Guía 3

Rodrigo Estay Muñoz

Usuario de Timus: RodrigoEstay

ID de Timus: 247803OV

Problema 1:

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

/* Programa hecho sin ayuda. */

/* Validamos la entrada y luego ocupamos %c para caracter, %u para entero
sin signo, %o para octal y %x para hexadecimal. */

int main(){
    int num;
    do{
        printf("Ingrese un numero entre 0 y 255:\n");
        scanf("%d", &num);
    }while(num<0 || num>255);
    printf("Caracter:\t\t%c\n", num);
    printf("Entero sin signo:\t%u\n", num);
    printf("Numero octal:\t\t%o\n", num);
    printf("Numero hexadecimal:\t%x\n", num);
    return 0;
}
```

Problema 2:

29:
Binario: 11101
Octal: 35
Hexadecimal: 1D

131:
Binario: 10000011
Octal: 203
Hexadecimal: 83

2048:
Binario: 100000000000
Octal: 4000
Hexadecimal: 800

121817:
Binario: 11101101111011001
Octal: 355731
Hexadecimal: 1DBD9

4000000001:
Binario: 11101110011010110010100000000001
Octal: 35632624001
Hexadecimal: EE6B2801

Problema 3:

```c
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

/* Programa hecho sin ayuda. */

int main (){
    int n, i, j, exp=0;
    scanf("%d",&n);
    double num[n], sum=0, temp, prom, desvM=0, var=0;
    for(i=0;i<n;++i){
        scanf("%lf", &num[i]);
    }
    /* Ordenamos el arreglo de mayor a menor. (mientras sumamos todos los numeros)*/
    for(i=0;i<n;++i){
        for(j=i;j<n;++j){
            if(num[i]<num[j]){
                temp=num[i];
                num[i]=num[j];
                num[j]=temp;
            }
        }
        sum+=num[i];
    }
    /* Calculamos el promedio, y las sumas de la desviacion media y varianza sin
    dividir, luego los dividimos. */
    prom=sum/n;
    for(i=0;i<n;++i){
        desvM=desvM+fabs(num[i]-prom);
        var=var+pow((num[i]-prom),2);
    }
    }
    desvM/=n;
    var/=n;
    /* Hay que notar que la desviacion estandar es la raiz cuadrada de la varianza. */
    printf("El minimo: %e\n", num[0]);
    printf("El maximo: %+10lf\n", num[n-1]);
    printf("La media: %10.2lf\n", prom);
    printf("La mediana: %.4lf\n", (n%2==0)?((num[n/2]+num[n/2-1])/2):(num[n/2]));
    printf("La desviacion media: %.2lf\n", desvM);
    printf("La desviacion estandar: %.3lf\n", sqrt(var));
    printf("La varianza: %lf\n", var);
    return 0;
}
```

Problema 4:

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

/* Programa hecho sin ayuda. */

/* Revisamos en cada intervalo (el de minusculas, mayusculas
y numeros) si se encuentra en alguno de esos intervalos, imprimimos
de cual se trata. */

int main(){
    char c;
    printf("Ingrese un caracter:\n");
    scanf("%c", &c);
    if(c>='a' && c<='z'){
        printf("minuscula\n");
    }
    else if(c>='A' && c<='Z'){
        printf("mayuscula\n");
    }
    else if(c>='0' && c<='9'){
        printf("numero\n");
    }
    else{
        printf("otro\n");
    }
    return 0;
}
```

Problema 5:

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

/* Programa hecho sin ayuda, solo con la ayuda de la calculadora de windows(en modo programador).

