ESTUDIOS
GENERALES
CIENCIAS

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

FUNDAMENTOS DE PROGRAMACIÓN
LABORATORIO 4
PROPUESTAS DE SOLUCIÓN
SEMESTRE ACADÉMICO 2021-2

Horarios: Todos los horarios

Elaborado por Dra. Layla Hirsh

INDICACIONES:

- Debe utilizar variables descriptivas, comentarios y mensajes descriptivos.
- El orden y la eficiencia de su implementación serán considerados en la calificación.

RESULTADOS ESPERADOS:

- Al finalizar la sesión, el alumno construirá programas usando programación modular usando paso de parámetro por valor y simulando el paso de parámetros por referencia

CONSIDERACIONES:

- La solución presentada para cada problema corresponde a una propuesta de solución por parte del autor.
- En programación pueden existir muchas soluciones para un mismo problema pero debe cumplir con todo lo solicitado, incluyendo las restricciones brindadas.

Desarrolle los siguientes problemas en lenguaje C:

1. Número Katadrome

Se considera que un número es un katadrome en una base dada b (a menudo 10 o 16) si sus dígitos están en orden estrictamente decreciente en esa base.

Por ejemplo, 43210, 76521 y 9630 son todos Katadromos en base 10.

Se considera además que un número es un plaindrome en una base dada b (a menudo 10 o 16) si sus dígitos están en orden no decreciente en esa base. Por ejemplo, 1234, 2222, 25667 y 2468 son todos plaindrome en base 10.

Se le pide elaborar un programa en lenguaje C, con al menos cuatro módulos incluído el principal que dado un número en base 10 ingresado por el usuario, calcule la cantidad de dígitos del número usando la fórmula del logaritmo, y haciendo uso de este valor verifique si el número ingresado es Katadrome y si es plaindrome, recorriendo dicho número de izquierda a derecha e identifique la parte del número donde se cumple el decremento interno del número en el caso del katadrome, para el plaindrome no es necesario identificar el incremento interno.

Por ejemplo en el caso del del número 5432 la verificación de katadrome dará 1, en la verificación plaindrome dará 0, la parte del número donde se cumple el decremento sería 5432. En el caso de 765243 la verificación dará 0 y la parte será 765. Finalmente, debe mostrar el mensaje: **El número es Katadrome** o el mensaje: **El número no es Katadrome**, según corresponda. Adicionalmente deberá mostrar el número que cumple el decremento. Ademas mostrar el mensaje: **El número es plaindrome** o el mensaje **El número no es plaindrome**.

Para resolver este problema, deberá implementar al menos cuatro módulos incluído el principal, según el siguiente detalle:

- Un módulo que no modifique ni devuelva valores.
- Un módulo que modifique al menos dos valores y que internamente use una estructura iterativa y al menos una estructura selectiva (paso por referencia).
- Un módulo que devuelva un valor.

Nota:

- Asuma que el usuario siempre ingresa un valor adecuado mayor a 0.
- La cantidad de cifras de un número cualquiera se obtiene sumando 1 a la parte entera de su logaritmo decimal.
- En C, la función log10 de la biblioteca math.h permite calcular el logaritmo decimal de un número.

Casos de prueba:

```
Ingrese el número a evaluar: 87645
El número no es Katadrome
Se cumple el decremento en el número 8764
El número no es plaindrome
```

```
Ingrese el número a evaluar: 875643
El número no es Katadrome
Se cumple el decremento en el número 875
El número no es plaindrome
```

```
Ingrese el número a evaluar: 12345
El número no es Katadrome
Se cumple el decremento en el número 1
El número es plaindrome
```

```
Ingrese el número a evaluar: 9876
El número es Katadrome
Se cumple el decremento en el número 9876
El número no es plaindrome
```

Programa 1: Propuesta de solución - Número Katadrome

```
#include<stdio.h>
#include<math.h>
void evaluarSiEsKatadrome(int num,int cantDig, int*resp,int*nnum);

void mostrarMensaje(int esKatadromes);
int CalcularCantidadDigitos(int num);

int main(){
        char evaluacion;
        int num,cantDig,esKatadromes,nnum;
        printf("Ingrese el número a evaluar: ");
        scanf(" %d",&num);
        cantDig=log10(num)+1;
        evaluarSiEsKatadrome(num,cantDig,&esKatadromes,&nnum);
        mostrarMensaje(esKatadromes);
        printf("\nSe cumple el decremento en el número  %d",nnum);
        return 0;
}

