ESTUDIOS
GENERALES
CIENCIAS

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

FUNDAMENTOS DE PROGRAMACIÓN
LABORATORIO 7
PROPUESTAS DE SOLUCIÓN
SEMESTRE ACADÉMICO 2022-2

Horarios: Todos los horarios

Elaborado por Mag. David Allasi

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 comprenderá el funcionamiento de las estructuras algorítmicas iterativas anidadas.
- Al finalizar la sesión, el alumno diseñará algoritmos usando estructuras algorítmicas iterativas anidadas.

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 el siguiente pseudocódigo en PSeInt:

1. El misterioso número 6174 - Horarios: 0382, 0383, 0390

El número 6174 parece ser un número cualquiera, sin embargo, presenta una situación muy curiosa, la cual se origina al repetir una serie de pasos que se describen a continuación:

- Se debe ingresar un número de 4 dígitos que este formado por al menos dos dígitos diferentes, incluido el cero. Por ejemplo: 1234.
- Se debe organizar los dígitos del número original en orden ascendente, con lo cuál quedaría en 1234.
- Se debe organizar los dígitos del número original en orden descendente, con lo cuál quedaría en 4321.
- Se resta el número más pequeño del número más grande: 4321 - 1234 = 3087.
- Considerando al último resultado como el nuevo número, se deben repetir los pasos desde el item 2.

En algún momento, luego de repetir los pasos, se obtendrá como resultado de la resta del número más pequeño del número más grande el valor de 6174. Cuando se logre esto, ya no tiene sentido seguir repitiendo los pasos porque siempre encontraremos el mismo resultado, por lo que caeremos en un bucle infinito. A este número 6174 se le conoce como la constante de Kaprekar para números de 4 dígitos.

A continuación se muestra un ejemplo del proceso para el número 1234

```
Número Inicial: 1234
Número ordenado ascendente: 1234
Número ordenado descendente: 4321
Diferencia: 3087
Número Inicial: 3087
Número ordenado ascendente: 378 (el 0 a la izquierda no tiene valor)
Número ordenado descendente: 8730
Diferencia: 8352
Número Inicial: 8352
Número ordenado ascendente: 2358 (el 0 a la izquierda no tiene valor)
Número ordenado descendente: 8532
Diferencia: 6174
```

Como se puede ver en el ejemplo, se necesitaron 3 iteraciones para llegar al número 6174 que es la constante de Kaprekar.

Se le pide que diseñe un algoritmo, expresado en pseudocódigo, que solicite un número de 4 cifras y determine la cantidad de iteraciones que se deben realizar para llegar al número 6174 (Constante de Kaprekar). Para este caso, solo debe validar que el número realmente tenga 4 cifras, en caso que no sea así, no debe procesar nada y debe emitir un mensaje diciendo: El número no tiene 4 cifras. Si el número tiene 4 cifras, debe asumir que el número está formado por al menos dos dígitos diferentes, incluyendo al cero y debe procesar lo solicitado.

El proceso se debe realizar siguiendo estrictamente los pasos indicados, **considerando que para formar el número ordenado ascendente y número ordenado descendente debe utilizar estructuras iterativas**. Por ningún motivo debe obtener los 4 dígitos al mismo momento, ni guardar los valores de los mismos en 4 variables distintas que representen a los dígitos, por lo que siempre debe iterar para obtener los dígitos y a partir de ahí formar los números ordenados. Si obtiene los 4 dígitos al mismo tiempo (sin iterar) o los guarda en 4 variables distintas, se le colocará la nota de 00.

En los casos de prueba se indican los mensajes que debe usar dentro del pseudocódigo.