EXPLICACION DE LOS NUMEROS OCUPADOS(a la derecha estan sus valores en bits en 4 bytes):
255         = 00000000 00000000 00000000 11111111
65280       = 00000000 00000000 11111111 00000000
16711680    = 00000000 11111111 00000000 00000000
4278190080  = 11111111 00000000 00000000 00000000

De esta manera solo comparamos el caracter escaneado con cada uno de los bytes del numero
escaneado, ya que al aplicar el operador & al numero escaneado con cada uno de estos numeros
anteriores y luego mover los bits para eliminar los bytes de solo 0, obtendremos un numero
correspondiente a cada byte del numero escaneado (num). si es contenido en alguno de estos
imprimimos que es contenido.*/

int main(){
    int num;
    char c;
    printf("Ingrese un entero:\n");
    scanf("%d", &num);
    getchar();
    printf("Ingrese un caracter:\n");
    scanf("%c", &c);
    if(c==(num&255)){
        printf("Contenido\n");
    }
    else if(c==((num&65280)>>8)){
        printf("Contenido\n");
    }
    else if(c==((num&16711680)>>16)){
        printf("Contenido\n");
    }
    else if(c==((num&4278190080)>>24)){
        printf("Contenido\n");
    }

    /* En el caso que no esta contenido, hacemos una suma del caracter "c" movido 3, 2, 1
    y 0 bytes hacia la izquierda, lo cual es equivalente a multiplicarlo por 2 por n veces,
    donde n serian los numeros anteriores. Al terminar este proceso se concatenara 4 veces
    el caracter "c".
    Se hace casting a int a cada "c" ocupada ya que en char no se pueden almacenar mas de
     1 byte, asi no se podrian realizar estas sumas. */

    else{
        int conc=((int)c<<24)+((int)c<<16)+((int)c<<8)+c;
        printf("Al concatenar a 4 veces se obtiene:\n%d\n", conc);
    }
    return 0;
}
```

Problema 6:

```c
#include <stdio.h>
#include <stdlib.h>
/* Programa hecho sin ayuda.

Escaneamos el numero de punto flotante como un string, ya que asi es mas simple,
luego mientras avanzamos por el string, y para cada numero lo sumamos a "sum", si
encontramos un caracter que no es un numero o mas de un punto, significa que no se
ingreso un numero de punto flotante correcto, asi que mandamos un error. */

int main(){
    char num[250], isPoint=1;
    int sum=0, i;
    scanf("%s", num);
    for(i=0;num[i]!='\0';++i){
        if(num[i]>='0' && num[i]<='9'){
            sum+=(num[i]-'0');
        }
        else{
            if(num[i]!='.' || !isPoint){
                printf("Numero ingresado invalido.\n");
                return 0;
            }
            isPoint--;
        }
    }
    printf("%d\n", sum);
    return 0;
}
```

Problema 7:

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

/* Programa hecho sin ayuda.

Escaneamos cada uno como string, luego avanzamos los string desde la posicion 2 ya que
desde alli empieza el numero, luego vamos asignandole al numero correspondiente el
numero que era anteriormente mas el valor del caracter contenido en la posicion en la
que se va del string menos el valor del caracter '0', (notese que el algoritmo funciona
ya que los N1 y N2 se incicializan con valor inicial 0) de esta manera se obtiene el
numero de cada string, luego se imprime el menor. */

int main(){
    char s1[100], s2[100];
    int N1=0, N2=0, i;
    scanf("%s%s", s1, s2);
    for(i=2;s1[i]!='\0';++i){
        N1=N1*10+s1[i]-'0';
    }
    for(i=2;s2[i]!='\0';++i){
        N2=N2*10+s2[i]-'0';
    }
    printf("El menor es:\n%d\n", (N1>N2)?N2:N1);
    return 0;
}
```

Problema 8:

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

/* Programa hecho sin ayuda.

NOTA: el orden de los bytes que denote es de iquierda a derecha.

ya que char solo aguanta un byte de memoria, al asignarle un numero de mayor
cantidad de bytes, este char solo almacenara el ultimo byte del numero, asi
vamos imrpimiendo lo pedido en cada byte, ya que nos vamos moviendo de a un
byte y haciendo casting a char para que nos diga el ultimo byte. */

