CS101 Lab 4: Arrays with For Loops: Flat and Nested Iterations

1 What to Do?

1. Read Section 2. Refer to lecture slides/programs.
2. Try sample programs. Make changes, relate to output.
3. Solve problems in Section 5.

2 Syntax of For Loops

```
for (<initialization>; <termination condition>; <step>)  <statement>
```

The initialization is carried out first. Then the termination condition is checked. After this, the body statement is executed if the loop is not to terminate. After the body is executed, the step is executed next, and then next iteration starts with the next termination check.

3 Sample Program 1

```
#include <iostream>
using namespace std;

int main () {
   int n, i;

   cin >> n;            // n is read
   int A[n];            // now an array is declared of size n

   for (i=0; i<n; i++) cin >> A[i];     // we read n numbers
   for (i=0; i<n; i++) cout << (A[i]*A[i]) << " "; // the array is traversed

   cout << endl;                // for beautification of output
}
```

4 Sample Program 2

```
// Name:
// Roll No:
// Program to print a right handed right angle triangle of n rows
// with the '*' symbol

#include<iostream>
using namespace std;

int main(){

  int n;

  cout << "Enter number or rows.";
  cin >> n;

  for(int i = 0; i < n; i++){
    for(int j = 0; j <= i; j++)
      cout << "*";
    cout << endl;
  }
  return 0;
}
```

5 Problems

1. Start with program given in section 3 and write a program that finds out the frequency of occurance of a given value v inside an array of size N. Read N, the actual numbers, and value v from the keyboard before you compute the frequency.

2. Modify the above program to compute the frequencies of all distinct numbers that appear in the array.

3. Try the program given in Section 4. Given number n, observe what it prints. Modify this program to print a full isosceles triangle of n rows.

4. Write a program to display Pascal's triangle. Pascal's triangle represents the binomial coefficients. The first few rows of Pascal's triangle are displayed below. Observe how a row is related to the row above it.

```
          1
        1  1
      1  2  1
    1  3  3  1
  1  4  6  4  1
 1  5  10  10  5  1
1  6  15  20  15  6  1
```