Resolución de Problemas y Algoritmos

Clase 10
Resolución de Problemas con secuencias

Grace Murray Hopper

Dr. Diego R. García

Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Bahía Blanca - Argentina

Conceptos: diferentes clases de errores en programas

- **Error de compilación:** es un error detectado por el compilador al momento que se está realizando la compilación de un código fuente, por eso también se llama *error en tiempo de compilación*.
- **Error lógico:** también llamado **error de programación** (*bug*), es un error en la lógica del algoritmo o programa el cual causa que no se resuelva correctamente la tarea que debe hacer el programa.
- **Error de ejecución:** ocurre cuando al momento de la ejecución del programa hay una situación anormal, y generalmente provoca que la ejecución se corte abruptamente. También se lo llama *error en tiempo de ejecución*. Ejemplo: intentar leer de un archivo vacío.

Un profesional debe lograr que los programas no tengan errores.

Para ello *deben probarse* los *programas* (en Inglés *testing*) con los suficientes casos de prueba.

Si se detecta un mal funcionamiento con algún caso de prueba, entonces se deben *buscar y eliminar* los *errores* (*debugging*).

Resolución de Problemas y Algoritmos Dr. Diego R. García

Conceptos: debugging (depuración)

Debugging (depuración): se refiere al proceso metodológico de buscar y reducir el número de errores o defectos (*bugs*) de un programa, con el objetivo de lograr que el programa tenga el comportamiento esperado. http://en.wikipedia.org/wiki/Debugging

Información adicional:

El término "debugging" es atribuido a la Almirante Grace Murray Hopper.

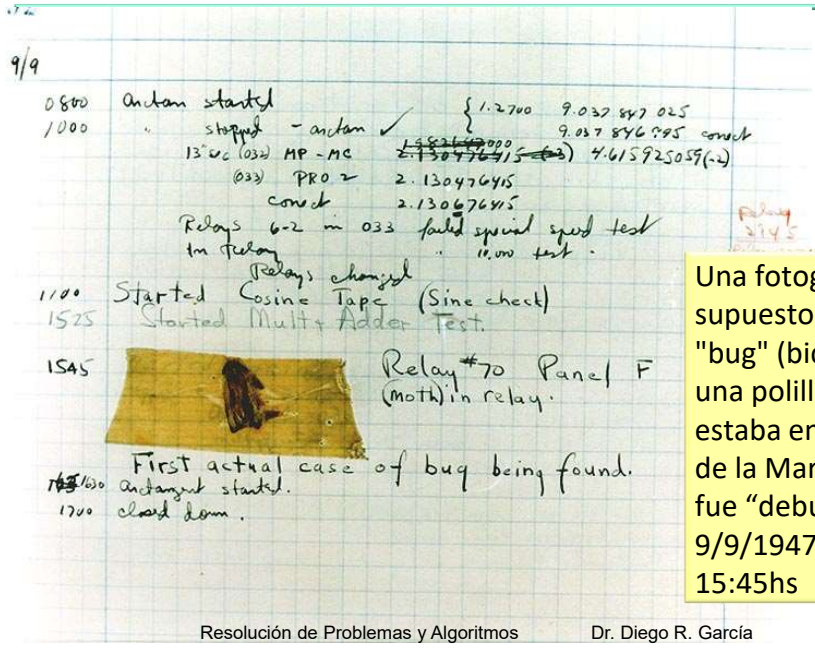
En 1947 mientras trabajaba en una Mark II encontró una polilla atrapada en un relé que impedía las operaciones de dicha computadora.

Grace dijo que al sacar el “bug” (bicho) habían hecho un "debugging“ (desbichado) del sistema.

En una entrevista Grace comentó que el término “bug” como sinónimo de error ya era usado en la jerga (ver más sobre esto en software-bug).

