escuela Superior Tepeji del Río

Amor, Orden y Progreso

Área Académica: Ingeniería Industrial

Asignatura: Computación 1

Profesor(a): Luis Alejandro Duarte Esparza

Periodo: Julio – Diciembre 2011

Asignatura: Computación 1

Abstract

Understanding technical introduction to programming, considering the basic concepts and learning algorithms for the design and construction of the same, so that the student is able to tackle other subjects using the tool as important as programming.

Keywords:

Programa, matriz, arreglos, sentencia, ciclos, programación, diagrama de flujo, seudocódigo, algoritmo.

Desarrollo del Contenido

Programa de Estudio
Licenciatura: En Ingeniería Industrial
Asignatura: Computación 1 Clave: II001CICC1
Carga horaria semanal: Teoría: 2 Practica: 3

OBJETIVO DE LA ASIGNATURA

Comprender las técnicas de iniciación a la programación, considerando los conceptos básicos de algoritmos y el aprendizaje, para el diseño y construcción de los mismos, para que el alumno sea capaz de afrontar otras asignaturas haciendo uso de la herramienta tan importante como lo es la programación.

UNIDAD 1. CONCEPTOS DE PROGRAMACIÓN.

OBJETIVO: ***Comprender la importancia de diseñar correctamente un algoritmo para la solución de un problema.***

1.1.- Definiciones de programa.
1.2.- Etapas de desarrollo de un programa.
1.3.- Algoritmo definición
1.4. Los lenguajes de programación
1.5. Estructuras de control
1.6. Tipos de datos y operaciones primitivas
1.7. Constantes y variables
1.8. Expresiones
1.9. Operaciones de asignación

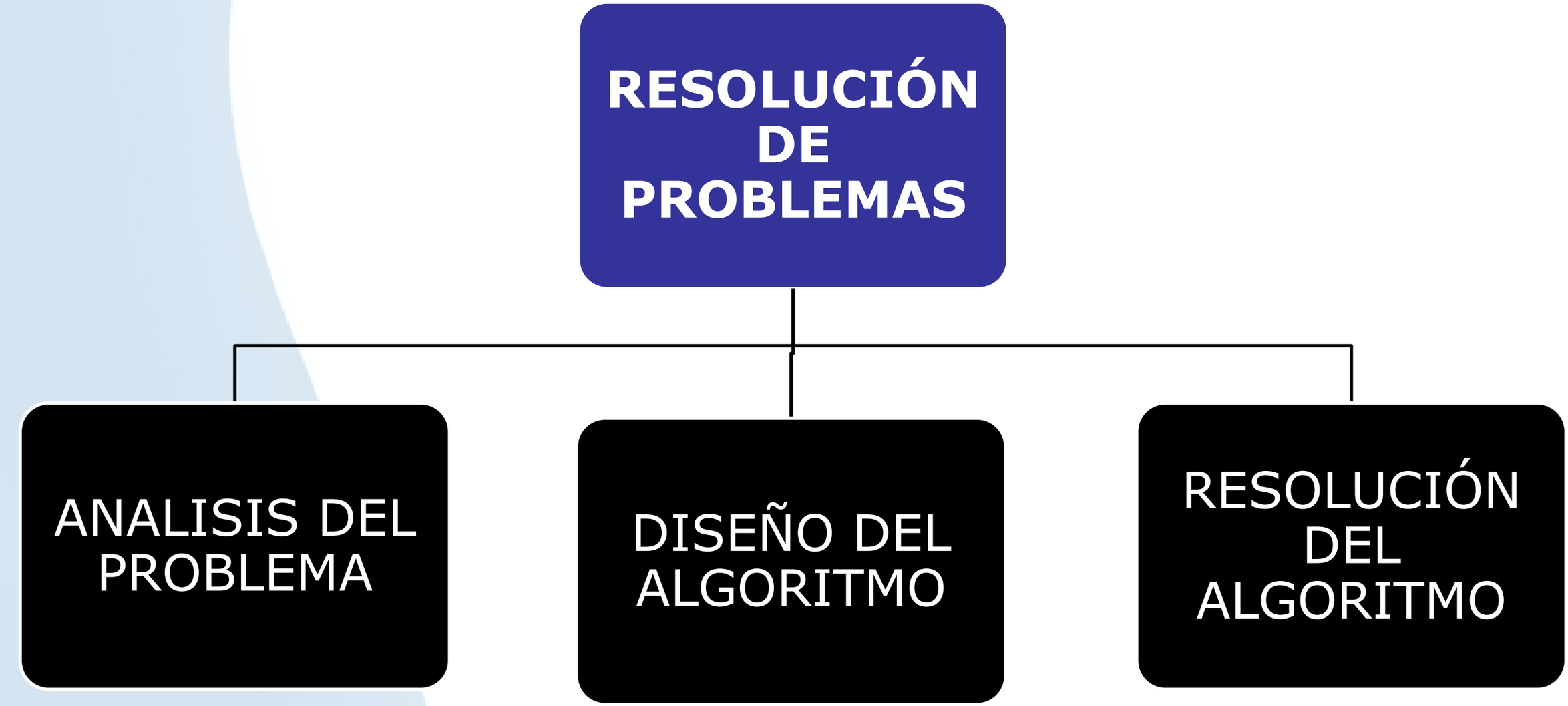
UNIIDAD 2. HERRAMIENTAS PARA LA PROGRMACION POR COMPUTADORAS.

