4 Chalices (ABCD) The Real Chalice is 101st in the sequence ABCDCBABCD…. **The Real Chalice is Chalice C.**

Abstraction

We want to develop an algorithm that will solve the problem initially for 4 chalices and location 101. We should be able to change those initial conditions and our algorithm should still work.

Some of the key components of the problem are :

- The number of chalices and the location of the real chalice.
- The rules for the pattern of counting the chalices are incorporated into some data structure. For example an array, list or dictionary.
- Once the pattern is computed, the solution is easily read.

Writing your Thinking

Take 5 minutes to think about how you tackled this problem.

- **Did you use pen and paper to help visualise possible solutions?**
- Did you break it down to a small number of chalices first to make the problem easier to understand?
- **Did you look for patterns in order to make predictions?**
- Did you try different scenarios to verify your solution worked?

Using Think-Pair-Share-Square (TPSS), go through how you and your partner were thinking about how to solve the problem.

Write an algorithm to solve this problem in a computational way.

Test your algorithm for different locations of the real chalice such as 102,103,104 and 105.

Pseudo-Code

```
#Initialise the conditions
numberChalices = 4; realChalice = 101;
Create a list of 4 letters: chaliceList=  [A,B,C,D]        #'A' is 65 in ASCII
#You could also create an alphaList [A,B,C,D, …Y,Z] and from that make a chaliceList
#Set up the pattern of counting the chalices
For i = 2 to 0 {
        Make a backward list of [C,B,A]
}
For i = 1 to 3 {
        Make a forward list of [B,C,D]
}
While length of chaliceList < 101 {
        Append backward list
        Append forward list
}
101st element of chaliceList will be the real chalice
```

An extension of the challenge

We could abstract our computational solution even further away from the actual counting of the chalices, by using the modular arithmetic solution (101 mod 6).

The user could just supply the 2 key components of the number of chalices and the location of the real one, and the program simply applies modular arithmetic. (In Python this is 101 % 6).

There is still one vital piece of the jigsaw missing when you calculate the number 6. Can you write a program at this slightly higher level of abstraction.