Grace Murray Hopper frente al teclado de UNIVAC I (en 1960)

El primer “bug” en un programa



Una fotografía del supuesto primer "bug" (bicho) real: una polilla (moth) que estaba en el relay#70 de la Mark II. La cual fue “debugged” el 9/9/1947 a las 15:45hs

Ejemplo propuesto

Problema: escriba un programa que busque el menor elemento en un archivo de enteros "numeros.enteros" el cual ya fue creado.

```
Algoritmo buscar el menor
 Abrir el archivo para leer
 Si es fin de archivo entonces
    mostrar no hay menor
 Sino
    Leer el primer elemento
       Guardar el primero como el menor encontrado hasta ahora
    Repetir mientras no sea fin de archivo (EOF)
       Leer un elemento nuevo del archivo
       Si el elemento recién leído es menor al encontrado hasta ahora
       entonces el elemento recién leído es el nuevo menor encontrado
    Mostrar el elemento menor
 Cerrar el archivo.
fin.
```

Ejemplos significativos / casos de prueba

Archivo = 1, 2, 3 Archivo = 8, 3, 4, 3, 4

Archivo = vacío Archivo = 8

Archivo = 3, 2, 1

Realizaremos el programa en el pizarrón

Resolución de Problemas y Algoritmos Dr. Diego R. García 5

Parte de la implementación que hicimos en el pizarrón

```
Program buscar_menor;
TYPE  mis_nums = FILE OF integer; // tipo nuevo creado por el constructor
VAR  numeros: mis_nums; // manejador del archivo de enteros
     menor, leido : integer;  // el menor encontrado y el último leído
begin
assign(numeros, 'numeros.enteros'); // vincula nombre con el manejador
reset(numeros); // abre el archivo para leer
IF  eof(numeros) THEN writeln( ' archivo vacío, no hay menor' )
ELSE BEGIN
     read(números,menor); //guardo el primero como el menor encontrado
      // ahora miro el resto del archivo para ver si hay otro menor….
     WHILE NOT eof(numeros) DO // mientras no sea el fin del archivo
       begin

          … // el resto del programa fue desarrollado en el pizarrón
```

Resolución de Problemas y Algoritmos Dr. Diego R. García 6

Tarea propuesta

Problema: escriba un programa que indique cual es y cuantas veces aparece el menor elemento en un archivo de enteros "numeros.enteros" el cual ya fue creado.

Ejemplos significativos / casos de prueba

Archivo = -9, 2, 3, -9, 2, -9	el menor es el -9 y está tres veces
Archivo = 8, 8, 4, 3, 4, 3	el menor es el 3 y está dos veces
Archivo = vacío	no existe el menor
Archivo = 8	el menor es el 8 y está una vez
Archivo = 3, 3, 3, 1	el menor es el 1 y está una vez

Algoritmo cuantas veces el menor

Tarea para practicar

(no deje de mostrarle a un docente su propuesta)

fin.

Resolución de Problemas y Algoritmos Dr. Diego R. García 7

Eliminar un elemento de un archivo

Problema: escriba un programa que solicite al usuario un elemento y elimine todas las apariciones de ese elemento del archivo de enteros "numeros.enteros"

Ejemplos significativos / casos de prueba

Archivo = 1, 2, 3 eliminar el 2	Archivo = 1, 3
Archivo = 1, 2, 3 eliminar el 5	Archivo = 1, 2, 3
Archivo = 8, 3, 4, 3, 4 eliminar el 4	Archivo = 8, 3, 3
Archivo = vacío eliminar el 2	Archivo = vacío
Archivo = 8 eliminar el 8	Archivo = vacío

Realizaremos el algoritmo y programa en el pizarrón

Resolución de Problemas y Algoritmos Dr. Diego R. García 8

Eliminar un elemento de un archivo

Problema: escriba un programa que solicite al usuario un elemento y elimine todas las apariciones de ese elemento del archivo de enteros "numeros.enteros"

Realizaremos el programa en el pizarrón

Algoritmo eliminar un elemento

Pedir y leer el elemento a eliminar

Abrir el archivo original para leer, y abrir un archivo auxiliar para escribir

mientras no sea fin de archivo original

Leer un elemento del original

Si el elemento recién leído es distinto al que quiero eliminar

entonces escribir el recién leído al archivo auxiliar

Cerrar ambos archivos

Abrir auxiliar para leer y original para escribir

Copiar todos los elementos del auxiliar al original.

Como fue explicado en un problema anterior

fin.