A continuación se presentan los casos de prueba:

```
Ingrese un número:
> 12345
El número no tiene 4 cifras
```

```
Ingrese un número:
> 1234
Número Inicial: 1234
Número ordenado ascendente: 1234
Número ordenado descendente: 4321
Diferencia: 3087
Número Inicial: 3087
Número ordenado ascendente: 378
Número ordenado descendente: 8730
Diferencia: 8352
Número Inicial: 8352
Número ordenado ascendente: 2358
Número ordenado descendente: 8532
Diferencia: 6174
Se necesitaron 3 iteraciones para llegar al número 6174.
```

```
Ingrese un número:
> 2005
Número Inicial: 2005
Número ordenado ascendente: 25
Número ordenado descendente: 5200
```

```
Diferencia: 5175
Número Inicial: 5175
Número ordenado ascendente: 1557
Número ordenado descendente: 7551
Diferencia: 5994
Número Inicial: 5994
Número ordenado ascendente: 4599
Número ordenado descendente: 9954
Diferencia: 5355
Número Inicial: 5355
Número ordenado ascendente: 3555
Número ordenado descendente: 5553
Diferencia: 1998
Número Inicial: 1998
Número ordenado ascendente: 1899
Número ordenado descendente: 9981
Diferencia: 8082
Número Inicial: 8082
Número ordenado ascendente: 288
Número ordenado descendente: 8820
Diferencia: 8532
Número Inicial: 8532
Número ordenado ascendente: 2358
Número ordenado descendente: 8532
Diferencia: 6174
Se necesitaron 7 iteraciones para llegar al número 6174.
```

Programa 1: Propuesta de solución - El misterioso número 6174

```
Algoritmo Propuesta1
    Escribir "Ingrese el número: "
    Leer num
    Si (num>=1000 y num<=9999) Entonces
        centinela <- Verdadero
        cantVeces <- 0
        Mientras (centinela) Hacer
            Escribir 'Número Inicial: ', num
            //Forma el número de forma decreciente.
            cantVeces <- cantVeces + 1
            copiaNum <- num
            cantCifras <- 1
            nuevoNumCrec <- 0
            Mientras (cantCifras<=4) Hacer
                digitoMenor <- 9
                copiaNum2 <- num
                Mientras (num>0) Hacer
                    digito <- num mod 10
                    num <- trunc(num/10)
                    Si (digito<digitoMenor) Entonces
                        digitoMenor <- digito
                    FinSi
                FinMientras
                nuevoNumCrec <- nuevoNumCrec*10 + digitoMenor
                num <- copiaNum2
                nuevoSinMenor <- 0
                exponente <- 0
                encontrado <- Falso
                Mientras (num>0) Hacer
                    digito <- num mod 10
                    num <- trunc(num/10)
                    Si (digito <> digitoMenor o (digito = digitoMenor y encontrado)) Entonces
                        nuevoSinMenor <- nuevoSinMenor + (10^exponente)*digito
                        exponente <- exponente + 1
                    SiNo
```

```
                                        encontrado <- verdadero
                                    FinSi
                                FinMientras
                                num <- nuevoSinMenor
                                cantCifras <- cantCifras + 1
                            FinMientras
                            Escribir "Número ordenado ascendente: ", nuevoNumCrec
                            //Calculamos el decreciente - Invertimos
                            copiaNum <- nuevoNumCrec
                            nuevoNumDec <- 0
                            cantCifras2 <- 1
                            Mientras (cantCifras2<=4) hacer
                                digito <- copiaNum mod 10
                                copiaNum <- trunc(copiaNum/10)
                                nuevoNumDec <- nuevoNumDec*10 + digito
                                cantCifras2 <- cantCifras2 + 1
                            FinMientras
                            Escribir "Número ordenado descendente: ", nuevoNumDec
                            diferencia <- nuevoNumDec - nuevoNumCrec
                            Escribir "Diferencia: ", diferencia
                            Si (diferencia=6174) entonces
                                centinela <- Falso
                            SiNo
                                num <- diferencia
                            FinSi
                    FinMientras
                    Escribir "Se necesitaron ", cantVeces, " iteraciones para llegar al número 6174"
            SiNo
                    Escribir "El número no tiene 4 cifras"
            FinSi
FinAlgoritmo
```

2. Los números de Kaprekar - Horarios: B301, 0380, 0381, 0391

La constante de Kaprekar no fue la única contribución de Dattatreya Ramchandra Kaprekar por los números a las matemáticas recreativas.

Entre su colección de ideas también está el número Kaprekar.

El número de Kaprekar es un número con la propiedad interesante de que si está al cuadrado, al sumar dos partes iguales del resultado, te da el número original. Esa operación es la operación de Kaprekar.

