Universidad Tecnica Federico Santa Maria
Departamento de Física

UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Tarea 3

Fisica computacional

Integrante: Marcelo Durán
Profesor: Edson Carquin
Ayudante: Gonzalo Muñoz

Índice de Contenidos

Índice de Figuras

1. problema 1

Problem #1:

a) Please create an algorithm that solves the ancient Towers of Hanoi puzzle. To solve the puzzle, you must come up with a series of steps to move a stack of different sized rings from one pole to another. They must be moved one at a time, using only simple intermediate pole, so that not ring is ever placed on top of a smaller ring.
b) Please provide an estimate of the asymptotic behavior of the execution time (for $n \to \infty$) of the algorithm developed in (a).
c) Please create a program in C++, that implements your algorithm in (a).
d) Be free to represent the output of your program by using a plot, a table or whatever you think is the best way to do it so. You could use an alternative language like Mathematica or Python in order to display the results obtained with your program, but in that case, you should provide us with precise instructions to manage properly your codes.

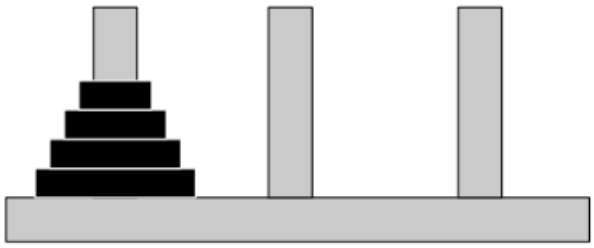

Figura 1

1.1. pregunta a)

solución:

El algoritmo es de tipo recursivo. Primero debemos mover los n-1 discos superiores desde el pilar de inicio hasta el pilar comodín o auxiliar. Posteriormente, debemos ubicar el disco de la base desde el inicio hasta el pilar final. Por último, movemos la torre de n-1 discos desde el pilar auxiliar hasta el pilar final, siempre respetando las reglas básicas del juego. Los pasos mínimos necesarios para resolver este juego son $2^n - 1$, donde n es el número de discos. Como se puede ver, el número de pasos crece exponencialmente.

1.2. pregunta b)

solución:
El tiempo de ejecución del programa propuesto en el inciso anterior es del orden de 2^{n-1}. Se puede conjeturar que la forma completa del tiempo en función del número de discos es $f(n) = a2^{n-1}$, donde a es un coeficiente que determina la tasa de crecimiento de la función exponencial. Este valor puede ser calculado mediante mínimos cuadrados, pero como escapa a los fines de esta tarea, no se calculará.

1.3. pregunta c)

solución:
El código escrito en C++ para resolver el problema nos pide ingresar el número de discos que tendrá el juego, así como elegir cuál será el pilar de inicio, final y auxiliar de entre los pilares A, B, C. En el inciso siguiente se muestra el código ejecutado. (El código esta en el archivo TAREA3MarceloDuránS.zip)

1.4. pregunta d)

solución:
El código mencionado anteriormente se ve de la siguiente manera una vez compilado y ejecutado.

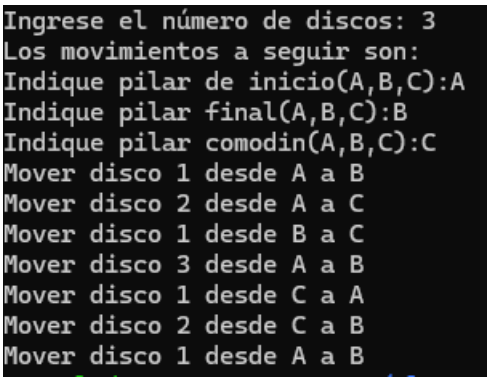

Figura 2

2. problema 2

Problem #2:

Write a program using C++, which implements an algorithm to calculate recursively $\sqrt{2}$, with a precision of 6 decimal digits.