OBJETIVO. ***Resolver problemas analizando y diseñando el algoritmo correcto haciendo uso de herramientas como pseudocódigo, diagramas de flujo y diagramas N-S.***

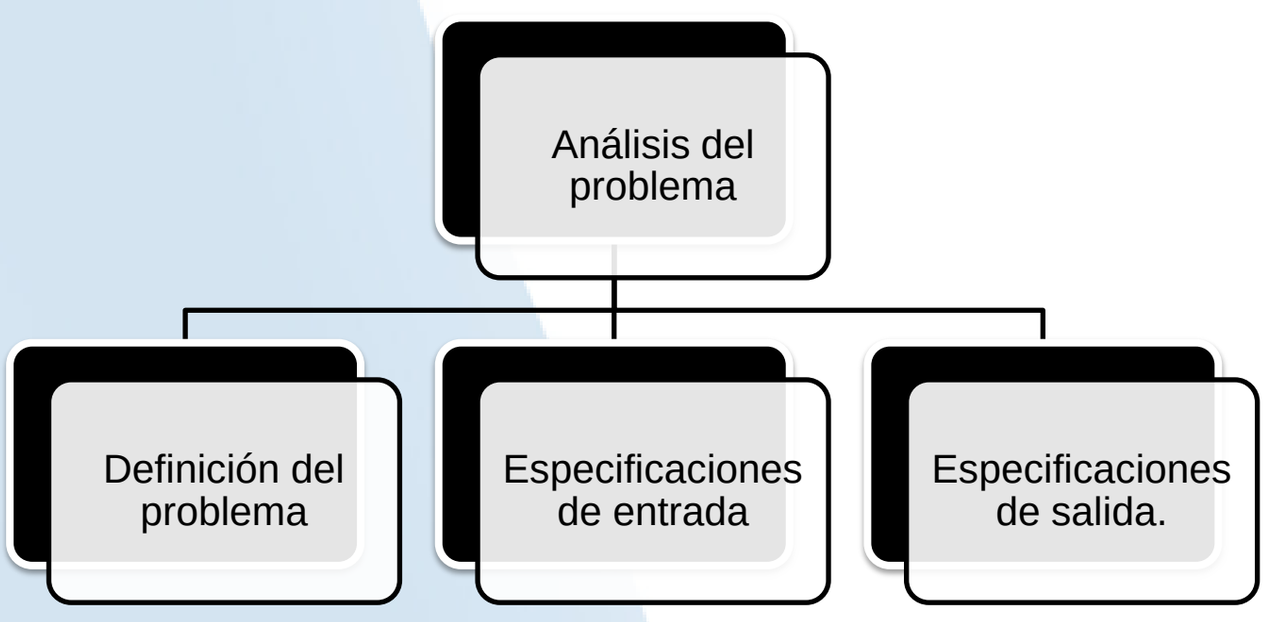
2.1. Análisis del problema.
2.2. Diseño del Algoritmo.
2.3. Representación gráfica de los algoritmos.
2.4. Evaluación de la lógica del diseño del diagrama de flujo
2.5. Pseudocódigo
2.6. Actividades de programación.