int main(){
    int num;
    printf("Ingrese un numero entero:\n");
    scanf("%d", &num);
    printf("primer byte:\t%c %d\n", (char)(num>>24), (char)(num>>24));
    printf("segundo byte:\t%c %d\n", (char)(num>>16), (char)(num>>16));
    printf("tercer byte:\t%c %d\n", (char)(num>>8), (char)(num>>8));
    printf("cuarto byte:\t%c %d\n", (char)(num), (char)(num));
    return 0;
}
```

Problema 9:

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

/* Programa hecho sin ayuda.

Para revisar si el primer decimal es mayor a 4, multiplicamos el flotante por 10 y luego
le hacemos casting a long long int (ya que el numero flotante puede ser mayor a lo que
puede almacenar int) y de esta manera nos desacemos de los demas decimales, le aplicamos
modulo con 10, y obtendremos el valor del primer decimal, si este es menor a 4 imprimimos
el entero de "num", si no le sumamos uno. */

int main(){
    float num;
    scanf("%f", &num);
    printf("Redondeado al entero: %lld\n", ((long long int)(num*10)%10>4)?(long long int)num+1:(long long int)num);
    return 0;
}
```

Problema 10:

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

/* Programa hecho sin ayuda.

Revisamos si cada caracter del string es una letra, si esta es mayuscula
le restamos la diferencia del valor de A y a lo cual lo dejara siendo la
misma letra pero en minuscula, hacemos lo mismo con las letras minusculas
pero en vez de restar la diferencia la sumamos.

Notamos que este programa solo funciona con palabras sin espacio en blanco
si se quiere que funcione con espacios en blanco cambiamos el %s de la linea
20 por un %[^EOF], que escanea hasta el final del archivo. (pero no se si se
puede usar) */

int main(){
    char word[1000];
    int i;
    printf("Ingrese una cadena de caracteres:\n");
    scanf("%s", word);
    for(i=0;word[i]!='\0';++i){
        if(word[i]>='A' && word[i]<='Z'){
            word[i]-=('A'-'a');
        }
        else if(word[i]>='a' && word[i]<='z'){
            word[i]+=('A'-'a');
        }
    }
    printf("%s\n", word);
    return 0;
}
```

Problema 11:

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

/* Programa hecho sin ayuda

Primero escaneamos hasta el salto de linea y luego revisamos si se encuentran
caracteres en el intervalo de mayusculas o minusculas, si es que no se cumple
ninguno de esos casos, entonces analizamos el caracter anterior, siempre y cuando
"i" sea distinto de 0 (para no ingresar a un lugar de memoria que no deberiamos),
asi si es que el caracter anterior era una letra, se trataba de una palabra.
Hacemos lo mismo cuando se llega al cambio de linea (cuando se termina el ciclo). */

int main(){
    char line[1000];
    int i, mayus=0, minus=0, words=0;
    scanf("%[^\n]", line);
    for(i=0;line[i]!='\0';++i){
        if(line[i]>='A' && line[i]<='Z'){
            ++mayus;
        }
        else if(line[i]>='a' && line[i]<='z'){
            ++minus;
        }
        else if(i!=0){
            if((line[i-1]>='a' && line[i-1]<='z') || (line[i-1]>='A' && line[i-1]<='Z')){
                ++words;
            }
        }
    }
    if((line[i-1]>='a' && line[i-1]<='z') || (line[i-1]>='A' && line[i-1]<='Z')){
        ++words;
    }
```

```c
    if((line[i-1]>='a' && line[i-1]<='z') || (line[i-1]>='A' && line[i-1]<='Z')){
        ++words;
    }

    /* Notemos que el numero de caracteres scaneados es igual al "i" sin contar el salto de linea. */

    printf("Mayusculas:\t%d\nMinusculas:\t%d\nPalabras:\t%d\nCaracteres\t%d\n", mayus, minus, words, i);
    return 0;
}
```

Problema 12:

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

/* Programa hecho sin ayuda.

Al declarar las variables notemos el largo maximo de las palabras es 200 y el numero maximo
de paabras es 400. */

int main(){
    char words[400][200], c;
    int lines=-2, wordLetter=0, wordPos=0, i, j, repeat;
    while(1){
        c=getchar(); // Obtenemos caracter a caracter.
        if(!lines){ // Si lines llega a 0 terminamos.
            break;
        }

        /* Si se encuentran letras mayusculas las pasamos a minusculas, de esta manera
        mas adelante no importaran si se hayan escrito con mayusculas o minusculas las
        palabras, aun asi se compararan correctamente. Notemos que considero palabras
        que solo estan escritas con letras de nuestro alfabeto (sin considerar la ñ). */

        if(c>='A' && c<='Z'){
            words[wordPos][wordLetter]=c-('A'-'a');
            ++wordLetter;
        }
        else if(c>='a' && c<='z'){
            words[wordPos][wordLetter]=c;
            ++wordLetter;
        }

