1a.

```
/*
** Problem 1a
*/
public static void hasPair( int[] arr, int sum ) {                  //
    int test = 0;                                                   // 1
    for(int i=0; i<arr.length; i++ ) {                              // n+1
        for(int j=i+1; j<arr.length; j++ ) {                        // (n(n+1))/2
            if( arr[i] + arr[j] == sum ) {                          // n
                System.out.println("Output: " + arr[i] + " + "       // 3
                    + arr[j] + " = " + sum + " , true");            //
                return;                                             //
            }                                                       //
        }                                                           //
    }                                                               //
    System.out.println("Output: false");                            // 1
}
```

The total number of operations is $\frac{n(n+1)}{2} + 2n + 6$.

The time complexity is $O(n^2)$.

1b.

```
/*
** Problem 1b
*/
public static void hasTriplet( int[] arr, int sum ) {               //
    int test = 0;                                                   // 1
    for(int i=0; i<arr.length; i++ ) {                              // n+1
        for(int j=i+1; j<arr.length; j++ ) {                        // (n(n+1))/2
            for( int k=j+1; k<arr.length; k++ ) {                   // (n(n+1)(n+2))/6
                if( arr[i] + arr[j] + arr[k] == sum ) {             // n
                    System.out.println("Output: " + arr[i] +        // 4
                     " + " + arr[j] + " + " + arr[k] +              //
                     " = " + sum + " , true");                      //
                    return;                                         //
                }                                                   //
            }                                                       //
        }                                                           //
    }                                                               //
    System.out.println("Output: false");                            // 1
}
```

The total number of operations is $\frac{n(n+1)(n+2)}{6} + \frac{n(n+1)}{2} + 2n + 7$.

The time complexity is $O(n^3)$.

1c.

```
/*
 * Problem 1c
 */

for( int i = 0; i < n; i++ ) {                  // n+1
    for( int j = 0; j < n; j++ ) {              // (n(n+1))/2
        double sum = 0;                         // (n(n+1))/2
        for( int k = 0; k < n; k++ ) {          // (n(n+1)(n+2))/6
            sum += a[i][k] * b[k][j];           // (n(n+1)(n+2))/6
        }                                       //
        c[i][j] = sum;                          // (n(n+1))/2
    }
}
```

The total number of operations is $\frac{n(n+1)(n+2)}{3} + \frac{3n(n+1)}{2} + n + 1$.

The time complexity is $O(n^3)$.