LA RESOLUCION DE PROBLEMAS CON COMPUTADORAS Y LAS HERRAMIENTAS DE PROGRMACIÓN

RESOLUCION DE PROBLEMAS



ANALISIS DEL PROBLEMA



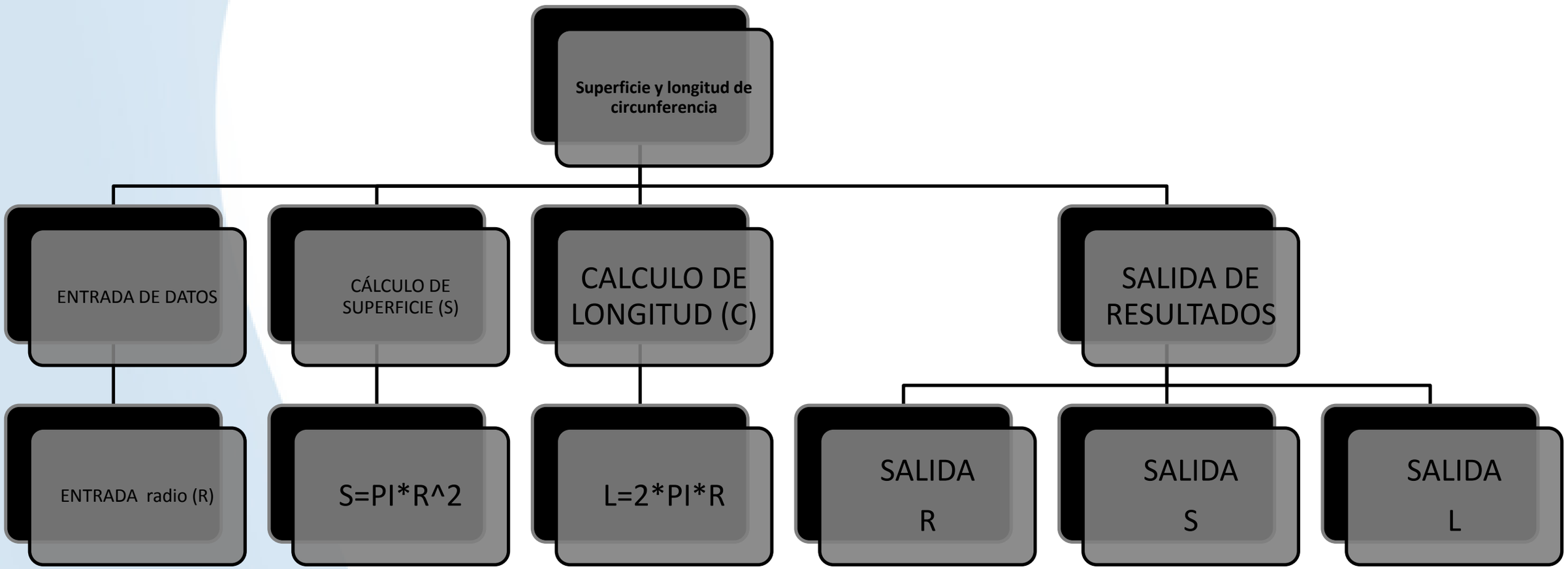
Ejemplo: **Leer el radio de un circulo y calcular e imprimir su superficie y la longitud de la circunferencia.**

ENTRADAS: Radio del circulo (variable RADIO)
SALIDAS: Superficie del circulo (variable AREA)
CIRCUNFERENCIA DEL CIRCULO(Variable CIRCUNFERENCIA)
VARIABLES: RADIO, AREA, CIRCUNFERENCIA (**TIPO REAL)**

DISEÑO DEL ALGORITMO

Diseño de un algoritmo

- Diseño descendente
- Refinamiento por pasos.
- Herramientas de programación (3)
  - Diagrama de flujo
  - pseudocódigo.
  - Diagrama N.S


Superficie y longitud de circunferencia
ENTRADA DE DATOS
CÁLCULO DE SUPERFICIE (S)
CALCULO DE LONGITUD (C)
SALIDA DE RESULTADOS
ENTRADA radio (R)
S=PI*R^2
L=2*PI*R
SALIDA R
SALIDA S
SALIDA L

Ventajas del diseño descendente

- El problema se comprende mas fácilmente al dividirse en partes mas simples denominadas módulos.

