



geog  
**INSTITUTO POLITÉCNICO NACIONAL**

**Escuela Superior de Cómputo**



**Hernández Hernández Alejandro  
Rangel Moreno Jose Luis Irepan**

**Boleta:**

**2016630177**

**2016630321**

**Análisis de Algoritmos**

**M. En C. Luz María Sanchez García**

**Practica 8**

Fecha de entrega: 20-05-2018

# INTRODUCCIÓN

En esta práctica, se analiza el funcionamiento de los algoritmos del problema del cambio y de la multiplicación óptima de matrices.

## PLANTEAMIENTO DEL PROBLEMA

- Implementar los algoritmos siguientes de **programación dinámica** en C:
  - Multiplicación óptima de matrices.
  - El problema del cambio
- Comprobar las soluciones que arrojan y su optimalidad.
- Medir los tiempos de ejecución de las implementaciones para distintos tamaño de problema.

## Solución

Dado que ya se conoce el funcionamiento de cada algoritmo, solo resta codificarlo en lenguaje C.

## Códigos

### Cambio.c

```
#include <stdio.h>
#define MAX 1000

void makeChange(int coin[], int n, int value){
    int i, j, k;
    int min_coin[MAX];
    int count[MAX+1][50]={0};
```

```

int min;
min_coin[0] = 0;
if(value==0){
    min=0;
}else{
    if(n==0){
        min=value;
        printf("\nNo hay monedas que dar");
    }else{
        for (i=1; i <= value; i++){
            min = value;
            for (j = 0; j<n; j++){
                if (coin[j] <= i){
                    if (min > min_coin[i-coin[j]]+1){
                        min = min_coin[i-coin[j]]+1;
                        for(k = 0; k < n; ++k){
                            count[i][k] = count[i-coin[j]][k];
// copiar coin en counts cuando value=i-coin[j]
                        }
                        count[i][j]++;
                    }
                }
            }
            min_coin[i] = min;
        }
    }
}
printf("\nMinimo de monedas que se requieren %d \n", min_coin[value]);
for(int i = 0; i < n; ++i){
    printf("%d: %d\n", coin[i], count[value][i]);
}
for(int i = 0; i < n; i++){
    for(int j = 0; j <= value; j++){
        printf("%d,",count[j][i]);
    }
    printf("\n");
}
}

int main(){
    int n,i,value;
    printf(";Cuantas monedas hay? ");
    scanf("%d",&n);
    int coin[n];
    printf("Ingresa los valores de las monedas: \n");
    for(i=0; i<n; i++){
        scanf("%d",&coin[i]);
    }
    printf("Cambio que se dara: ");
    scanf("%d",&value);
    makeChange(coin,n,value);
}

```

## Multiplicación óptima de matrices

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

```

#include <stdio.h>

/*
 * according to introduction to algorithm,
 * we alloc array(n+1)but make use of the last n
 * just for convinence
 */
void MATRIX_CHAIN_ORDER(int *p,int n,int **m,int **s)
{
    int i;
    int row = n+1;
    //initialize our array
    for(i=1;i<=n;i++)
        *((int*)m+row*i+i) = 0;

    int l;
    for(l=2;l<=n;l++)
    {
        for(i=1;i<=(n-l+1);i++)
        {
            int j = i+l-1;
            *((int*)m+row*i+j) = -1;
            int k;
            for(k=i;k<=(j-1);k++)
            {
                int tmp1 = *((int*)m+row*i+k);
                int tmp2 = *((int*)m+row*(k+1)+j);
                int q = tmp1+tmp2+p[i-1]*p[k]*p[j];

                int old = *((int*)m+row*i+j);
                if(q<old || old == -1)
                {
                    *((int*)m+row*i+j) = q;
                    *((int*)s+row*i+j) = k;
                }
            }
        }
    }
}

void PRINT_OPTIMAL_PARRNS(int **s,int i,int j,int row)
{
    if(i==j)
        printf("A%d",i);
    else{
        printf("(");
        PRINT_OPTIMAL_PARRNS(s,i,*((int*)s+row*i+j),row);
        PRINT_OPTIMAL_PARRNS(s,*((int*)s+row*i+j)+1,j,row);
        printf(")");
    }
}

int main()
{
    int n = 6;
    int p[7];