Resolución de Problemas y Algoritmos Dr. Diego R. García 9

Más problemas para practicar

Exclusivo para aquellos alumnos que ya terminaron:

- **todos los prácticos y**
- **todos los que fueron planteados en clase y dejados como tarea.**

Problemas

Problema: escriba un programa que indique cual es y cuantas veces aparece el menor elemento en un archivo de enteros "numeros.enteros" el cual ya fue creado. Plantee casos de prueba.

Problema: escriba un programa que solicite al usuario un elemento y elimine todas las apariciones de ese elemento del archivo de enteros "numeros.enteros". Plantee casos de prueba.

Problema: escriba un programa que solicite al usuario un elemento y determine si el elemento está o no en un archivo de enteros "numeros.enteros". Plantee casos de prueba. ¿Necesita recorrer todo el archivo?

Problema: escriba un programa que determine si dos archivos de enteros (llamados "enteros.uno" y "enteros.dos") son iguales. Considere que dos archivos son iguales si: tienen exactamente los mismos elementos en el mismo orden y la misma cantidad de elementos. Plantee casos de prueba.

Resolución de Problemas y Algoritmos Dr. Diego R. García

Problemas

Problema: Escribir un programa para determinar si un número natural N es primo

Definición: un número es primo si es un entero positivo mayor que 1, que es divisible solamente por si mismo y la unidad.

Problema: escriba un programa que busque cuantas veces está el elemento E (ingresado por el usuario) en una secuencia de números naturlales ingresada por teclado (buffer) y usando 0 como terminador.

Problema: escriba un programa que busque cuantas veces está el dígito D (ingresado por el usuario) en un número.

Resolución de Problemas y Algoritmos Dr. Diego R. García

Problemas ¿más díciles?

Problema (intersección): escriba un programa que dados dos archivos de enteros (llamados "enteros.uno" y "enteros.dos"), genere un tercer archivo "enteros.comunes" con los elementos que están en los dos archivos. Plantee casos de prueba. Puede hacer (a) una versión que no considere las repeticiones y luego (b) otra versión que considere las repeticiones de los elementos.

Problema (diferencia): escriba un programa que dados dos archivos de enteros (llamados "enteros.uno" y "enteros.dos"), genere un tercer archivo "nocomunes.dif" con los elementos que están en el primer archivo, pero no están en el segundo. Plantee casos de prueba. Puede hacer (a) una versión que no considere las repeticiones y luego (b) otra versión que considere las repeticiones de los elementos.

Problema propuesto

Problema: Escribir un programa para determinar si un número natural N es primo

Definición: un número es primo si es un entero positivo mayor que 1, que es divisible solamente por si mismo y la unidad.

Ejemplos (futuros casos de pruba):

2 , 3, 7, 11 y 2003: son primos
4 no es primo es divisible por 2
2001 no es primo, es divisible por 3
121 no es primo, es divisible por 11

Una solución: si N tiene un divisor distinto de N o 1 entonces NO es primo, de lo contrario es primo. Entonces para hacer un algoritmo puedo asumir que es primo hasta que se demuestre lo contrario (esto es, voy viendo una a uno si encuentro un divisor)

Algoritmo para "esPrimo"

```
ALGORITMO EsPrimo
COMIENZO
 leer el número N
 SI (N > 1)
 ENTONCES  EsPrimo ← Verdadero {asumo  que es primo}
 SINO EsPrimo ← Falso
 Divisor ← 2  { el primer divisor a considerar es 2  ... }
 REPETIR MIENTRAS (Divisor <  N) y (esPrimo = verdadero)
     SI   N es divisible por Divisor    ( usando módulo: N// Divisor = 0)
         ENTONCES esPrimo ← falso {Si Divisor divide a N, NO ES PRIMO}
      Divisor ← Divisor + 1 {paso al próximo posible Divisor }
 fin repetir mientras
FIN ALGORITMO
```

Tarea 1: Implemente en Pascal este algoritmo usando WHILE.
Tarea 2: Luego implemente usando REPEAT-UNTIL

Resolución de Problemas y Algoritmos Dr. Diego R. García