- Las modificaciones en los módulos son mas fáciles.

- La comprobación del problema se puede verificar fácilmente.

UNIDAD 3. FUNDAMENTOS PARA LA PROGRAMACIÓN EN TURBO C.

OBJETIVO: Aprender el manejo del editor de Turbo C, al igual que unos aspectos básicos y esencial para el inicio de desarrollo de programas sencillos.

3.1 Introducción a Turbo C.

3.2 El entorno de Turbo C.

3.3 Principios de programación en Turbo C.

Introducción a “C++”

- C++ esta considerado como el lenguaje de programación por excelencia para el desarrollo de aplicaciones orientadas a objetos y además el mas popular.

- C++ combina programación estructurada tradicional y orientada a objetos lo que le convierte en uno de los lenguajes de programación mas eficientes.

Orígenes

- C++ es una extensión del leguaje “c”, y fue creado por Bjarne Stroustrup de AT&T.

- El objetivo de Stroustrup fue mantener la eficiencia de “c” proporcionando la potencia de POO.

- El borrador estándar de C++ se publico en febrero de 1994.

- C es un lenguaje de programación de propósito general que ofrece economía sintáctica, control de flujo, estructuras sencillas y un buen conjunto de operadores.

- No es un lenguaje de muy alto nivel, más bien un lenguaje sencillo. Esto lo hace un lenguaje potente, con un campo de aplicación ilimitado y sobre todo, se aprende rápidamente.

Programación en ++

- La estructura u organización general de un programa c++ se compone esencialmente de directivas de inclusión, datos y funciones globales/externas y un conjunto de funciones.

Datos y funciones globales

```
Main ( )
{
.............
}
```

Funciones definidas por el usuario

Ejemplo de programa

```
#include <stdio.h>---------declara cabecera de entrada
y salida
main()------------------------inicialización del programa
{
printf(“hola mundo de c++”);-----sentencia a ejecutar
}---------------------------------------fin de la función main
```

En la primera línea indica que se tengan en cuenta las funciones y tipos definidos en la librería stdio (standard input/output).

Ahora, en la función main se incluye una única sentencia que llama a la función printf. Esta toma como argumento una cadena de caracteres, que se imprimen van encerradas entre dobles comillas " ".