Por ejemplo:

$297^2 = 88209$ y del resultado 88209 podemos formar los grupos 88 + 209 = 297. Por lo tanto, el número 297 es considerado como un número de Kaprekar.

Algo que debes tener en cuenta, como en el ejemplo, es que al dividir el número cuyas partes vas a sumar, en caso el número de cifras sea impar, debes probar con dejar la parte más larga a la derecha (en el ejemplo, al dividir en dos 88209 te quedan obligatoriamente dos grupos: uno con dos dígitos y otro con tres, por eso, siguiendo las indicaciones, al separarlo quedan 88 y 209) o a la izquierda (en el ejemplo, al dividir en dos 88209 te quedan obligatoriamente dos grupos: uno con tres dígitos y otro con dos, por eso, siguiendo las indicaciones, al separarlo quedan 882 y 09). Cuando el número tenga una cantidad de cifras par, los dos números a dividir deben tener la misma cantidad de cifras.

Se le pide que diseñe un algoritmo, expresado en pseudocódigo que, solicite al usuario que ingrese la cantidad de cifras de los números que desea procesar. Este cantidad debe ser mayor que 0 y menor que 7, en caso de no cumplir debe enviar el siguiente mensaje: La cantidad de cifras no es correcta y terminar el algoritmo.

Si la cantidad de cifras es correcta, debe calcular todos los números de Kaprekar que existen en el rango de números que tienen la cantidad de cifras ingresada. Por ejemplo, si ingresa como cantidad de cifras 1, debe evaluar todo el

rango del 1 al 9. Si ingresará como cantidad de cifras 2, debe evaluar todo el rango del 10 al 99.

Al final de la evaluación debe imprimir los números que son considerados Kaprekar en dicho rango, la cantidad de números Kaprekar encontrados y el menor y mayor número Kaprekar encontrado dentro del rango.

OJO, si desea calcular la cantidad de cifras que tiene un número, debe realizar esté cálculo con alguna estuctura iterativa.

En los casos de prueba se indican los mensajes que debe usar dentro del pseudocódigo.

A continuación se presentan los casos de prueba:

```
Ingrese la cantidad de cifras:
> -6
La cantidad de cifras no es correcta.
```

```
Ingrese la cantidad de cifras:
> 7
La cantidad de cifras no es correcta.
```

```
Ingrese la cantidad de cifras:
> 3
Del rango 100 al 999 tenemos:
El número 100 es de Kaprekar porque 100000 genera los números 100 + 0 = 100
El número 297 es de Kaprekar porque 88209 genera los números 88 + 209 = 297
El número 703 es de Kaprekar porque 494209 genera los números 494 + 209 = 703
El número 999 es de Kaprekar porque 998001 genera los números 998 + 1 = 999
Existen 4 números de Kaprekar
El menor número de Kaprekar del rango es: 100
El mayor número de Kaprekar del rango es: 999
```

```
Ingrese la cantidad de cifras:
> 2
Del rango 10 al 99 tenemos:
El número 10 es de Kaprekar porque 100 genera los números 10 + 0 = 10
El número 45 es de Kaprekar porque 2025 genera los números 20 + 25 = 45
El número 55 es de Kaprekar porque 3025 genera los números 30 + 25 = 55
El número 99 es de Kaprekar porque 9801 genera los números 98 + 1 = 99
Existen 4 números de Kaprekar
El menor número de Kaprekar del rango es: 10
El mayor número de Kaprekar del rango es: 99
```