```

```

p[0]=30;//estos valores determinan el tamaño de las matrices
p[1]=300;
p[2]=13;
p[3]=22;
p[4]=112;
p[5]=201;
p[6]=20;

int m[n+1][n+1];
int s[n+1][n+1];

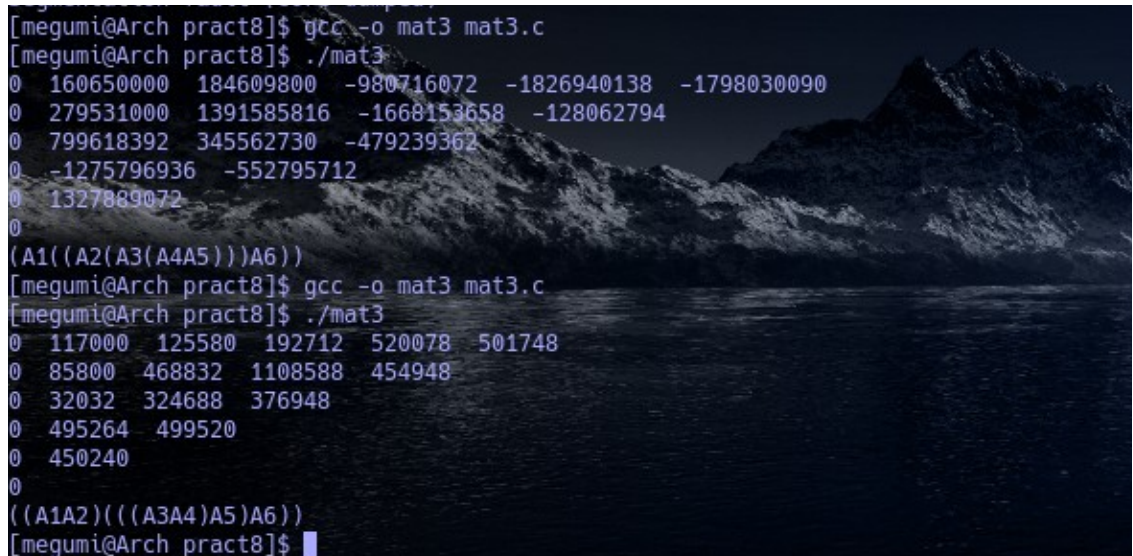
MATRIX_CHAIN_ORDER(p,n,(int**)m,(int**)s);
int i,j;
for(i=1;i<=n;i++)
{
    for(j=i;j<=n;j++)
    {
        printf("%d  ",m[i][j]);
    }
    printf("\n");
}

PRINT_OPTIMAL_PARRNS((int**)s,1,6,7);

printf("\n");
return 0;
}

```

**SALIDAS:** Los algoritmos fueron probados en una máquina Asus x540sa, 2gb de RAM con sistema operativo Arch Linux, compilador gcc para Linux y Sublime Text como editor de texto,



```

[megumi@Arch pract8]$ gcc -o mat3 mat3.c
[megumi@Arch pract8]$ ./mat3
0 160650000 184609800 -980716072 -1826940138 -1798030090
0 279531000 1391585816 -1668153658 -128062794
0 799618392 345562730 -479239362
0 -1275796936 -552795712
0 1327889072
0
(A1((A2(A3(A4A5)))A6))
[megumi@Arch pract8]$ gcc -o mat3 mat3.c
[megumi@Arch pract8]$ ./mat3
0 117000 125580 192712 520078 501748
0 85800 468832 1108588 454948
0 32032 324688 376948
0 495264 499520
0 450240
0
((A1A2)((((A3A4)A5)A6)))
[megumi@Arch pract8]$

```

irepan@irepan-VirtualBox: ~/Escritorio

Cambio que se dara: 126

Minimo de monedas que se requieren 14

1: 1

2: 0

5: 1

10: 12

[illegible]

0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,  
0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,  
0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,  
0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,0,0,1,1,2,  
0,0,1,1,2,0,0,

0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,  
0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,  
0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,  
0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,  
0,0,0,0,0,1,1,

0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3,3,3,  
4,4,4,4,4,4,4,4,4,5,5,5,5,5,5,5,5,5,6,6,6,6,6,6,6,6,7,7,7,7,7,7,7,7,7,7,  
8,8,8,8,8,8,8,8,9,9,9,9,9,9,9,9,10,10,10,10,10,10,10,10,10,11,11,11,11,  
11,11,11,11,11,11,11,12,12,12,12,12,12,12,

```
irepan@irepan-VirtualBox:~/Escritorio$
```

```
irepan@irepan-VirtualBox:~$ cd Escritorio
```

```
irepan@irepan-VirtualBox:~/Escritorio$ gcc Cambio.c
```

```
irepan@irepan-VirtualBox:~/Escritorio$ ./a.out
```

¿Cuántas monedas hay? 4

Ingrese los valores de las monedas:

1

2

5

19

Cambio que se dara: 126

Minimo de monedas que se requieren 14

1: 1

2: 0

5: 1

10: 12

[illegible]

```
irepan@irepan-VirtualBox:~/Escritorio$
```



## CONCLUSIONES

Jose Luis: Con esta practica vimos algoritmos con los que habíamos trabajado anteriormente en las exposiciones y de esta forma pudimos ver su funcionamiento en la practica añadiendo así más análisis a los algoritmos dinámicos por nuestra parte y poder saber de manera mas fácil tanto su funcionamiento como situación de cuando deben usarse

Alejandro: en esta practica, pudimos analizar el funcionamiento de los algoritmos de programación dinámica, enfocándonos en el problema del cambio, y la multiplicación de matrices, ambos con la finalidad de devolver la solución mas óptima para cada problema. En el caso del cambio, se devuelve la combinación de monedas mas viable para devolver un resto, y en el caso de las matrices, devolver la mejor combinación para realizar la multiplicación de  $N$  matrices de tamaño  $m \times n$ , acotando el numero de multiplicaciones a realizar.