# escomipnINSTITUTO POLITECNICO NACIONAL

# Escuela Superior de Cómputo

## Hernández Hernández Alejandro

## Boleta:2016630177

## Análisis de Algoritmos

## M. En C. Luz María Sanchez García

## Practica 1

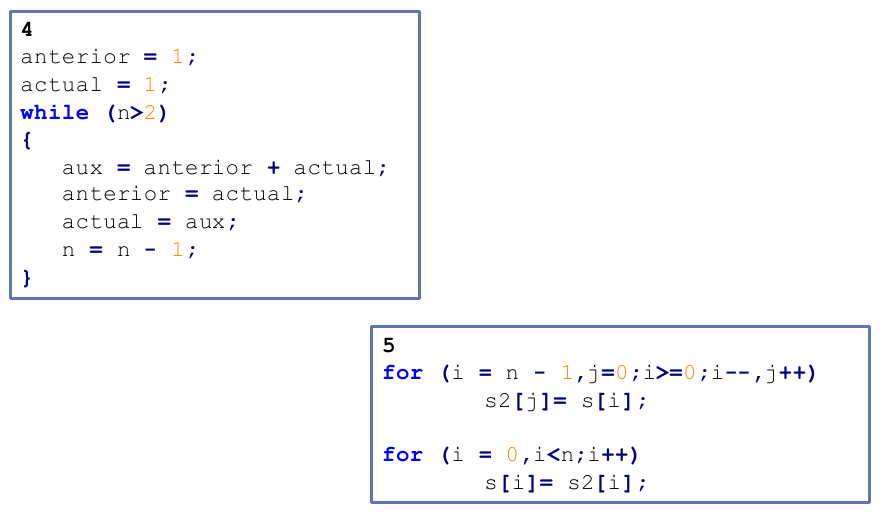
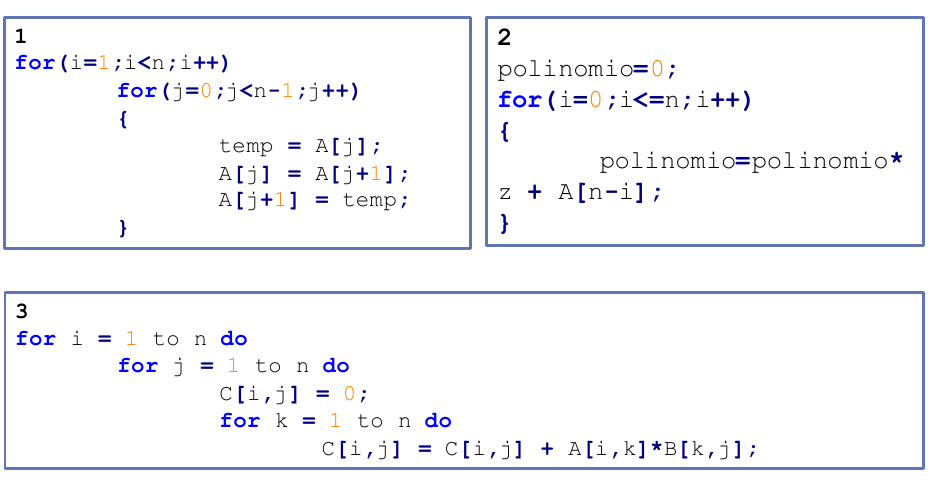
Fecha de entrega: 25-02-2018

# INTRODUCCIÓN

En esta practica, se muesran 5 algoritmos “basicos” utilizados en programacion, con el fin de conocer cuan costosa es la ejecucion de ellos, haciendo las correspondientes pruebas con distintos valores ingresados, siendo estos desde los mas pequeños, hasta datos consideralemente grandes, y ademas conociendo el costo espacial de los mismos programas.

# PLANTEAMIENTO DEL PROBLEMA

Conocer la complejidad temporal y espacial de los algoritmos proporcionados, y que seran mostrados a continuacion, al llevarse a cabo en un ambito real, siendo programados en lenguaje C.



### Solucion

Al momento de programar cada algoritmo, tener una variable que se incremente en cada accion que realize el mismo, mostrando al final de la ejecucion el tiempo que tardo en ejecutarse, dependiendo de los datos de entrada.

Para ello, se crearon 2 funciones: el main, y la funcion que contiene el algoritmo a probar.