        }

        /* Si es que no se trata de una letra, podria tratarse del cambio de linea
        si es asi aumentamos el contador del cambio de linea, si este contador llega
        a 0 se termina el ciclo, asi solo se escanearan dos lineas (ya que al principio
        declaramos "lines" con el valor de -2). Luego si la letra en la que se iba es la
        primera, significa que el caracter escaneado se escaneo antes que cualquier
        palabra, asi evitamos de crear una palabra con solo un vector nulo poniendo un
        continue. Luego, si no se cumplen los casos anteriores, terminamos la palabra con
        un vector nulo y seguimos con la siguiente palabra, desde la primera letra. */

        else{
            if(c=='\n'){
                ++lines;
            }
            if(wordLetter==0){
                continue;
            }
            words[wordPos][wordLetter]='\0';
            wordLetter=0;
            ++wordPos;
        }
    }

```

```
    }

    /* Luego, para cada palabra almacenada, la comparamos con todas las palabras almacenadas
    antes de esta, si se encuentra una palabra que es igual, decimos que se repite, y luego
    si es que la palabra no se repite, esta se imprime. */

    for(i=0;i<wordPos;++i){
        repeat=0;
        for(j=0;j<i;++j){
            if(!strcmp(words[j], words[i])){
                repeat=1;
                break;
            }
        }
        if(!repeat){
            printf("%s\n", words[i]);
        }
    }
    return 0;
}
```

Problema 13:

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

/* Programa hecho sin ayuda. */

int main(){
    char line[10000], line2[5000], temp;
    int n, i, j;
    scanf("%[^\n]", line);
    getchar(); // El getcgar es para saltarnos el '\n' al final de la primera linea.
    scanf("%[^\n]", line2);

    /* Obtenemos el largo de la linea 2 y luego vamos reemplazando el ultimo valor de
    la linea con el primero, el segundo con el penultimo, y asi hasta invertir la linea
    completamente, luego hacemos lo mismo con la linea 1. */

    n=strlen(line2);
    for(i=0;i<n/2;++i){
        temp=line2[i];
        line2[i]=line2[n-i-1];
        line2[n-i-1]=temp;
    }
    n=strlen(line);
    for(i=0;i<n/2;++i){
        temp=line[i];
        line[i]=line[n-i-1];
        line[n-i-1]=temp;
    }

    }

    /* Ya que el "n" almacena el valor del largo de la linea 1, lo ocupamos para empezar a
    concatenar las dos lineas, desde el final de la linea 1 empezamos a poner los valores de
    la linea 2, cuando termina de poner valores, asignamos a la siguiente posicion el vector
    nulo para terminar el string. */

    for(i=n,j=0;line2[j]!='\0';++i,++j){
        line[i]=line2[j];
    }
    line[i]='\0';

    // Recorremos el arreglo resultante e imprimimos las vocales tanto mayusculas como minusculas.

    for(i=0;line[i]!='\0';++i){
        if(line[i]=='A' || line[i]=='E' || line[i]=='I' || line[i]=='O' || line[i]=='U' ||
           line[i]=='a' || line[i]=='e' || line[i]=='i' || line[i]=='o' || line[i]=='u'){
            printf("%c", line[i]);
        }
    }
    printf("\n");
    return 0;
}
```

Problema 14:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/* Programa hecho sin ayuda. */

/* Primero inicializamos un arreglo con el conteo de cada tipo de pinguino, luego dos arreglos,
uno almacenara siempre la palabra "penguin", y el otro el tipo del pinguino para cada caso, y el
ultimo arreglo almacena los 3 nombres de los tipos de penguinos.
Luego para los "n" casos, escaneamos dos strings, el que nos interesa es el tipo, asi que lo
comparamos con cada tipo, si es igual a alguno aumentamos su respectivo contador.
Luego buscamos el valor maximo y lo imprimimos. */

int main(){
    int n, countType[3]={0,0,0};
    scanf("%d", &n);
    char penguin[10], type[15], types[3][14]={"Little", "Macaroni", "Emperor"};
    while(n--){
        scanf("%s %s", type, penguin);
        if(!strcmp(type,types[0])){
            countType[0]+=1;
        }
        else if(!strcmp(type,types[1])){
            countType[1]+=1;
        }
        else if(!strcmp(type,types[2])){
            countType[2]+=1;
        }
    }
    int max=0;
    for(n=0;n<3;++n){
        if(countType[max]<countType[n]){
```