```

```
void mostrarMensaje(int esKatadromes){
	if(esKatadromes){
		printf("\nEl número es Katadrome ");
	}
	else{
		printf("\nEl número no es Katadrome ");
	}
}

int evaluarSiEspalindome(int num,int cantDig){
	int dig,norig=num,digant=num/pow(10,cantDig-1),continua=1,resp=1;
	num=digant;
	num=num %(int)pow(10,cantDig-1);
	cantDig--;
	while(cantDig>0){
		dig=num/(int)pow(10,cantDig-1);
		num=num %(int)pow(10,cantDig-1);
		if(dig>digant){
			resp=resp&&1;
		}
		else{
			resp=resp&&0;
		}
		cantDig--;
	}
	return resp;
}
void evaluarSiEsKatadrome(int num,int cantDig, int*resp,int*nnum){
	int dig,norig=num,digant=num/pow(10,cantDig-1),continua=1;
	*nnum=digant;
	num=num %(int)pow(10,cantDig-1);
	cantDig--;
	while(cantDig>0){
		dig=num/(int)pow(10,cantDig-1);
		num=num %(int)pow(10,cantDig-1);
		if(dig<digant && continua){
			*nnum=(*nnum)*10+dig;
		}
		else{
			continua=0;
		}
		digant=dig;
		cantDig--;
	}
	if(*nnum==norig){
		*resp=1;
	}
	else{
		*resp=0;
	}
}
```

2. Número interprimo

Se considera a un número como un número interprimo cuando es el promedio de dos primos consecutivos, es decir, está a la misma distancia del primo anterior y del primo siguiente.

Por ejemplo, 21 es un interprimo ya que es el promedio de los dos primos consecutivos 19 y 23.

Se le pide elaborar un programa en lenguaje C, con al menos cuatro módulos incluído el principal que dado dos números ingresados por el usuario, verifique que ambos sean primos, calcule el promedio de ambos, verifique que sean equidistantes, verifique que no existe dentro del rango otro número que sea divisible entre 2, 3, 5 ni 7 y que la raiz del número no sea exacta, finalmente debe verificar si el número es interprimo y debe mostrar los mensajes

correspondientes.

Para resolver este problema, deberá implementar al menos cuatro módulos incluído el principal, según el siguiente detalle:

- Un módulo que no modifique ni devuelva valores.
- Un módulo que modifique al menos dos valores (paso por referencia) y que utilice otro módulo.
- Un módulo que devuelva un valor y que internamente use una estructura iterativa y al menos una estructura selectiva.

Nota: Asuma que el usuario siempre ingresa valores adecuados mayores a 0. Además que en lugar de evaluar que sean primos consecutivos, bastara con hacer la verificación de que no sea divisible entre 2, 3, 5 ni 7 y que la raiz del número no sea exacta

Casos de prueba:

```
Ingrese dos números primos consecutivos: 19 23
El número es interprimo
```

```
Ingrese dos números primos consecutivos: 20 23
El número no es interprimo
El número 20 es primo 0
El número 23 es primo 1
Los números son equidistantes 1
Existe un primo identificado 0
```

```
Ingrese dos números primos consecutivos: 5 11
El número es interpimo
```