Bibliotecas de funciones

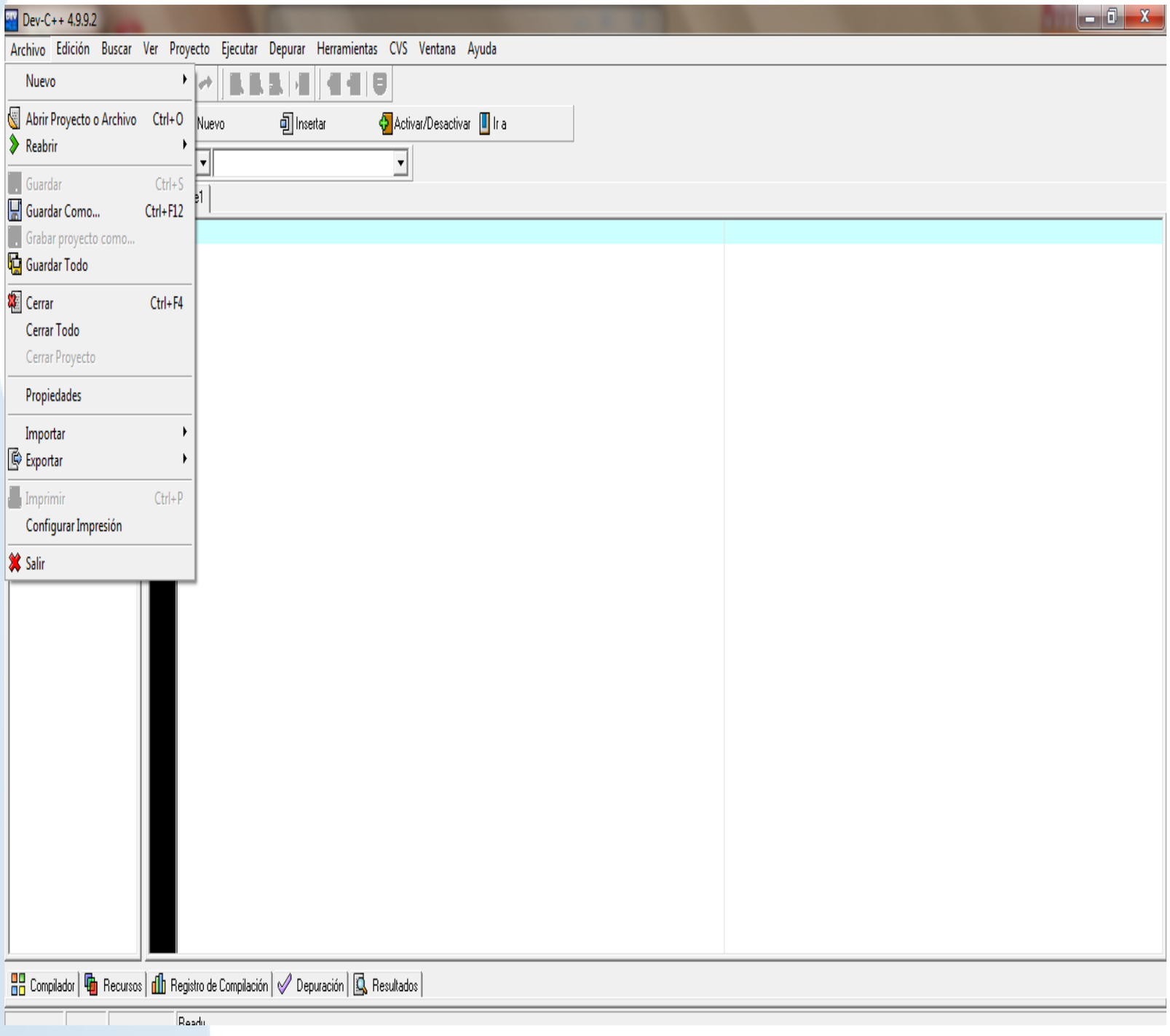
- Las bibliotecas de c++ es un conjunto de mas de 450 funciones y marcos preparados para utilizar y diseñadas para su uso en programas.

- Las bibliotecas hacen la programación mas fácil, ya que incluyen una amplia variedad de tareas, manipulación de cadenas y archivos, asignación de memoria, control de procesos, cálculos matemáticos, etc.

El entorno integrado de desarrollo

- El **EID** contiene un editor, un compilador, un enlazador y un depurador integrados, que permiten desarrollar programas mas fácil y eficiente.

