DS [Day 2]

UID: 24MCI10204

Name: Rahul Saxena

Branch: 24MCA – AI & ML

Question 1: Write a program in c/c++ to check whether a given string of brackets is balanced. Example: {[()]} → Yes; {[(])} → No.

Code:

```
#include <iostream>
#include <stack>
using namespace std;
bool isBalanced(const string& s) {
   stack<char> st;
   for (char ch : s) {
      if (ch == '(' || ch == '[' || ch == '{') {
         st.push(ch);
      } else {
         if (st.empty()) return false;
         char top = st.top();
         if ((ch == ')' && top != '(') ||
            (ch == ']' && top != '[') ||
            (ch == '}' && top != '{')) {
            return false;
         }
         st.pop();
      }
   }
   return st.empty();
}
int main() {
   string input;
   cout << "Enter bracket string: ";
   cin >> input;
   if (isBalanced(input)) {
      cout << "Yes\n";
   } else {
      cout << "No\n";
   }
   return 0;
}
```

Question 2: You need to simulate a printer queue where each document has a priority. Print only the highest-priority document first without heap.

Code:

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

struct Document {
   int id;
   int priority;
};

int main() {
   int n;
   cout << "Enter number of documents: ";
   cin >> n;

   queue<Document> q;
   vector<int> priorityCount(10, 0);

   for (int i = 0; i < n; ++i) {
      int p;
      cout << "Enter priority for document " << i + 1 << ": ";
      cin >> p;
      q.push({i + 1, p});
      priorityCount[p]++;
   }

   cout << "\nPrint order:\n";
   while (!q.empty()) {
      Document front = q.front();
      q.pop();

      bool hasHigher = false;
      for (int i = front.priority + 1; i <= 9; ++i) {
         if (priorityCount[i] > 0) {
            hasHigher = true;
            break;
         }
      }
      if (hasHigher) {
         q.push(front);
      } else {
         cout << "Printing Document " << front.id << " (Priority: " << front.priority << ")\n";
         priorityCount[front.priority]--;
      }
   }
   return 0;
}
```