```c
        }
    }
    int max=0;
    for(n=0;n<3;++n){
        if(countType[max]<countType[n]){
            max=n;
        }
    }
    printf("%s Penguin\n", types[max]);
    return 0;
}
```

Problema 15:

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

/* Programa hecho sin ayuda. */

/* almacenamos la cantidad total de invitados en "total", al principio
le sumamos los "n" invitados y los que se van a casar (2), luego para cada
string escaneado, le buscamos si tiene un "+", lo cual significa que trae
un invitado, asi que sumamos 1 a "total", luego si son exactamente 13,
sumamos uno mas por lo especificado en el problema. */

int main(){
    char guest[26];
    int n, total=0, i;
    scanf("%d", &n);
    total=total+n+2;
    while(n--){
        scanf("%s", guest);
        for(i=0;guest[i]!='\0';++i){
            if(guest[i]=='+'){
                ++total;
            }
        }
    }
    if(total==13){
        ++total;
    }
    printf("%d\n",total*100); // Lo multiplicamos por 100 por el precio por dummy.
    return 0;
}
```

Problema 16:

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/* Me han ayudado a interpretar las formulas de este problema.

Podemos observar que las posiciones de los 1 estan en:
1,2,4,7,11,... = 1+(0),1+(0+1),1+(0+1+2),1+(0+1+2+3),1+(0+1+2+3+4),...
lo cual es equivalente a 1+sum(i) con i desde 0 hasta n, donde n es la posicion
del numero "k" en el orden anterior, luego por definicion de esa sumatoria,
y algo de algebra tenemos:

1+sum(i)=1+n*(n+1)/2=(n^2+n+2)/2

entonces "k" debe cumplir que para algun n se de:

k=(n^2+n+2)/2=>2k=n^2+n+2=>0=n^2+n+(2-2k)=>n=(-1+-sqrt(1-4(2-2k)))/2
=>n=(-1+-sqrt(-7+8k))/2

Luego, como dijimos anteriormente, n es la posicion del "k" ingresado, entonces
mientras n sea entero significa que "k" se encuentra en la primera secuencia enunciada
entonces hay un 1 en la posicion "k", con mas algebra tenemos:

2n=-1+-sqrt(-7+8k)=>2n+1=+-sqrt(-7+8k)

Entonces observamos que a la izquierda de la ultima igualdad deberian presentarse solo enteros,
entonces para que se cumpla que n sea entero se debe cumplir que sqrt(-7+8k) sea entero, de esta
manera, para revisar que lo anterior es entero, hacemos que sea una variable "form", y luego
revisamos si la resta de esa raiz menos la parte entera de esa raiz es cero, si lo es, significa
que la raiz es un cuadrado perfecto, y de esta manera, un entero. */

int main(){
```

```c
int main(){
    long long int N, k, i;
    double form;
    scanf("%lld", &N);
    while(N--){
        scanf("%lld", &k);
        form=(sqrt(-7.0+8.0*(double)k));
        if(form-(int)form==0){
            printf("1");
        }
        else{
            printf("0");
        }
        if(N!=0){
            printf(" ");
        }
        else{
            printf("\n");
        }
    }
    return 0;
}
```

En Timus:

7885486	00:40:22 18 May 2018	RodrigoEstay	1209. 1, 10, 100, 1000...	GCC 7.1	Accepted		0.031	196 KB
7881532	05:33:37 14 May 2018	RodrigoEstay	2100. Wedding Dinner	GCC 7.1	Accepted		0.015	212 KB
7872914	22:33:09 4 May 2018	RodrigoEstay	1585. Penguins	GCC 7.1	Accepted		0.001	208 KB