Programa 2: Propuesta de solución - Los números de Kaprekar

```
1  Algoritmo Propuesta2
2          Escribir "Ingrese la cantidad de cifras: "
3          Leer cantCifras
4          Si (cantCifras>0 y cantCifras<7) Entonces
5                  inicio <- 1*10^(cantCifras-1)
6                  final <- 1*10^(cantCifras)
7                  num <- inicio
8                  contador <- 0
9                  menorKaprekar <- final
10                 mayorKaprekar <- inicio
11                 Escribir "Del rango ", inicio, " al ", final-1, " tenemos: "
12                 Mientras (num<final) hacer
13                         numEvaluar <- num*num
14                         //Calcular la cantidad de cifras que tiene el número numEvaluar
15                         copiaNum <- numEvaluar
16                         centinela <- Verdadero
17                         cantCifras <- 0
```

```
                    Mientras (centinela) hacer
                        cantCifras <− cantCifras + 1
                        numEvaluar <− trunc(numEvaluar/10)
                        Si (numEvaluar=0) Entonces
                            centinela <− Falso
                        FinSi
                    FinMientras
                    Si (cantCifras>1) entonces
                        numEvaluar <− copiaNum
                        Si (cantCifras mod 2=0) Entonces
                            cifrasGrupo1 <− trunc(cantCifras/2)
                            cifrasGrupo2 <− trunc(cantCifras/2)
                        SiNo
                            cifrasGrupo1 <− trunc(cantCifras/2) + 1
                            cifrasGrupo2 <− trunc(cantCifras/2)
                        FinSi
                        grupo1 <− trunc(numEvaluar/10^(cifrasGrupo1))
                        grupo2 <− numEvaluar mod 10^(cifrasGrupo1)
                        Si (grupo1 + grupo2=num) entonces
                            Escribir "El número ", num, " es de Kaprekar porque ", numEvaluar, " genera los números ",
                                grupo1, " + ", grupo2, " = ", num
                            contador <− contador + 1
                            Si (num>=mayorKaprekar) Entonces
                                mayorKaprekar <− num
                            FinSi
                            Si (num<menorKaprekar) Entonces
                                menorKaprekar <− num
                            FinSi
                        SiNo
                            Si (cantCifras mod 2<>0) Entonces
                                cifrasGrupo1 <− cifrasGrupo1 − 1
                                cifrasGrupo2 <− cifrasGrupo2 + 1
                                grupo1 <− trunc(numEvaluar/10^(cifrasGrupo1))
                                grupo2 <− numEvaluar mod 10^(cifrasGrupo1)
                                Si (grupo1 + grupo2=num) entonces
                                    Escribir "El número ", num, " es de Kaprekar porque ", numEvaluar, "
                                        genera los números ", grupo1, " + ", grupo2, " = ", num
                                    contador <− contador + 1
                                    Si (num>=mayorKaprekar) Entonces
                                        mayorKaprekar <− num
                                    FinSi
                                    Si (num<menorKaprekar) Entonces
                                        menorKaprekar <− num
                                    FinSi
                                FinSi
                            FinSi
                        FinSi
                    FinSi
                    num <− num+1
                FinMientras
                Escribir "Existen ", contador, " números de Kaprekar"
                Escribir "El menor número Kaprekar del rango es: ", menorKaprekar
                Escribir "El mayor número Kaprekar del rango es: ", mayorKaprekar
            SiNo
                Escribir "La cantidad de cifras no es correcta"
            FinSi
FinAlgoritmo
```

3. El curioso número 142857 - Horarios: 0384, 0385, 0388, 0389

En matemáticas, existen algunos números que presentan ciertas curiosidades matemáticas. Dentro de ellos está el número 142857.

Este número se destaca, dentro de otras cosas, porque tiene la particularidad que al ser multiplicado por la secuencia de 2 a 6, el producto resultante corresponde a las mismas cifras del número original en otro orden. Por esta razón, este tipo de números se denominan cíclicos.

Por ejemplo:

- 142857 x 2 = 285714
- 142857 x 3 = 428571
- 142857 x 4 = 571428
- 142857 x 5 = 714285
- 142857 x 6 = 857142

Se le pide que diseñe un algoritmo, expresado en pseudocódigo, que permita identificar si un número es cíclico. Para ello debe leer el número, el cuál debe validar que sea mayor que 0. En caso no cumpla debe enviar el mensaje: El número debe ser mayor que 0 y el algoritmo debe terminar. Si el número ingresado es correcto, debe validar si el número es cíclico considerando la curiosidad descrita en el enunciado.

En los casos de prueba se indican los mensajes que debe usar dentro del pseudocódigo.

