Fundamentals of Programming

Assignment # 1

Name: Hassan Aun Ali

CMS: 463654

ME-15B

1. Check if an integer value is between 10 and 20

```
#include <iostream>

int main() {
   int num;
   std::cout << "Enter an integer: ";
   std::cin >> num;

   int result = (num > 10 && num <= 20) ? 1 : 0;
   std::cout << "Result: " << result << std::endl;

   return 0;
}
```

2. Find the largest prime number less than a given positive integer N:

```
#include <iostream>

bool isPrime(int n) {
   if (n <= 1) return false;
   for (int i = 2; i * i <= n; ++i) {
      if (n % i == 0) return false;
   }
   return true;
}

int main() {
   int N;
   std::cout << "Enter a positive integer: ";
   std::cin >> N;

   int largestPrime = 0;
   while (N > 0) {
      if (isPrime(N)) {
         largestPrime = N;
         break;
      }
      --N;
   }

   std::cout << "Largest prime less than or equal to N: " << largestPrime << std::endl;
```

```
    return 0;
}
```

3. Check if two strings are equal and make them unequal by rotating if they are:

```
#include <iostream>
#include <string>

int main() {
    std::string str1, str2;
    std::cout << "Enter two strings: ";
    std::cin >> str1 >> str2;

    if (str1 == str2) {
        std::rotate(str1.begin(), str1.begin() + 1, str1.end());
        std::cout << "Strings made unequal: " << str1 << " " << str2 << std::endl;
    } else {
        std::cout << "Strings are already unequal." << std::endl;
    }

    return 0;
}
```

4. Perform division without using '/' using for loops:

```
#include <iostream>

int main() {
    int dividend, divisor;
    std::cout << "Enter dividend and divisor: ";
    std::cin >> dividend >> divisor;

    int quotient = 0;
    while (dividend >= divisor) {
        dividend -= divisor;
        quotient++;
    }

    std::cout << "Quotient: " << quotient << std::endl;

    return 0;
}
```

5. Suppose an integer array a[5] = {1,2,3,4,5}. Add more elements to it and display them in C++.

```
#include <iostream>

int main() {
   int a[8] = {1, 2, 3, 4, 5}; // Initial array
   int additionalElements[3] = {6, 7, 8}; // Additional elements to add

   // Adding additional elements to the array
   for (int i = 5; i < 8; ++i) {
      a[i] = additionalElements[i - 5];
   }

   // Displaying the updated array
   std::cout << "Updated array elements: ";
   for (int i = 0; i < 8; ++i) {
      std::cout << a[i] << " ";
   }
   std::cout << std::endl;

   return 0;
}
```

9. Given an integer array and an integer X. Find if there's a triplet in the array which sums up to the given integer X.

```
#include <iostream>

bool findTriplet(int arr[], int n, int target) {
   for (int i = 0; i < n - 2; ++i) {
      for (int j = i + 1; j < n - 1; ++j) {
         for (int k = j + 1; k < n; ++k) {
            if (arr[i] + arr[j] + arr[k] == target) {
               std::cout << "Triplet found: " << arr[i] << ", " << arr[j] << ", " << arr[k] << std::endl;
               return true;
            }
         }
      }
   }
   return false;
}

int main() {
```

```
    int arr[] = {1, 4, 45, 6, 10, 8};
    int target = 22;
    int n = sizeof(arr) / sizeof(arr[0]);

    if (!findTriplet(arr, n, target)) {
        std::cout << "No triplet found with the sum equal to " << target << std::endl;
    }

    return 0;
}
```

10. Implement Bubble Sort on an array of 6 integers.

```
#include <iostream>

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n - 1; ++i) {
        for (int j = 0; j < n - i - 1; ++j) {
            if (arr[j] > arr[j + 1]) {
                // Swapping elements if they're in the wrong order
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

int main() {
    int arr[] = {64, 34, 25, 12, 22, 11};
    int n = sizeof(arr) / sizeof(arr[0]);

    bubbleSort(arr, n);

    std::cout << "Sorted array: ";
    for (int i = 0; i < n; ++i) {
        std::cout << arr[i] << " ";
    }
    std::cout << std::endl;

    return 0;
}
```