Ahora bien, para calcular los costos de cada linea ejectuada, se declara una variable de tipo entero, llamada costo o contador, que se incrementara cada vez que se ejecuteuna linea del codigo original, evidentemente esto reresenta costos extra al programa como tal, pero para fines didacticos, se hará asi.

Codigos:

1)

1. #include <stdio.h>
2. #include<stdlib.h>
4. **void** pfor(**int** n,**int** \*A);
5. **void** algor();
7. **int** main(){
8. **int** val;
9. printf("ingrese valor de n: ");
10. scanf("%d",&val);
11. **int** A[val];
12. **for**(**int** i=0;i<val;i++){
13. A[i]=i+1;
14. }
15. algor();
16. pfor(val,A);
17. }
19. **void** pfor(**int** n, **int** \*A){
20. **int** contador=0;
21. **int** temp;contador+=3;
22. **for**(**int** i=1;i<n;i++){contador+=2;
23. **for**(**int** j=0;j<n-1;j++){contador+=2;
24. temp=A[j];contador+=1;
25. A[j]=A[j+1];contador+=1;
26. A[j+1]=temp;contador+=1;
27. }
28. }
29. contador+=2;
30. printf("Costo del algoritmo: %d \n",contador);
31. printf("Costo espacial del algoritmo: %d\n",3+n);
32. }
34. **void** algor(){
35. printf("int temp; //costo=1;\nfor(int i=1;i<n;i++){ //costo:1 asignacion, n comparaciones, n-1 incrementos\nfor(int j=0;j<n-1;j++){ //costo: 1 asignacion, n comparaciones, n-1 incrementos\n");
36. printf("temp=A[j];//costo=1;\nA[j]=A[j+1];//costo=1;\nA[j+1]=temp;//costo=1;\n}\n}\n");
37. }

2)

1. #include <stdio.h>
2. #include<stdlib.h>
3. #include<time.h>
5. **void** polinomio(**int** n, **int** z, **int** \*A);
6. **void** algor(**int** z,**int** n);
8. **int** main(){
9. srand(time(NULL));
10. **int** n;
11. printf("Ingrese valor de n: ");
12. scanf("%d",&n);
13. **int** randz=rand()%50,A[n];
14. **for**(**int** i=0;i<n;i++){
15. A[i]=rand()%20;
16. }
17. algor(A,randz,n);
18. polinomio(n,randz,A);
20. }
22. **void** polinomio(**int** n,**int** z, **int** \*A){
23. **int** cont=0;
24. **int** poli=0;cont=2;
25. **for**(**int** i=0;i<=n;i++,cont++){
26. cont+=1;
27. poli=poli\*z+A[n-1];cont+=3;
28. }
29. cont+=1;
30. printf("costo del algoritmo: %d \n",cont);
31. printf("Costo espacial del algoritmo: %d\n",2+n);
33. }
35. **void** algor(**int** z,**int** n){
36. printf("int poli=0;//costo: 1\nfor(int i=0;i<=n;i++){//1 asig, n+2 comp, n+1 incrementos\n poli=poli\*z+A[n-1];costo: 3\n}\n");
37. printf("valor de z=%d\n",z );
38. printf("Tamaño de A: %d\n",n);
40. }

3)

1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<time.h>
5. **void** for2(**int** n, **int** A[][n],**int** B[][n],**int** C[][n]);
6. **void** algor();
8. **int** main(){
9. srand(time(NULL));
10. **int** n;
11. printf("Ingrese valor de n: ");
12. scanf("%d",&n);
13. **int** C[n][n],A[n][n],B[n][n];
14. **for**(**int** i=0;i<n;i++){
15. **for**(**int** j=0;j<n;j++){
16. C[i][j]=rand()%10;
17. A[i][j]=rand()%10;
18. B[i][j]=rand()%10;
19. }
20. }
22. algor();
23. for2(n,A,B,C);
25. }
27. **void** for2(**int** n, **int** A[][n],**int** B[][n],**int** C[][n]){
28. **int** costo=3;
29. **for**(**int** i=1;i<=n;i++,costo++){
30. **for**(**int** j=1;j<=n;j++,costo++){
31. C[i][j]=0;costo+=1;
32. **for**(**int** k=1;k<=n;k++,costo++){
33. C[i][j]=C[i][j]+A[i][k]\*B[k][j];costo+=3;
34. }costo+=1;
35. }costo+=1;
36. }costo+=1;
38. printf("Costo del algoritmo: %d\n",costo);
39. printf("Costo espacial del algoritmo: %d\n",3+3\*(n\*n));
40. }
42. **void** algor(){
43. printf("for(int i=1;i<=n;i++){//1 asig, n+1 comp,n incr\nfor(int j=1;j<=n;j++){//1 asig, n+1 comp,n incr\nC[i][j]=0;//costo:1\nfor(int k=1;k<=n;k++){//1 asig, n+1 comp,n incr\n");
44. printf("C[i][j]=C[i][j]+A[i][k]\*B[k][j];//costo: 3\n}\n}\n}\n");
45. }