A continuación se presentan los casos de prueba:

```
Ingrese un número:
> -5
El número debe ser mayor que 0
```

```
Ingrese un número:
>  5
5 x 2 = 10
Los dígitos del número no se encuentran en el resultado.
```

```
Ingrese un número:
>  875
875 x 2 = 1750
Los dígitos del número no se encuentran en el resultado.
```

```
Ingrese un número:
>  65784
65784 x 2 = 131568
Los dígitos del número no se encuentran en el resultado.
```

```
Ingrese un número:
>  142857
142857 x 2 = 285714
142857 x 3 = 428571
142857 x 4 = 571428
142857 x 5 = 714285
142857 x 6 = 857142
El número 142857 cumple con la curiosidad
```

Programa 3: Propuesta de solución - El curioso número 142857

```
1 Algoritmo Propuesta3
2     Escribir "Ingrese un número: "
```

```
    Leer num
    Si num>0 Entonces
        multiplicador <- 2
        centinela <- Verdadero
        copiaNum <- num
        cumpleCuriosidad <- Verdadero
        Mientras (centinela) Hacer
            resultado <- num * multiplicador
            Escribir num," x ", multiplicador, " = ", resultado
            //Validamos que el resultado no tenga cifras repetidas
            copiaResultado <- resultado
            //Verificamos que todos los dígitos del número están en el número resultado
            centinela2 <- Verdadero
            num <- copiaNum
            aux <- copiaResultado
            cumplePatron <- Verdadero
            Mientras (centinela2) Hacer
                digito1 <- num mod 10
                num <- trunc(num/10)
                encontro <- Falso
                centinela3 <- Verdadero
                nuevoNum <- 0
                Mientras (centinela3) Hacer
                    digito2 <- aux mod 10
                    aux <- trunc(aux/10)
                    Si (digito1=digito2 y encontro=Falso) Entonces
                        encontro <- Verdadero
                    SiNo
                        nuevoNum <- nuevoNum*10 + digito2
                    FinSi
                    Si (aux=0) Entonces
                        centinela3 <- Falso
                    FinSi
                FinMientras
                Si (encontro=Falso) Entonces
                    centinela2 <- Falso
                    cumplePatron <- Falso
                SiNo
                    Si (num=0) Entonces
                        centinela2 <- Falso
                    FinSi
                FinSi
                aux <- nuevoNum
            FinMientras
            Si cumplePatron=falso Entonces
                Escribir "Los dígitos del número no se encuentran en el resultado"
                centinela <- Falso
                cumpleCuriosidad <- Falso
            FinSi
            multiplicador <- multiplicador + 1
            num <- copiaNum
            Si (multiplicador>6) Entonces
                centinela <- Falso
            FinSi
        FinMientras
        Si (cumpleCuriosidad) Entonces
            Escribir "El número ", copiaNum, " cumple la curiosidad"
        FinSi
    SiNo
        Escribir "El número debe ser mayor que 0"
    FinSi
FinAlgoritmo
```

4. El curioso número 142857 - Horarios: 0386, 0387

En matemáticas, existen algunos números que presentan ciertas curiosidades matemáticas. Dentro de ellos está el número 142857.

Este número se destaca, dentro de otras cosas, porque tiene la particularidad que al ser multiplicado por la secuencia del 8 al 13, el producto resultante corresponde a que sus cifras centrales, es decir sin considerar las cifras de los extremos, son las mismas cifras del número original en otro orden pero además la cifra que falta del número original es la suma del primer y el último dígito del producto resultante.

Por ejemplo:

- 142857 x 8 = 1142856
- 142857 x 9 = 1285713
- 142857 x 10 = 1428570
- 142857 x 11 = 1571427
- 142857 x 12 = 1714284
- 142857 x 13 = 1857141

Analizaremos una de las operaciones para entender mejor, por ejemplo: 142857 x 9 = 1285713, de esta operación vemos que del resultado 1285713, las cifras centrales (sin considerar los extremos) son 28571 y vemos que estas cifras se encuentran en el número 142857 a excepción del número 4, sin embargo, si sumamos las cifras de los extremos del resultado 1285713, es decir 1 y 3, la suma me da el número faltante que es el 4. Por lo tanto se va cumpliendo la curiosidad para dicha multiplicación. Esta misma regla se debe cumplir para los resultados de multiplicar por la secuencia del 8 al 13.

