```
#ifndef TBuffer_h
#define TBuffer_h
#include <iomanip>
#include <iostream>
#pragma once

using namespace std;

template <class T, int dataCapacity>
class TBuffer {
protected:
    T data[dataCapacity];
    int dataLength;
    T nullValue;
public:
    TBuffer(T nullValue) {
        dataLength = 0;
        this->nullValue = nullValue;
    }


    int add(T value) {
        if (dataLength < dataCapacity) {
            data[dataLength] = value;
            ++dataLength;
            return 1;
        }
        else {
            return 0;
        }
    }


    int add(const T array[], int arrayLength) {
        int count = 0;
        for (int i = 0; i < arrayLength; ++i) {
            count += add(array[i]);
        }
        return count;
    }

    void print() const {
        for (int i = 0; i < dataLength; ++i) {
            if (i > 0 && i % 10 == 0)
                cout << '\n';
            cout << setw(4) << data[i];
        }
        cout << endl;
    }

    T sum() {
        T sum = nullValue;
        for (int i = 0; i < dataLength; ++i) {
            sum += data[i];
        }
        return sum;
    }

    const int getDataLength() { return dataLength; }
```

```
};
#endif
```