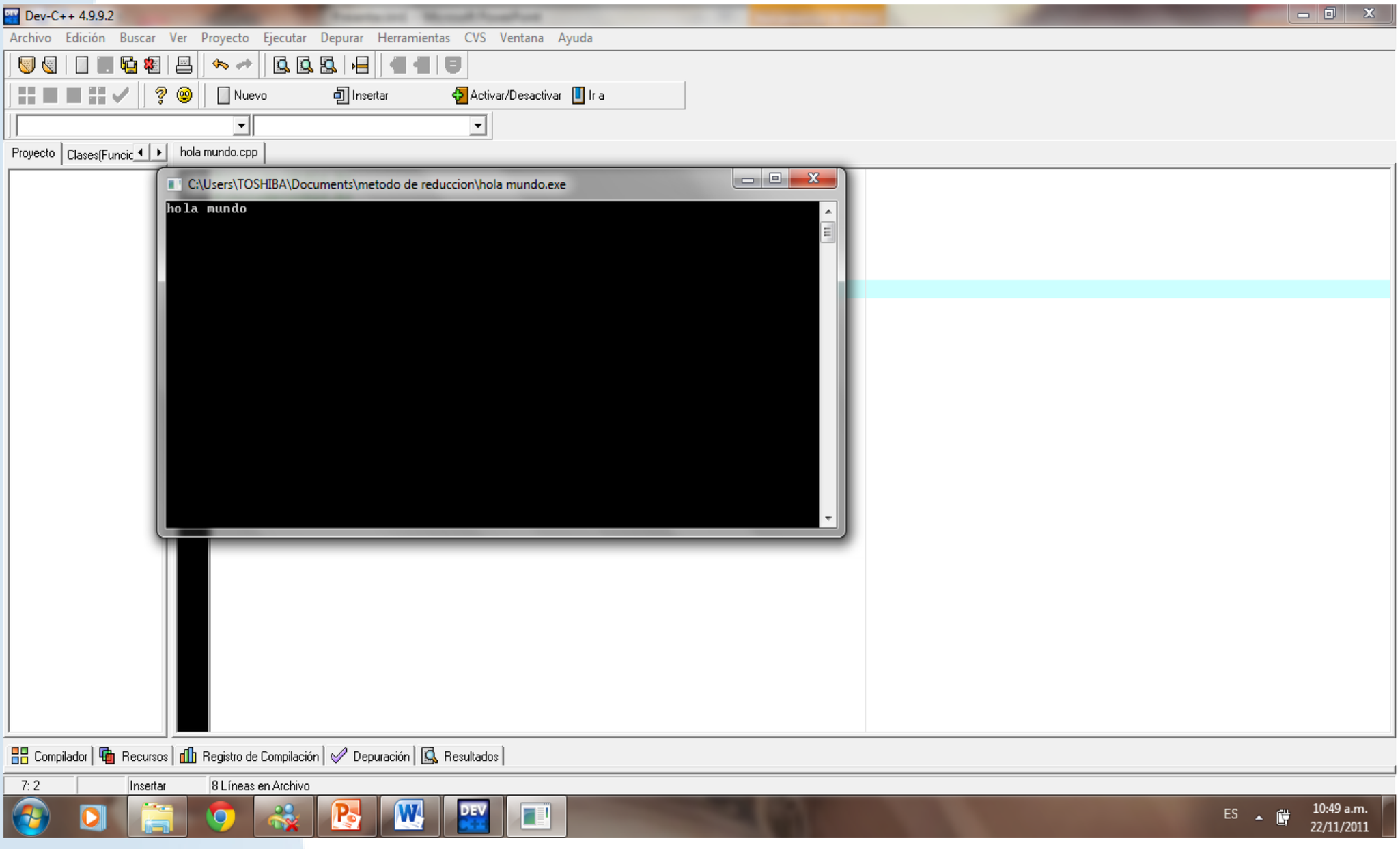
Pantalla principal



- El EID incluye una barra de menús en la línea superior de la pantalla.

- DEV C++ esta organizado en ventanas. Una ventana es un área de la pantalla que puede abrirse, cerrarse, moverse, etc.

Ventana ejecutable



- El EID incluye una barra de menús en la línea superior de la pantalla.

- DEV C++ esta organizado en ventanas. Una ventana es un área de la pantalla que puede abrirse, cerrarse, moverse, etc.

UNIIDAD 4. PROGRAMACIÓN ESTRUCTURADA.

OBJETIVO. Conocer el concepto de programación estructurada, aplicando estructuras clásicas de control en pequeños programas.

4.1. Metodología de la programación estructurada.

4.2. Estructura secuencial

4.3. Estructura de Selección

4.4 Estructura de repetición, cíclica o de bucle.

4.5. Ejercicios

UNIDAD 5. FUNCIONES.

OBJETIVO: Aprender el concepto de subprograma y el medio de comunicación entre ellos para una aplicación posterior en programas más complejos.

5.1. Definición de función

5.2. Declaraciones de funciones

5.3. Invocación de funciones

5.4. Funciones que devuelven valor

5.5. Ejercicios

Funciones de programación

Una función debe ser: declarada, definida y utilizada o invocada

Declaración de una función

La declaración de una función debe estar declarada antes de poder ser utilizada.