Se le pide que diseñe un algoritmo, expresado en pseudocódigo, que permita identificar si un número cumple la curiosidad indicada. Para ello debe leer el número, el cuál debe validar que sea mayor que 0. En caso no cumpla debe enviar el mensaje: El número debe ser mayor que 0 y el algoritmo debe terminar. Si el número ingresado es correcto, debe validar si el número cumple la curiosidad descrita en el enunciado.

En los casos de prueba se indican los mensajes que debe usar dentro del pseudocódigo.

A continuación se presentan los casos de prueba:

```
Ingrese un número:
> -5
El número debe ser mayor que 0
```

```
Ingrese un número:
>  5
5 x 8 = 40
Los dígitos del número original no se encuentran en los dígitos centrales del
resultado ni en la suma de los dígitos extremos del resultado.
```

```
Ingrese un número:
>  876
876 x 8 = 7008
Los dígitos del número original no se encuentran en los dígitos centrales del
resultado ni en la suma de los dígitos extremos del resultado.
```

```
Ingrese un número:
>  65784
65784 x 8 = 526272
Los dígitos del número original no se encuentran en los dígitos centrales del
resultado ni en la suma de los dígitos extremos del resultado.
```

```
Ingrese un número:
>  142857
142857 x 8 = 1142856
142857 x 9 = 1285713
142857 x 10 = 1428570
142857 x 11 = 1571427
142857 x 12 = 1714284
142857 x 13 = 1857141
El número 142857 cumple con la curiosidad
```

Programa 4: Propuesta de solución - El curioso número 142857

```
Algoritmo Propuesta4
    Escribir "Ingrese un número: "
    Leer num
    Si num>0 Entonces
        multiplicador <- 8
        centinela <- Verdadero
        copiaNum <- num
        cumpleCuriosidad <- Verdadero
        Mientras (centinela) Hacer
            resultado <- num * multiplicador
            Escribir num," x ", multiplicador, " = ", resultado
            //Validamos que el resultado no tenga cifras repetidas
            copiaResultado <- resultado
            //Verificamos que todos los dígitos del número están en el número resultado
            centinela2 <- Verdadero
            num <- copiaNum
            aux <- copiaResultado
            cumplePatron <- Verdadero
            i <- 1
            suma <- 0
            Mientras (centinela2) Hacer
                digito1 <- aux mod 10
                aux <- trunc(aux/10)
                encontro <- Falso
                centinela3 <- Verdadero
                nuevoNum <- 0
                Si (i=1) Entonces
                    suma <- suma + digito1
                SiNo
                    Si (aux=0) entonces
                        suma <- suma + digito1
                        centinela2 <- Falso
                        Si (suma <> num) Entonces
                            cumplePatron <- Falso
                        FinSi
                    Sino
                        Mientras (centinela3) Hacer
                            digito2 <- num mod 10
                            num <- trunc(num/10)
                            Si (digito1=digito2 y encontro=Falso) Entonces
                                encontro <- Verdadero
                            SiNo
                                nuevoNum <- nuevoNum*10 + digito2
                            FinSi
                            Si (num=0) Entonces
```

```
                                                centinela3 <- Falso
                                            FinSi
                                        FinMientras
                                        Si (encontro=Falso) Entonces
                                            centinela2 <- Falso
                                            cumplePatron <- Falso
                                        FinSi
                                        num <- nuevoNum
                                    FinSi
                                FinSi
                                i <- i + 1
                            FinMientras
                            Si cumplePatron=falso Entonces
                                Escribir "Los dígitos del número original no se encuentran en los dígitos centrales del resultado ni en
                                    la suma de los dígitos extremos del resultado"
                                centinela <- Falso
                                cumpleCuriosidad <- Falso
                            FinSi
                            multiplicador <- multiplicador + 1
                            num <- copiaNum
                            Si (multiplicador>13) Entonces
                                centinela <- Falso
                            FinSi
                        FinMientras
                        Si (cumpleCuriosidad) Entonces
                            Escribir "El número ", copiaNum, " cumple la curiosidad"
                        FinSi
                SiNo
                        Escribir "El número debe ser mayor que 0"
                FinSi
FinAlgoritmo
```

Puede usar cualquier estructura selectiva y cualquier estructura iterativa