```
////////// ( practice questions )
```

QUES 1:

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

void calAvgAndSum(int& sum, int& avg){

        sum = 0;
        for (int i = 1; i <= 10; i++) {
                int num;
                cout << "Enter a number " << endl;
                cin >> num;
                sum = sum + num;
        }
        avg = sum / 10;
}
int main() {
        int sum, avg;

        calAvgAndSum(sum, avg);
        cout << "The sum is " << sum << endl;
        cout << "the average is " << avg << endl;
}
```

QUES 5:

```
#include<iostream>
using namespace std;
int main() {
        int n, m;
        cout << "Enter values of n & m ";
        cin >> n >> m;
        int sum = n + m;

        if (n >= m) {      // cheak
                cout << "Invalid input . ";
        }
        int count = 0;
        for (int i = n + 1; i < m; i++) {
                if (sum % i == 0) {
                        count ++ ;
                }
        }

        cout << "Num b/w n and m (n<m) that divide the " << sum << " is " << count <<
endl;
}
```

QUES 4:

```
#include <iostream>
using namespace std;
// Function to check if a number is prime
bool isPrime(int number) {
    if (number <= 1) {
        return false;
    }
    for (int i = 2; i <= number/2; i++) {
        if (number % i == 0) {
            return false;
        }
    }
    return true;
}

// Function to count the number of prime values
void countPrimes(int& primeCount) {
    int number;
    int i = 0;
    while (i<10) {
        cout << "Enter a positive number (-1 to quit): ";
        cin >> number;

        if (number == -1) {
            break;
        }

        if (number <= 0) {
           cout << "Invalid input. Please enter a positive number." << endl;
            continue;
        }

        if (isPrime(number)) {
            primeCount++;
        }
        i++;
    }
}

int main() {
    int primeCount = 0;
    int compositeCount = 0;

    // Call the function to count the number of prime values
    countPrimes(primeCount);

    compositeCount = primeCount; // The rest are composites

    cout << "Number of prime values entered: " << primeCount << endl;
    cout << "Number of composite values entered: " << compositeCount << endl;

    return 0;
}
```

QUES 2:

```
#include <iostream>
using namespace std;
// Function to check if a number is prime
void checkPrime(int number, int& primeCount, int& compositeCount) {
    if (number <= 1) {
        compositeCount++;
    }
    else {
        bool isPrime = true;
        for (int i = 2; i * i <= number; i++) {
            if (number % i == 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) {
            primeCount++;
        }
        else {
            compositeCount++;
        }
    }
}

int main() {
    int primeCount = 0;
    int compositeCount = 0;

    int number;
    while (true) {
        std::cout << "Enter a positive number (-1 to quit): ";
        std::cin >> number;

        if (number == -1) {
            break;
        }

        if (number <= 0) {
            std::cout << "Invalid input. Please enter a positive number." <<
std::endl;
            continue;
        }

        checkPrime(number, primeCount, compositeCount);
    }

    std::cout << "Number of prime values entered: " << primeCount << std::endl;
    std::cout << "Number of composite values entered: " << compositeCount <<
std::endl;

    return 0;
}
```

1)Write down a void function that accepts two integers and evaluates their sum using the concept of
reference parameters. Then use this function in main to calculate the sum and average of 10 integers
entered by user.

2) Write down a void function that accepts two values, a number and a counter. If the number received is prime it increments the counter. The main program uses this function to calculate the number of
prime values entered in a user defined input. The input is sentinel controlled and accepts positive numbers only. The program just skips to next iteration, if the value is neither sentinel nor positive. The
sentinel value may be assumed to be -1 At the end number of primes are displayed.

3) Write down above program of question2 using a value returning function instead. The function returns a Boolean value; and does not have a reference parameter. Everything else remains the same

4) 2) Write down a void function that accepts two values, a number and a counter. If the number received is prime it increments the counter. The main program uses this function to calculate the number of prime values entered in a user defined input. The input is sentinel controlled and accepts positive numbers only. The program just skips to next iteration, if the value is neither sentinel nor positive. The sentinel value may be assumed to be -1 At the end number of primes and composites are displayed.

5) Write down a program with following specifications

Input: Positive integer’s n and m

Output: Number of integers between n and m (where n&lt;m) that divide the sum n+m.
For example, if n=8, and m=12, the program returns 1 (because 10 is the only number between n and m
which divides n+m=20); and if n=4 and m=28, the program returns 2 (because 8 and 16 are two numbers
between n and m which divide n+m=32)