Los prototipos de las funciones indican al compilador el nombre y la naturaleza de las funciones que utiliza el programa.

INT f(int i, int j); - declaración de f
void cuenta (int valor);-declaración de cuentas

DEFINICION DE UNA FUNCIÓN

- UNA FUNCION DEBE DE ESTAR DE ESTAR DEFINIDA EN ALGUNA PARTE DEL PROGRAMA.SE TRATA DE SIMPLEMENTE DE INDICAR LA TAREA QUE DEBE HACER.

- SU SINTAXIS ES:

```
tipo nombre  funcion (tipo arg1, arg2,…..)

{

declaraciones sentencias

 }
```

LLAMADA DE UNA FUNCION

LAS FUNCIONES SE UTILIZAN POR INVOCACION O LLAMADA. SU SINTAXIS ES:

nombre(arg1, arg2,…..)

contar(25);

FUNCIONES QUE DEVUELVEN VALORES (SENTENCIA return)

- LA SENTENCIA return SE UTILIZA PARA DOS PROPOSITOS: 1) DEVOLVER EL CONTROL DEL PROGRAMA A LA FUNCION. 2)DEVOLVER UN VALOR DE EXPRECION A LA FUNCION.

1. return;

2. return expresión

return (expresión)

EL TIPO void

- UNA FUNCION NO PUEDE DEVOLVER NINGUN VALOR (ES SIMILAR A UN PROCEDIMIENTO EN PASCAL O UNA SUBRUTINA EN FORMA FORTRAN).

```
//función máximo de tres números
 void max 3 (intX, intY, intZ,)
{
   if  (y>x)  x=y;
  if (z>x)  x=z;
   cout <<“El mayor es :” << x << end l ;
    }
```

ARGUMENTOS POR DEFECTOS

- ES UN PARAMETRO QUE UNA LLAMADA A LA FUNCION NO ES NECESARIO QUE PROPORCIONE. SI SE PASA UN VALOR, SE UTLILIZA; SI NO SE PASA SE UTILIZA UN VALOR POR DEFECTO O ARGUMENTO.

```
void guiones ( int fila, int col, int num, int c='-');
```

puede ser invocada por

```
 guiones (10,0,30);
```

que equivale a

```
  guiones (10,0,30,'-');
```

PASO DE PARAMETROS

- EN C UN ARGUMENTO UN ARGUMENTO SE PASA SIEMPRE POR VALOR , Y NO SE PUEDE MODIFICAR SU VALOR.

```
long factorial  (int numero)
{
long valor = 1 ;
 while (numero > 1)
  valor *= numero --;
   return valor;
   }
factorial (15);
```

PARAMETROS REFERENCIA Y LLAMADA POR REFERENCIA

- LA LLAMADA POR REFERENCIA ESTA DISPONIBLE EN PASCAL, PERO NO EN C. SIN EMBARGO, C++ ADEMAS DEL PASO POR VALOR PERMITE REFERENCIA A DECLARACIONES.

```
tipo&identificador = objetos

int&nn= n;

double&ultimo= a[q];

char&nuevalinea= ‘\n’;
```

VARIABLES TIPO REGISTRO

- LAS VARIABLES REGISTRO SE DECLARAN ANTEPONIENDO A LA DEFINICION DE LA VARIABLE LA PALABRA RESERVADA register.

```
register int j ;
```

FUNCIONES inline

- Una función inline se declara poniendo la palabra la palabra reservada inline al principio de su definición . No tiene prototipo y normalmente se declara antes del main.

```
inline float cil (float rad, float alt)
{ ret ((PI*(rad*rad)*alt);}
 main( )
```

UNIIDAD 6. ARRAYS

OBJETIVO. Aprender el concepto e importancia de arreglos para un fácil manejo de información y evitar la redundancia en programas complejos.

6.1. Arreglos

6.2. Arreglos unidimensionales

6.3. Arreglos bidimensionales

6.4. Ejercicios

Bibliografía

JOYANES, AGUILAR, LUIS RODRÍGUEZ Y MATILDE FERNÁNDEZ AZUELA FUNDAMENTOS DE PROGRAMACIÓN EDITORIAL MC GRAW HILL, MEXICO 1999.