

Compresión de datos

Allan Jair Escamilla Hernández

A lo largo del desarrollo de este ejercicio se va a realizar la compresión de 4 datos numéricos en una sola variable, para después obtener de manera separada cada valor almacenado en la variable. Posteriormente se realizarán operaciones sobre los datos almacenados en la variable.

Pseudocódigo

```
Principal( argc, argv | ){
    number =
    comprimirNumeros(argc, argv);
    n1 = getNumber(number, 1);
    n2 = getNumber(number, 2);
    n3 = getNumber(number, 3);
    n4 = getNumber(number, 4);
}

comprimirNumeros(argc, argv |
number){
    cant = 3;
    Desde i = 0 hasta i = argc{
        Si(argv[i] >= 0 && argv[i]
<= 16)
            number+= (aux <<
(cant*4));
            cant--;
    }
}

getNumber(number, pos | number){
    number <= (4*pos);
    number >= 12;
}
```

Código

```
#include <stdio.h> // Incluyendo
las librerias
#include <stdlib.h>

// Prototipos de las funciones
unsigned short int
comprimirNumeros(int cantArg,
char* Argumentos[]);
unsigned short int
getNumber(unsigned short int
number, int pos);
void menu();
void menu2(unsigned short int
number);
// Funcion principal
int main(int argc, char *argv[])
{
    //printf("%ld\n",
sizeof(unsigned short int));
    unsigned short int number = 0;
    int opcion = 0, pos;
    number =
comprimirNumeros(argc, argv);
    printf("El numero comprimido
es: %d\n", number);
    while (opcion != 4) {
        menu();
        scanf("%d", &opcion);
        switch (opcion) {
            case 1:
                printf("El numero
comprimido es %d\n", number);
                break;
            case 2:
                printf("Ingresar la
posicion del numero deseado->
");
                scanf("%d", &pos);
```

Compresión de datos

Allan Jair Escamilla Hernández

```
        if(pos > 0 && pos < 5){
            printf("El numero es
%d\n", getNumber(number,
pos-1));
        }else{
            printf("La posicion
ingresada no existe\n");
        }
        break;
    case 3:
        menu2(number);
        break;
    case 4:
        printf("Saliendo del
programa...\n");
        exit(0);
        break;
    default:
        printf("Ingresar una
opcion valida!\n");
    }
    printf("Presiona una tecla
para continuar...\n");
    __fpurge(stdin);
    getchar();
}
return 0;
}
```

// DESARROLLANDO LAS FUNCIONES

```
// Funcion que comprime Los 4
numeros en un solo integer
unsigned short int
comprimirNumeros(int cantArg,
char* Argumentos[]){
    unsigned short int number = 0;
    int cant = 3, aux;
    if(cantArg > 5 )
        return 0;
    for(int i = 1; i < cantArg;
i++){ // Recorriendo el array
        aux = atoi(Argumentos[i]);
        if (atoi(Argumentos[i]) &&
aux >= 0 && aux <= 16) { //
```

*Validando Los numeros que se han
ingresado*

```
        number+= (aux <<
(cant*4));
        cant--;
    }else{
        printf("Ha ocurrido un
error con los
argumentos...\nSaliendo del
programa...\n");
        exit(0);
        return 0;
    }
}
return number; // Retornando
el numero comprimido
}
```

// Funcion que despliega el menu

```
void menu(){
    system("clear");
    printf("\t1.- Ver numero
comprimido.\n");
    printf("\t2.- Obtener un
numero.\n");
    printf("\t3.- Realizar
operacion con los numeros.\n");
    printf("\t4.- Salir.\n\n");
    printf("Ingresar opcion-> ");
}
```

*// Funcion que obtiene un numero
dentro del numero comprimido*

```
unsigned short int
getNumber(unsigned short int
number, int pos){
    number <<= (4*pos);
    number >>= 12;
    return number;
}
```

*// Funcion que despliega el menu
dos*

```
void menu2(unsigned short int
number){
```

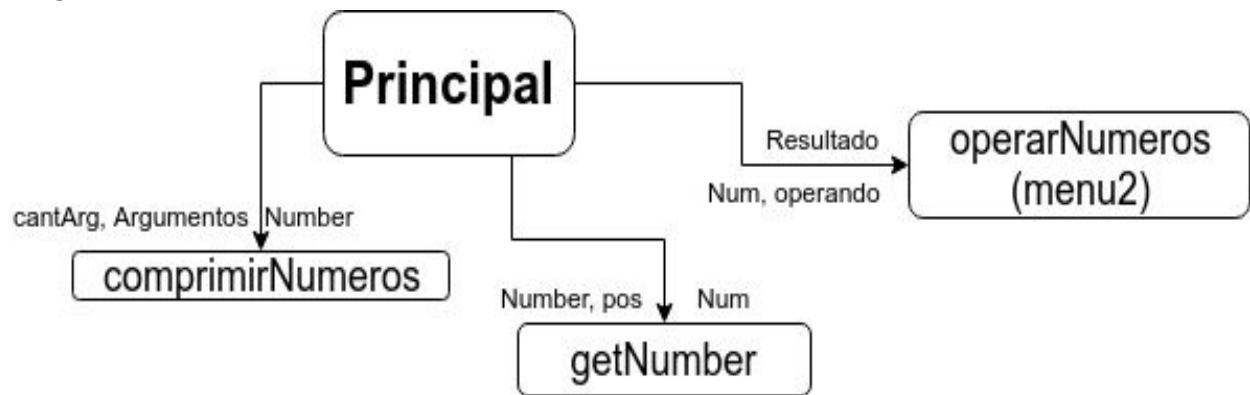
Compresión de datos

Allan Jair Escamilla Hernández

```
float resultado = 0;
int operacion;
printf("\n Operaciones con los
numeros\n");
printf("1.- Suma.\n");
printf("2.- Resta.\n");
printf("3.-
Multiplicacion.\n");
printf("4.- Division.\n");
printf("\n\nSeleccionar
operacion a realizar-> ");
scanf("%d", &operacion);
switch (operacion) {
    case 1:
        for(int i = 0; i < 4; i++)
            resultado+=
getNumber(number, i);
        break;
    case 2:
        for(int i = 0; i < 4; i++)
            resultado-=
getNumber(number, i);
        break;
    case 3:
        resultado = 1;
        for(int i = 0; i < 4; i++)
            resultado*=
getNumber(number, i);
        break;
    case 4:
        resultado = 1;
        for(int i = 0; i < 4; i++)
            resultado/=
getNumber(number, i);
        break;
    default:
        printf("Ingrese una opcion
valida!\n");
}
printf("El resultado es %f\n",
resultado);
}
```

Compresión de datos
Allan Jair Escamilla Hernández

Diagrama IPO



Conclusión

Al final de este ejercicio puedo concluir que los operadores a nivel de bits que existen en c, nos pueden ser sumamente útiles si es que queremos ahorrar memoria a la hora de escribir código, ya que podríamos disminuir el uso de memoria de nuestros algoritmos y hacer que sean más eficientes. .