The previous lectures explain how to count the number of options when selecting balls with restrictions. This lecture explains how to print the allowed options.

The problem we want to solve is to select red or blue balls, with the restriction that two adjacent balls must not be both red.

This slide shows the output when selecting 4 balls. As you can see, two adjacent balls cannot be both red. How do we write a program to produce output like this?

This is the program. As you can see, the program is actually quite short. I will zoom in and explain the program in details.

At the top of the program, a constant array is created. This array has two characters R or B representing the two possible colors.

The main function takes an integer as the number of balls to be selected. Suppose we enter four for A R G V one, the variable N U M B A L L is 4. Colors is a pointer for heap memory. The program uses malloc to allocate memory in heap. The amount of memory is the same as the number of balls to select. This array will store the colors of selected balls.

Then, the main function calls the function select ball with three arguments: the array that can store the colors of selected balls, the number of balls to select, and the number of already selected balls.

After calling the select ball function, the main function calls free to release the memory. Finally, the program returns exit success.

Next, let’s consider the select ball function. At the top of the function is the stop condition. When the second argument and the third argument are equal, the function prints the color array and returns. In other words, when the number of selected balls is the same as the total number of balls to be selected, this function returns.

Please remember that the second argument is the total number of balls to be selected and its value starts at 4. The third argument is the number of balls already selected. It starts at zero.

Since these two values are different at the first call of this select ball function, the stop condition is false.

The function moves down to the for loop of J. .

This for loop has values of 0 or 1 because there are two possible colors to choose. Remember the constant array we created earlier? The constant array has two elements: R and B. .

A Boolean variable stores whether a selection can be accepted or not. It value is initialized to true.

The next if condition checks whether we are selecting the very first ball. If this is the first ball, the number of selected balls is zero. Since no ball has been selected yet, there is no restriction.

Since the condition is true, the program sets the array of the color’s first element to the available color.

The constant array for available colors starts with R; thus, the first selected color is red.

The next line calls the select ball function. Since this is a function call, a new frame is pushed to the top of the stack memory. Please notice that the number of selected ball increases by one. Thus, the argument in the top frame is 1 now.

The function call will start from the top and check the stop condition whether the number of selected ball is the same as the total number of balls to be selected. If they are different, the program continues to select a color. The program checks whether the number of selected ball is zero or not. If it is not zero, at least one ball has already been selected.

Next, the program checks whether the previously selected colors is red and the currently selected color is also red.

If both are red, the condition is set to false, meaning that this selection cannot be accepted.

Then, j becomes 1. The condition is reset to true.

The number of selected ball is not zero. Thus, we have to check whether two adjacent balls are both red.

When j is 1, the color is blue. Thus, this if condition is not satisfied. The condition is unchanged and still true.

Since condition is true, the program continues to assign the color.

The second ball is set to blue. Please remember that the array index starts at zero. When the index is one, it is the second ball.

After assigning blue to the second ball, the program moves down and it is another function call.

As always, a function call means a new frame is pushed to the top of the stack.

The total number of balls to select is unchanged and still 4. The number of selected ball becomes 2.

Since these two numbers are different, the stop condition is not met.

The program continues to set J to zero and the condition to true.

The number of selected ball is not zero because we are now selecting the third ball. This if condition is true.

Next, the program checks whether the previous ball is red and the currently selected ball is also red.

The previously selected ball is blue and this if condition is false.

The program goes on and selects red ball.

If you continue tracing the program, you will discover that the program will select Blue for the fourth ball.

The first four selected balls are R B R B. . The second is R B B R. .

What will happen if we exchange the order of the two colors in the available array?

The first four selected balls are B B B B. . The second is B B B R. .

Please read this program carefully and fully understand every details.