



DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE CONCEPCIÓN
CONCEPCIÓN, CHILE.

Informe N°3

Los números pandigitales de 0-9

Profesor: Mario Medina

Alumno: Aldo Mellado Opazo

15 de septiembre de 2018

Programación Orientada al Objeto
Ingeniería Civil en Telecomunicaciones

1. Los números pandigitales de 0-9

Se nos introduce a los números pandigitales explicándonos que es el resultado de la combinatoria entre ciertos dígitos cuya principal propiedad es que estos aparecen sólo una vez. Es por esta definición que en la implementación de la resolución de esta tarea se consideró la combinatoria, llevada a cabo en el código por la siguiente función:

```
1 int factorial(int largol1)
2 {
3     int factorial=1;
4     for(int i=1;i<=largol1;i++)
5     {
6         factorial=factorial*i;
7     }
8     return factorial;
9 }
```

Esto, acorde a la combinatoria, es capaz de obtener el número de posibles combinaciones de dígitos entre 0-9, que de acuerdo a lo entregado por la función corresponde a 3628800.

Posteriormente se debían generar dichas permutaciones y para ello se hizo uso de la siguiente función.

```
1 void permutar(string l1,int factorial)
2 {
3     string l2;
4     l2 = l1;
5
6     string::iterator it=l2.begin();
7
8     for(int j=0;j<factorial;j++)
9     {
10         next_permutation(l2.begin(),l2.end());
11         propiedad(l2);
12     }
13 }
```

Notar que además de valerse de la función `next_permutation(l2.begin(),l2.end())` para obtener las posibles permutaciones, se hace uso de un archivo de texto generado al cual se le escriben los valores resultantes. Esto será especialmente útil para cuando deban hacerse ciertas consideraciones posteriores, pues permitirá que el procesamiento sea más rápido que si se mostrara por consola o se calculasen cada vez que se desee ejecutar una condición sobre los números de la permutación.

Luego se pide que para los números generados, se evalúe cuántos y cuáles de ellos satisfacen las condiciones dadas. Para dicho cometido se genera la siguiente función:

```
1 void propiedad(string l2)
2 {
3     list<int> aux;
4
5     char d1=l2[0],d2=l2[1], d3=l2[2],d4=l2[3],d5=l2[4];
6     char d6=l2[5],d7=l2[6], d8=l2[7],d9=l2[8],d10=l2[9];
7
8     //Contadores
9     int dv2=0,dv3=0,dv5=0,dv7=0,dv11=0,dv13=0,dv17=0;
10
11     string n2="{",n3="{",n5="{",n7="{",n11="{",n13="{",n17="{";
12
13     n2.push_back(d2);n2.push_back(d3);n2.push_back(d4);
14     dv2 = stoi(n2);
```

Como se puede ver, se tiene que `propiedad`, recibe un string, que corresponde a una de las permutaciones que realiza la función `permutar`. Este string es descompuesto, tal y como se muestra en la línea 3 y 4, en distintos elementos que señalan el lugar que ocupan dentro del string. Esto fue hecho para facilitar la verificación del cumplimiento de las propiedades que se debían evaluar.

Se construyen los números tomando los índices que le corresponden según la condición que deban evaluar. Hecho esto, eran empujadas a un string llamado `n` que, para evaluar la condición era transformado en un entero (usando `stoi`) y si se cumplía se evaluaba el siguiente número.

Sin embargo, se definió una condición que, representada por `flag`, señala si se cumple o no la condición. Por ejemplo, para el caso de `dv2`, construido a partir de las posiciones `d2,d3,d4`, se tiene que si este cumple la condición, entonces el valor de `flag` se modifica en favor de seguir evaluando las otras condiciones, tal y como se ve para :

```

1  if (dv2%2==0)
2  {
3      flag=1;
4  }
5  if (dv3%3==0 && flag==1)
6  {
7      flag=1;
8  }

```

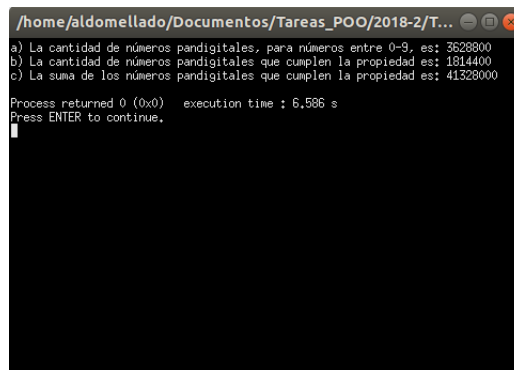
Si se llegaba con `flag=1` al término de la función, es decir, si el número, que entraba a la función en forma de string, y a partir del cual se construían los otros números con que se evaluaban las propiedades, cumplía las condiciones, este era convertido a una lista y sumado a `sumapandigitales` mediante el uso de las siguientes funciones:

```

1  void sumofList(list<int> l2)
2  {
3      transform (l2.begin(), l2.end(), sumapandigital.begin(), sumapandigital.begin(), plus<int>());
4  }
5
6  list<int> strToList(string str)
7  {
8      list<int> list1;
9      int j=0,k=0;
10
11     for(auto x:str)
12     {
13         k = int(x-48);
14         list1.push_back(k);
15     }
16     return list1;
17 }

```

1.1. Resultados



```

/home/aldomellado/Documentos/Tareas_POO/2018-2/T...
a) La cantidad de números pandigitales, para números entre 0-9, es: 3628800
b) La cantidad de números pandigitales que cumplen la propiedad es: 1814400
c) La suma de los números pandigitales que cumplen la propiedad es: 41528000

Process returned 0 (0x0) execution time : 6.586 s
Press ENTER to continue.

```

Fig. 1: Resultados Números pandigitales