Programa 2: Propuesta de solución - Número Interprimo

```
#include<stdio.h>
#include<math.h>
int ExisteotroPrimo(int primo1,int primo2);
void sonprimos(int primo1,int primo2,int *esPrimo1,int*esPrimo2);
int esPrimo(int num);
void sonprimos(int primo1,int primo2,int *esPrimo1,int*esPrimo2);
int main(){
        char evaluacion;
        double promedio;
        int primo1,equidistante,primo2,esPrimo1,existeotro,esPrimo2;
        printf("Ingrese dos números primos consecutivos: ");
        scanf("%d %d",&primo1,&primo2);
        sonprimos(primo1,primo2,&esPrimo1,&esPrimo2);
        promedio=(double)(primo1+primo2)/2;
        equidistante=fabs(primo1-promedio)==primo2-promedio;
        existeotro=ExisteotroPrimo(primo1,primo2);

        if(esPrimo1 && esPrimo2 && equidistante && !existeotro){
                printf("El número es interprimo\n");
        }
        else{
                printf("El número no es interprimo\n");
                printf("El número %d es primo %d\n",primo1,esPrimo1);
                printf("El número %d es primo %d\n",primo2,esPrimo2);
                printf("Los números son equidistantes %d\n",equidistante);
                printf("Existe otro primo identificado %d\n",existeotro);
        }
        return 0;
```

```
}
int ExisteotroPrimo(int primo1,int primo2){
	int esexacta,cont=0,resp;
	primo1++;
	while(primo1<primo2){
		esexacta=(sqrt(primo1)-(int)sqrt(primo1))==0;
		printf(" %d",primo1);
		if(primo1 %2!=0 && primo1 %3!=0 & primo1 %5!=0 && primo1 %7!=0 && !esexacta){
			cont++;
		}
		primo1++;
	}
	if(cont==0){
		resp=0;
	}
	else{
		resp=1;
	}
	return resp;
}
void sonprimos(int primo1,int primo2,int *esPrimo1,int*esPrimo2){
	*esPrimo1=esPrimo(primo1);
	*esPrimo2=esPrimo(primo2);
}
int esPrimo(int num){
	int resp=0,i=1,cont=0;
	while(i<=num){
		if(num %i==0){
			cont++;
		}
		i++;
	}
	if(cont==2){
		resp=1;
	}
	return resp;
}
```

3. Número de Moran

Se considera que un número n es un número de Moran si n dividido por la suma de sus dígitos da un número primo. Por ejemplo, 111 es un número de Moran porque 111 / (1 + 1 + 1) = 37 y 37 es un número primo.

Los números de Moran son un subconjunto de los números de Harshad.

Se le pide elaborar un programa en lenguaje C, con al menos cuatro módulos incluído el principal que dado un número ingresado por el usuario, calcule la cantidad de dígitos que posee el número usando la formula del logaritmo, y haciendo uso de dicho valor, calcule la suma de sus dígitos, luego calcule el resultado de dividir dicho número entre dicha suma, evalue si es o no Moran y finalmente muestre un mensaje que diga si es o no Moran y otro en el que se muestren tanto la suma de dígitos como el resultado de la división (como se muestra en los casos de prueba).

Para resolver este problema, deberá implementar al menos cuatro módulos incluído el principal, según el siguiente detalle:

- Un módulo que no modifique ni devuelva valores.
- Un módulo que modifique al menos dos valores (paso por referencia) y que internamente use una estructura selectiva y que llame a otro módulo.
- Un módulo que devuelva un valor.

Nota:

- Asuma que el usuario siempre ingresa un valor adecuado mayor a 0 con una cantidad variable de dígitos.
- La cantidad de cifras de un número cualquiera se obtiene sumando 1 a la parte entera de su logaritmo decimal.
- En C, la función log10 de la biblioteca math.h permite calcular el logaritmo decimal de un número.

Casos de prueba:

```
Ingrese el número a evaluar: 156
El número es Moran
La suma de dígitos es 12 y el resultado de la división es 13
```

```
Ingrese el número a evaluar: 54
El número no es Moran
La suma de dígitos es 9 y el resultado de la división es 6
```

```
Ingrese el número a evaluar: 11824
El número es Moran
La suma de dígitos es 16 y el resultado de la división es 739
```

```
Ingrese el número a evaluar: 201
El número es Moran
La suma de dígitos es 3 y el resultado de la división es 67
```

Programa 3: Propuesta de solución - Número de Moran

```c
#include<stdio.h>
#include<stdio.h>
int main(){
	char evaluacion;double div;
	int num,cantDig,esMoran,sum;
	printf("Ingrese el número a evaluar: ");
	scanf(" %d",&num);

	cant=log10(num)+1;
	evaluarSiEsMoran(num,&esMoran,&sum,&div,cant);
	if(esMoran){
		printf("El número es Moran ");
	}
	else{
		printf("El número no es Moran ");
	}
	printf("\nLa suma de dígitos es  %d y el resultado de la división es  %.0lf",sum,div);
	return 0;
}
void evaluarSiEsMoran(int num,int *resp,int*sum,double*div,int cant){
	int esPrim;
	*sum=CalcularSumaDigitos(num,cant);
	*div=(double)num/(*sum);
	esPrim=esPrimo(div);

	if(esPrim){
		*resp=1;
	}
	else{
		*resp=0;
```

```
        }
}

int esPrimo(double numm){
        int resp=0,i=1,cont=0,num=numm;
        while(i<=num){
                if(num %i==0){
                        cont++;
                }
                i++;
        }
        if(cont==2){
                resp=1;
        }
        return resp;
}

int CalcularSumaDigitos(int num,int cant){
        int suma=0,dig;
        while(num>0){
                dig=num %10;

                num=num/10;
                suma=suma+dig;
        }
        return suma;
}
```

