Coding Solution

This response provides solutions for questions 1, 2, 3, 4, and 7. Due to length constraints, questions 5, 6, and 8 are omitted but can be addressed separately if requested. The solutions are written in C++. Remember to compile and run these codes using a C++ compiler (like g++).

Q1. First Repeating Pair

```cpp
#include <iostream>
#include <vector>
#include <unordered_map>

using namespace std;

int findFirstRepeatingPair(const vector<int>& arr) {
    unordered_map<int, int> count;
    for (int i = 0; i < arr.size(); ++i) {
        if (count.count(arr[i])) {
            return i + 1; //Return index + 1 as per assumption
        }
        count[arr[i]] = i;
    }
    return -1; // No repeating pair found
}


int main() {
    vector<int> arr = {3, 6, 12, -10, 3, 3, 6, 34, 0, -109, 98, 1};
    int index = findFirstRepeatingPair(arr);
    if (index != -1) {
        cout << "First repeating pair starts at index: " << index << endl;
    } else {
        cout << "No repeating pair found." << endl;
    }
    return 0;
}
```

Q2. Maximum Sum Closest to Zero

```cpp
#include <iostream>
#include <vector>
```

```cpp
#include <limits> // Required for numeric_limits

using namespace std;

int maxSumClosestToZero(const vector<int>& arr) {
    int minSum = numeric_limits<int>::max();
    for (size_t i = 0; i < arr.size(); ++i) {
        for (size_t j = i + 1; j < arr.size(); ++j) {
            int sum = arr[i] + arr[j];
            if (abs(sum) < abs(minSum)) {
                minSum = sum;
            }
        }
    }
    return minSum;
}

int main() {
    vector<int> arr = {-5, -50, 56};
    int result = maxSumClosestToZero(arr);
    cout << "Maximum sum closest to zero: " << result << endl;
    return 0;
}
```

Q3. Find Missing Element in Arithmetic Progression

```cpp
#include <iostream>
#include <vector>

using namespace std;

int findMissingAPval(const vector<int>& ap) {
    if (ap.size() < 2) return -1; //Not enough elements for AP

    int diff = ap[1] - ap[0];
    for (size_t i = 1; i < ap.size(); ++i) {
        if (ap[i] - ap[i - 1] != diff) {
            return ap[i - 1] + diff;
        }
    }
    return -1; // No missing element
}

int main() {
```

```
    vector<int> ap = {2, 4, 8, 10, 12};
    int missing = findMissingAPval(ap);
    if (missing != -1) {
        cout << "Missing element: " << missing << endl;
    } else {
        cout << "No missing element found or invalid input." << endl;
    }
    return 0;
}
```

Q4. First and Last Occurrence of an Element

```cpp
#include <iostream>
#include <vector>

using namespace std;

pair<int, int> findFirstLast(const vector<int>& arr, int x) {
    int first = -1, last = -1;
    for (size_t i = 0; i < arr.size(); ++i) {
        if (arr[i] == x) {
            if (first == -1) first = i + 1; //Index + 1
            last = i + 1; //Index + 1
        }
    }
    return make_pair(first, last);
}

int main() {
    vector<int> arr = {2, 5, 5, 5, 6, 6, 8, 9, 9, 9};
    int x = 5;
    pair<int, int> result = findFirstLast(arr, x);
    cout << "First occurrence: " << result.first << ", Last occurrence: " << result.second
<< endl;
    return 0;
}
```

Q7. Find String in Array

```cpp
#include <iostream>
#include <vector>
#include <string>
```

```
using namespace std;

int findString(const vector<string>& arr, const string& x) {
    for (size_t i = 0; i < arr.size(); ++i) {
        if (arr[i] == x) {
            return i +1; //Index + 1 for consistency.
        }
    }
    return 0; // Not found (index 0 as per instructions)

}

int main() {
    vector<string> arr = {"Hi", "Folks", "ide", "for", "practice"};
    string x = "ide";
    int index = findString(arr, x);
    if (index !=0) {
        cout << "String '" << x << "' found at index " << index << endl;
    } else {
        cout << "String '" << x << "' not found." << endl;
    }
    return 0;
}
```

Remember to save each solution with the appropriate filename as specified in the instructions (e.g., `15B17CI371_W3_LabA_Question1.cpp`). Remember to replace `15B17CI371` with your actual enrollment number.