Figura 3

2.1. pregunta a)

solución:

Para resolver este problema existen múltiples algoritmos proporcionados por el análisis numérico, Para la resolución de este problema utilizare el método de Newton-Rhapson, para esto definimos la función $f(x) = x^2 - 2$. ahora recordemos la formula del método como:

$$x_{n+1} = x_n - \frac{f(x)}{f'(x)} \tag{1}$$

utilizando la función $f(x)$, definida anteriormente, obtenemos, la siguiente formula recursiva.

$$x_{n+1} = x_n - \frac{x_n^2 - 2}{2x_n} \tag{2}$$

reordenando.

$$x_{n+1} = \frac{1}{2}(x_n - \frac{2}{x_n}) \tag{3}$$

implementamos el algoritmo recursivo en c++, obteniendo.

```
Escribe un valor inicial de x: 1
La aproximación para la raíz cuadrada de 2 mediante el método de Newton-Raphson es: 1.414214
```

Figura 4

3. problema 3

Problem #3:

Write a C++ program that reads a word from the keyboard, stores it in a string and checks whether the word is a palindrome. A palindrome reads the same from left to right as from right to left (otto, level and deed are examples of palindromes). Use the subscript operator [ ]. Modify the code to continually read and check new words and store the output into a file with the test word and the result of the test in the same line.

Figura 5

3.1. pregunta a)

solución:

El código utilizado, nos pregunta cuantas palabras queremos verificar si son palíndromos o no. luego ejecuta un ciclo 'for' en el cual por cada palabra que ingresemos nos entrega el resultado de si es palíndromo, por ultimo guarda en un archivo.txt cada palabra con su respectivo resultado, "SI"para las palabras que son palíndromos y "NO"para las que no son. para mas detalles del programa lea el archivo README.txt". acá se ve un ejemplo de lo mencionado.

```
Numero de palabras a verificar:3
Ingrese una palabra:level
La palabra'level'es un palindromo
Se a añadido la palabra al archivo
Ingrese una palabra:otto
La palabra'otto'es un palindromo
Se a añadido la palabra al archivo
Ingrese una palabra:world
La palabra'world'no es un palindromo
Se a añadido la palabra al archivo
```

Figura 6

4. problema 4

Problem #4:

a) Explain each of the following definitions. Indicate whether any are illegal and if so why.

```
(a) int* ip;
(b) string s, *sp = 0;
(c) int i; double* dp = &i;
(d) int* ip, ip2;
(e) const int i = 0, *p = i;
(f) string *p = NULL;
```

b) Given a pointer, p, can you determine whether p points to a valid object? if so, how? It not, why not?

c) Why is the first pointer initialization legal and the second illegal?

```
int i = 42;
void *p = &i;
long *lp = &i;
```

Figura 7

4.1. pregunta a)

solución:

a) **int* ip;**= Esta linea de código es legal, ya que, esta declarando una variable llamada *ip*, la cual apunta a una variable de tipo entero en la dirección de memoria.

b) **string s, *sp = 0;**= Esta linea es legal, ya que, crea una variable "sz un puntero "sp"del tipo string.

c) **int i; double* dp=&i;**= Esta linea es ilegal, ya que, se intenta asignar una variable i a un puntero del tipo double.

d) **int* ip, ip2;**= Esta linea de código es legal, ya que crea un puntero de tipo entero y una variable asignada a ese puntero, también de tipo entero.

e) **const int i = 0, *p = i;** = Esta linea es ilegal, ya que, no se define el tipo del puntero, por lo tanto, no es posible inicializar el puntero con una variable de tipo int.

f) **string *p =NULL;**=es legal, ya que, nos dice que el puntero p, no apunta a nada en la dirección de memoria.

4.2. pregunta b)

solución:
No es posible, ya que, el puntero puede no estar inicializado, así como también apuntar a direcciones de memoria liberada. ademas existen casos donde la corrupción de memoria afecta este proceso.

4.3. pregunta c)

solución:
Porque la variable que se define es del tipo int, en la primera declaración se esta inicializando un puntero del tipo vid el cual es un puntero genérico que puede apuntar a cualquier tipo de variable, mientras que el segundo puntero es del tipo long, lo cual no concuerda con la variable i la cual es del tipo int.