4)

1. #include<stdio.h>
2. #include<stdlib.h>
4. **void** while1(**int** n);
5. **void** algor();
7. **int** main(){
8. **int** n;
9. printf("ingrese valor de n:");
10. scanf("%d",&n);
11. algor();
12. while1(n);
13. }
15. **void** while1(**int** n){
16. **int** costo;
17. **int** anterior=1,actual=1,aux;costo=3;
19. **while**(n>2){costo+=1;
20. aux=anterior + actual;costo+=2;
21. anterior=actual;costo+=1;
22. actual=aux;costo+=1;
23. n=n-1;costo+=2;
24. }costo+=1;
26. printf("Costo del algoritmo: %d \n",costo );
27. printf("Costo espacial del algoritmo: 4\n");
28. }
30. **void** algor(){
31. printf("int anterior=1,actual=1,aux;//costo: 3\nwhile(n>2){//n-1  comp\naux=anterior + actual;//costo: 2\nanterior=actual;//costo: 1");
32. printf("\nactual=aux;//costo: 1;\nn=n-1;//costo: 2;\n}\n");
33. }

5)

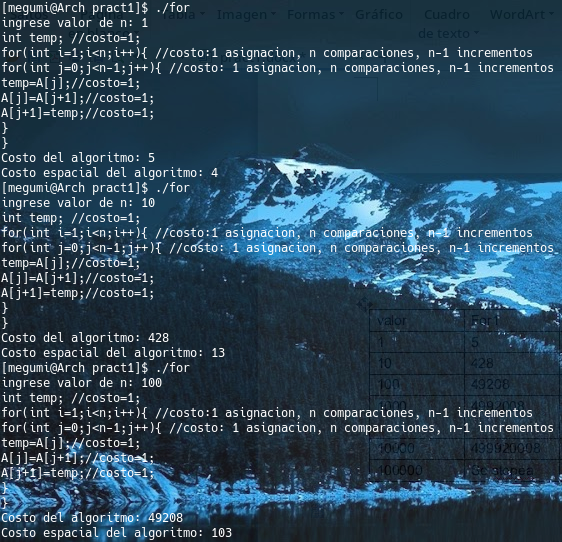
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<time.h>
5. **void** dobfor(**int** n, **int** \*s, **int** \*s2);
6. **void** algor();
8. **int** main(){
9. srand(time(NULL));
10. **int** n;
11. printf("inrese valor de n: ");
12. scanf("%d",&n);
13. **int** s[n],s2[n];
15. **for**(**int** i=0;i<n;i++){
16. s[i]=rand()%10;
17. s2[i]=rand()%10;
18. }
20. algor();
21. dobfor(n,s,s2);
22. }
24. **void** dobfor(**int** n, **int** \*s, **int** \*s2){
25. **int** costo=2;
26. **for**(**int** i=n-1, j=0;i>=0;i--,j++,costo+=2){costo+=1;
27. s2[j]=s[i];costo+=1;
28. }costo+=2;
29. **for**(**int** i=0;i<n;i++,costo++){costo+=1;
30. s[i]=s2[i];costo+=1;
31. }costo+=1;
33. printf("Costo del algoritmo: %d\n",costo);
34. printf("Costo espacial del algoritmo: %d\n",2+(2\*n));
35. }

38. **void** algor(){
39. printf("for(int i=n-1, j=0;i>=0;i--,j++){//2 asig, n+1 comp,n incr, n decr\n s2[j]=s[i];//costo: 1\n}\nfor(int i=0;i<n;i++){1 asig, n+1 comp, n incr\n s[i]=s2[i];//costo: 1\n}\n");
40. }

