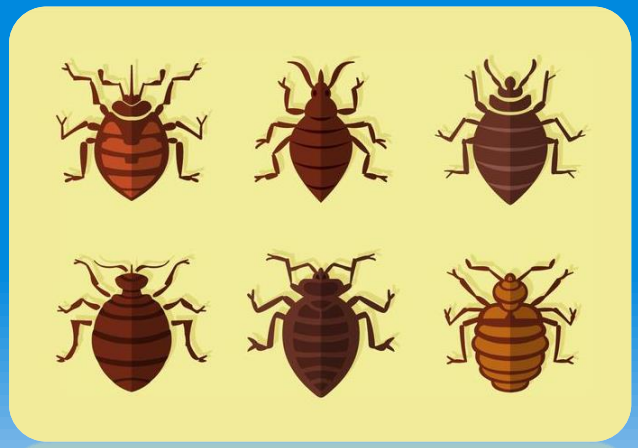
INSTITUTO TECNOLÓGICO BELTRÁN

Centro de Tecnología e Innovación

DEPURACIÓN DE PROGRAMAS

La depuración de programas es el proceso de identificar y corregir errores de programación (debugging). Si alguien busca la definición de bug en un diccionario de inglés descubrirá que significa insecto o bicho, pero para la informática un bug es cualquier error, mal funcionamiento o falla que produce que el software no funcione como se esperaba.

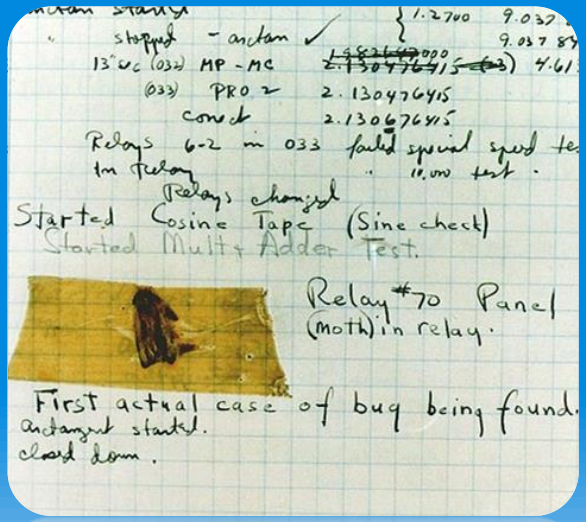
La mayoría de los bugs provienen de los errores cometidos al programar, aunque algunos otros pueden provenir de fallas en el diseño y, los menos, de la conversión que los compiladores hacen del código fuente a código de máquina o la configuración del equipo que se utiliza el ejecutable.



ETIMOLOGÍA

Cuando se construyeron las primeras computadoras, que ocupaban cuartos enteros, la circuitería consistía miles de elementos tales como relés, resistencias, capacitores y tubos de vacío. Los relés son llaves que abren y cierran circuitos eléctricos cuya activación se hace por medio de una corriente que activa un electroimán.

Las resistencias y tubos de vacío producían calor, lo cual atrae a los insectos. Si los insectos morían entre los dos contactos de un relé, el circuito podía cerrarse (o no llegar a hacerlo) lo cual producía resultados inesperados e incorrectos.



Esta polilla es posiblemente el primer bug detectado en una computadora.
Encontrada por Grace Hopper en la Universidad de Harvard, el 9 de Septiembre de 1947.

DIVISIÓN POR CERO

La división por cero no está definida como operación matemática. Es por esto que si intentamos dividir una variable por otra cuyo valor es cero, el programa se cuelga. La ejecución es finalizada automáticamente o, peor aún, continúa con un resultado incierto.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        float iValorA, iValorB;

        printf ("Ingrese el valor de A:");
        scanf("%f", &iValorA);
        printf ("Ingrese el valor de B:");
        scanf("%f", &iValorB); //Valor de B debe ser distinto de 0

        printf ("La división entre A y B es: %f", iValorA/iValorB);

}
```

USO DE VARIABLES SIN INICIALIZAR

No siempre las variables toman un valor inicial, esto depende del lenguaje, del compilador e incluso de opciones del compilador que son modificables por el usuario. Una variable sin inicializar contiene basura, es decir, su valor es cualquier valor, dado por la interpretación de los bytes en memoria que esa variable ocupa.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        int iValorA, iValorB;

        iValorA = 10;
        iValorB = iValorB + iValorA;

        printf ("La suma entre A y B es: %d ", iValorB);

}
```

DESBORDAMIENTO DE VARIABLES NUMÉRICAS

Dependiendo del tipo de variable, por ejemplo las numéricas, se tiene un rango posible de valores. Si utilizamos el tipo byte solo se pueden representar hasta 256 cifras distintas. Si se intenta almacenar un valor mayor se desborda su capacidad y el valor resultante no es el esperado.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        int iValorA, iValorB, iTotal;

        iValorA = 9999999999;
        iValorB = 1;
        iTotal = iValorA + iValorB;
        printf ("La suma entre A y B es: %d ", iTotal);
}
```

PÉRDIDA DE PRECISIÓN EN LA CONVERSIÓN DE TIPOS DE DATOS NUMÉRICOS

Este error sucede generalmente al asignar un valor con parte decimal a una variable entera. El compilador automáticamente efectúa un redondeo (acercar al entero más próximo) o un truncamiento (uso sólo de la parte entera).

NÚMERO ORIGINAL	REDONDEADO	TRUNCADO
5,24	5	5
2.99	3	2
4.5	5	4
7	7	7

INSTRUCCIONES FUERA DE BLOQUE

Por errores de identación en el código puede confundirse qué sentencias están dentro o fuera de un bloque.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        int iValorA, iValorB, iTotal;

        iValorA = 10;
        iValorB = 1;
        iTotal = iValorA + iValorB;

        if (iTotal > 10)
                printf ("Es mayor a 10");
                printf ("Mensaje fuera de la
condición");
}
```

VARIABLES PASADAS POR VALOR EN LUGAR DE PASADAS POR REFERENCIA

Las variables que se pasan como parámetro por valor no son modificadas dentro del procedimiento. Si se deseaba hacer modificaciones debe indicarse que su pasaje se realiza por referencia.

```
#include <stdio.h>
#include <stdlib.h>
void pSuma (int pValorA, int pValorB, int pTotal) {
        pTotal = pValorA+pValorB;
        //El parámetro pTotal no indicado como referencia
}

int main() {
        int iValorA, iValorB, iTotal;

        iValorA = 10;
        iValorB = 1;

        pSuma (iValorA, iValorB, &iTotal);

        printf ("Total: %d", iTotal);
}
```

ASIGNACIÓN EN LUGAR DE PREGUNTAR POR IGUAL

Es un problema típico en C en y los lenguajes que derivan su sintaxis de ellos. Dado que en C el operador de asignación es = y el de comparación por igual es == es común la confusión y el incorrecto funcionamiento del programa. Por dicha razón, es conveniente realizar la proposición de condición comenzando por el valor constante.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        int iValorA;

        iValorA = 10;

        //En vez de: iValorA = 1
        if (1 = iValorA) {
                printf ("Es uno");
        }else{
                printf ("No es uno");
        }
}
```

INSTITUTO TECNOLÓGICO BELTRÁN

Centro de Tecnología e Innovación