Computing Science 204 (B1)
Bit strings not containing the pattern 010
Monday, July 24

Example. *Bit strings not containing the pattern* 010

Let a_n denote the number of n-bit strings of 0's and 1's that do not contain the pattern 0 1 0.

(a) Find a recurrence relation for the sequence $\{a_n\}_{n \geq 1}$.

(b) How many 32-bit strings contain the pattern 0 1 0?

(c) Write a C program that will generate, for any $n \geq 1$, all n-bit strings that do not contain the pattern 0 1 0.

SOLUTION:

(a) Let α be an n–bit string not containing the pattern 0 1 0. We will count the number of such strings having k leading 0's.

For $k = 0$, there are a_{n-1} such strings. For $k = 1$, the string begins 0 1 1, so there are a_{n-3} such strings. Similarly, for $k = 2$, the string begins 0 0 1 1, so there are a_{n-4} such strings. Continuing in this manner, for $k = n - 3$, there are a_1 such strings. For $k = n - 2$, $k = n - 1$, or $k = n$, there is 1 such string. Therefore, a_n satisfies the recurrence relation

$$a_n = a_{n-1} + a_{n-3} + a_{n-4} + \cdots + a_1 + 3. \qquad (*)$$

If we replace n by $n - 1$ in $(*)$, we get

$$a_{n-1} = a_{n-2} + a_{n-4} + a_{n-5} + \cdots + a_1 + 3, \qquad (**)$$

and subtracting $(**)$ from $(*)$, then a_n satisfies the recurrence relation

$$\begin{aligned} a_n &= 2a_{n-1} - a_{n-2} + a_{n-3} \\ a_1 &= 2 \\ a_2 &= 4 \\ a_3 &= 7 \end{aligned}$$

for $n \geq 4$.

(b) The number of 32–bit strings that do not contain the pattern 0 1 0 is $2^{32} - a_{32}$, and this is

$$4,294,967,296 - 82,938,844 = 4,211,028,452.$$

A C program that will generate, for any $n \geq 1$, the number a_n of n–bit strings that do not contain the pattern 0 1 0 is given below:

```
#include <stdio.h>
void main()
{  int k,n,t,u,v,w;
   printf("Enter an index n > 3:\n");
   scanf("%d",&n);
   u = 7; v = 4; w = 2; k = 4;
   while (k <= n)
   {   t = 2*u - v + w;
       w = v; v = u; u = t;
       k = k + 1;
   }
   printf("The number of %d bit strings without 010 is: a_%d = %d\n",n,n,t);
}
```

(c) The following C program will generate, for any $n \geq 1$, all n–bit strings that do not contain the pattern 0 1 0.

```
#include <stdio.h>

void valid_patterns();
int pow2();

main()
{  int n;
   printf("Enter the number of bits n:\n");
   scanf("%d",&n);
   valid_patterns(n);
}

void valid_patterns(n)
int n;
{  int i,j,k,u,valid;
   int *A;
   A = (int *)malloc(n*sizeof(int));
   printf("The strings not containing  0 1 0  are:\n");
   for (u = 0; u < pow2(n); u++)
   {   valid = 1;
       k = u;
       for (i = n-1; i >= 0; i--)      /* Get the bit pattern   */
       {   A[i] = k%2;
           k = k/2;
       }
       j = 0;                          /* Check to see if valid */
       while (j <= n-3)
       {   if (A[j] == 0 && A[j+1] == 1 && A[j+2] == 0)
              valid = 0;
           j++;
       }
       if (valid == 1)                 /* Print the bit pattern */
       {   for (i = 0; i < n; i++)
               printf("%d",A[i]);
           printf("\n");
       }
   }
}

int pow2(n)
int n;
{  int k,pow;
   pow = 1;
   k = 0;
   while (k < n)
   {  pow = pow*2;
      k = k + 1;
   }
   return(pow);
}
```