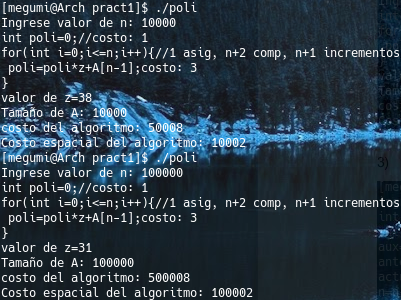
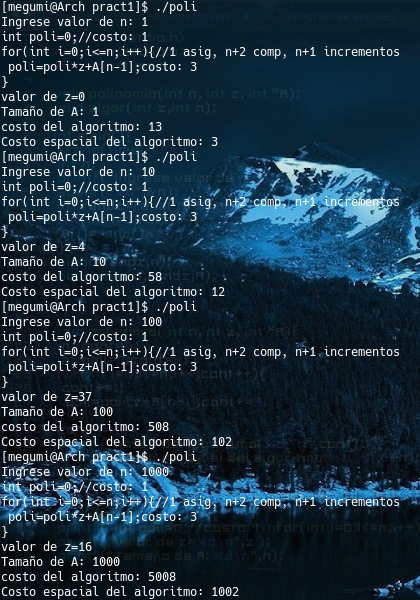
Salidas:

Se ejecutaron los programas, con el compilador gcc de linux, en Arch Linux como sistema operativo, y Sublime Text como editor del codigo, en una laptop marca Asus modelo X540SA, con 2gb de memoria RAM, y procesador Intel Celeron N3050 de 1.60GHz

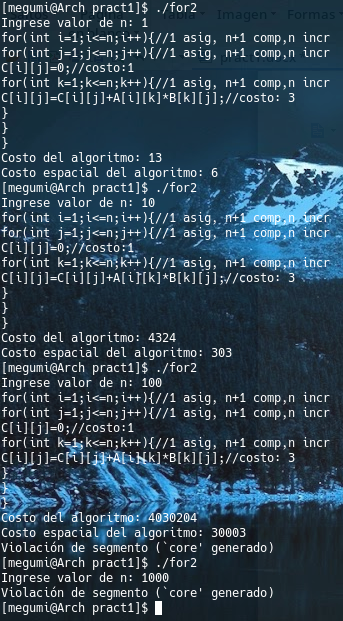
1)



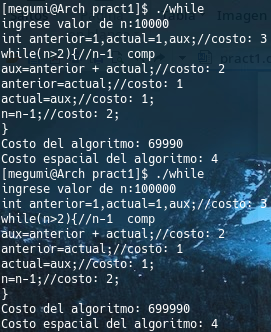
2)



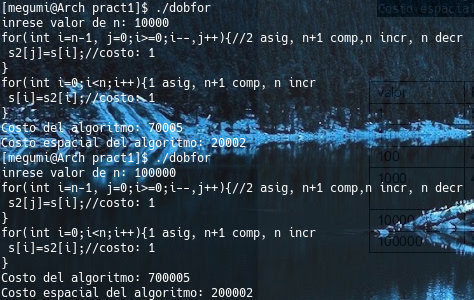
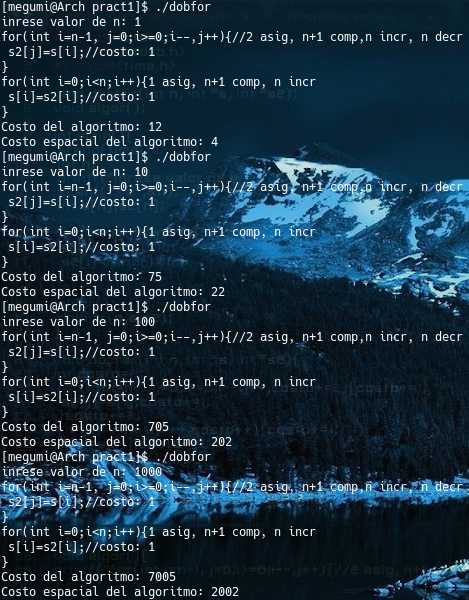
3)



4)