4. Número de Canadá

Se considera que un número n es un números de Canadá si la suma de los cuadrados de los dígitos de n es igual a la suma de los divisores no triviales de n, es decir, $sigma(n) - n - 1$. Por ejemplo, 581, cuyos divisores son 1, 7, 83 y 581, es un número de Canadá porque $5^2 + 8^2 + 1^2 = 7 + 83$.

El nombre de estos números se debe a que fueron definidos por algunos matemáticos de la Universidad de Manitoba para celebrar el 125 aniversario de Canadá.

Se le pide elaborar un programa en lenguaje C, con al menos cuatro módulos incluído el principal que dado un número ingresado por el usuario y la cantidad de dígitos que este posee, calcule la suma de sus dígitos al cuadrado, calcule la suma de los divisores no triviales de dicho número, evalue si es o no un número de Canadá y finalmente muestre un mensaje que diga si es o no de Canadá y otro en el que se muestren tanto la suma de dígitos al cuadrado como el resultado de la suma de sus divisores no triviales (como se muestra en los casos de prueba).

Para resolver este problema, deberá implementar al menos cuatro módulos incluído el principal, según el siguiente detalle:

- Un módulo que no modifique ni devuelva valores.
- Un módulo que modifique al menos dos valores (paso por referencia) y que internamente use una estructura selectiva y que llame al menos otro módulo.
- Un módulo que devuelva un valor y que use internamente una estructura iterativa.

Nota:

- Asuma que el usuario siempre ingresa un valor adecuado mayor a 0 con una cantidad variable de dígitos.
- Asuma que la cantidad de dígitos ingresado concuerda exactamente con la cantidad de dígitos del número.

Casos de prueba:

```
Ingrese el número a evaluar  y la cantidad de dígitos: 581 3
El número es Canada
La suma de los cuadrados de sus dígitos es 90 y de sus divisores es 90
```

```
Ingrese el número a evaluar  y la cantidad de dígitos: 8549 4
El número es Canada
La suma de los cuadrados de sus dígitos es 186 y de sus divisores es 186
```

```
Ingrese el número a evaluar  y la cantidad de dígitos: 16999 5
El número es Canada
La suma de los cuadrados de sus dígitos es 280 y de sus divisores es 280
```

```
Ingrese el número a evaluar  y la cantidad de dígitos: 763 3
El número no es Canada
La suma de los cuadrados de sus dígitos es 94 y de sus divisores es 116
```

Programa 4: Propuesta de solución - Número de Cánada

```c
#include<stdio.h>
#include<stdio.h>
int main(){
	char evaluacion;int div;
	int num,cantDig,esCanada,sum;
	printf("Ingrese el número a evaluar: ");
	scanf(" %d",&num);
	evaluarSiEsCanada(num,&esCanada,&sum,&div,cantdig);
	if(esCanada){
		printf("El número es Canada ");
	}
	else{
		printf("El número no es Canada ");
	}
	printf("\nLa suma de los cuadrados de sus dígitos es  %d y de sus divisores es  %d",sum,div);
	return 0;
}
void evaluarSiEsCanada(int num,int *resp,int*sum,int*div,int cantdig){
	int es;
	*sum=CalcularSumaDigitos(num,cantdig);
	*div=SumaDivisores(num);
	es=*sum==*div;

	if(es){
		*resp=1;
	}
	else{
		*resp=0;
	}
}

int SumaDivisores(int num){
	int resp=0,suma=0,i=2,cont=0;
	while(i<num){
		if(num %i==0){
			suma=suma+i;
		}
		i++;
	}

	return suma;
```

```
42 }
43
44 int CalcularSumaDigitos(int num,int cantdig){
45         int suma=0,dig;
46         while(i<cantdig){
47                 dig=num %10;
48
49                 num=num/10;
50                 suma=suma+dig*dig;
51                 i++;
52         }
53         return suma;
54 }
```

No debe usar estructuras algorítmicas selectivas anidadas, anidamiento de selectivas, iterativas anidadas, iterativas con salida controlada o por centinela.