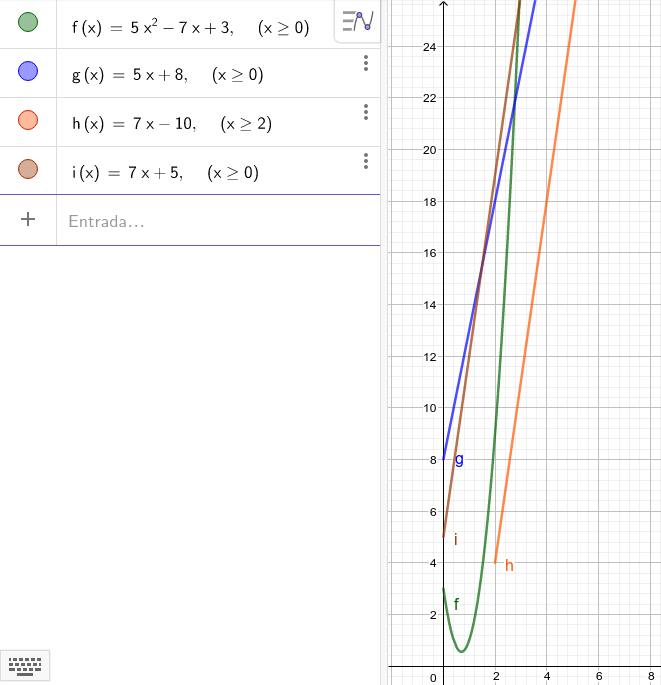


5)

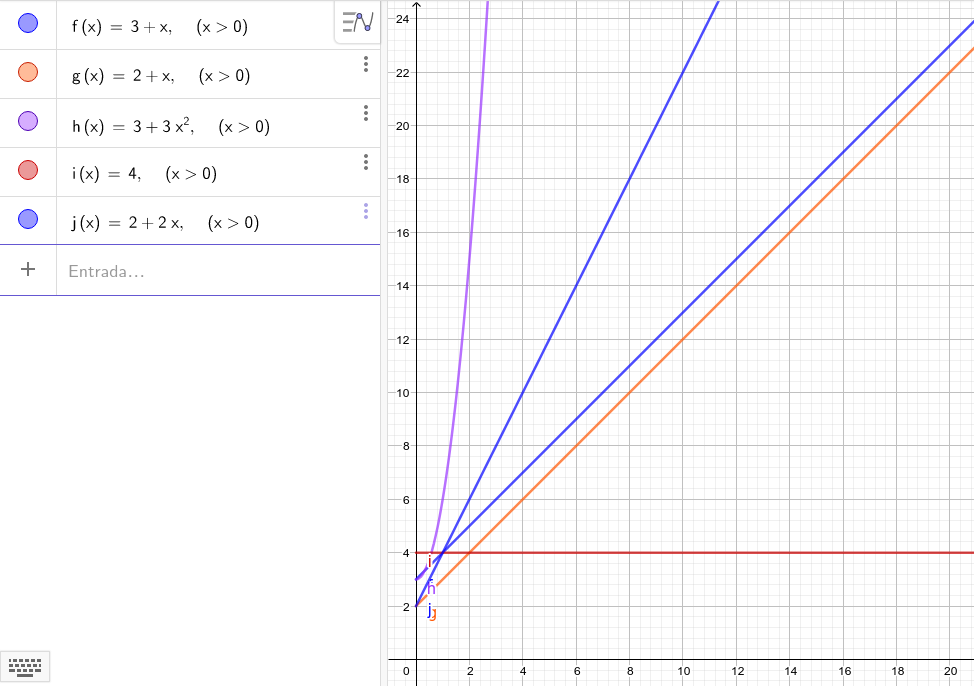


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| valor | For1 | polinomio | while | For2 | For anidado |
| 1 | 5 | 13 | 4 | 13 | 12 |
| 10 | 428 | 58 | 60 | 4324 | 75 |
| 100 | 49208 | 508 | 690 | 4030204 | 705 |
| 1000 | 4992008 | 5008 | 6990 | Core generado | 7005 |
| 10000 | 499920008 | 50008 | 69990 | ------------ | 70005 |
| 100000 | Se stopea | 500008 | 699990 | ------------- | 700005 |

Complejidad temporal aproximada



Complejidad espacial aproximada



# Conclusión

Esta practica me mostóo de forma explícita el tiempo de ejecución de varios programas, especificamente de algunas sentencias comunes en la programacion, como los for, while y for anidados que, desde una máquina con bajos recursos, pudo mostrarlos, al punto de llevarla al límite e intentar “matarla”, ingresando valores de entrada considerables como 10000 o 100000 que fueron los mas grandes, y en uno de ellos (el triple for anidado), solo me dejó ingresar un valor de 100 con un error de core generado al terminar la ejecución del programa, y en el for con arreglos, con la entrada 100000, el programa se tardaba para tratar de dar un resultado.