```
/**
* Multi-files project demonstration for most frequent product.
*/

---------
main.cpp
---------
/**
 * Popular Product Demo, i.e. most frequent items from the list.
 */
#include <malloc.h>
#include "arithmetic.h"
#include "console_utils.h"

/**
 * Program entry point.
 * @return 0 with successful execution.
 */
int main() {

    /**
     * Demo Data.
     */
    Product items[10] = {{1}, {6}, {1}, {2},  {1}, {6}, {9}, {9}, {2}, {1}};
    Sales *sales = (Sales *) malloc(sizeof(Product) * 10);
    sales->productItems = items;
    sales->count = 10;

    /**
     * Accumulate product statistics.
     */
    ProductStats *productStats = accumulateProductFrequency(sales);
    printf("\nTotal Stats: %ld", productStats->entryCount);

    /**
     * Highest frequent product.
     */
    ProductFrequency *product = maxOccurrence(productStats);

    /**
     * Display highest frequent product.
     */
    displayProduct(product);
    return 0;
}
```

```
-------------
arithmetic.h
-------------
/* ********************************************** */
/* This file is a part of popular. */
/* Created by santa on 2022-03-27. */
/* Everest Engineering College. */
/* Sanepa - 2, Lalitpur. */
/* https://www.eemc.com.np */
/* ********************************************** */

#ifndef POPULAR_ARITHMETIC_H
#define POPULAR_ARITHMETIC_H

const int MAX_SIZE = 100;

/**
 * Boolean Literals.
 */
const int TRUE = 1;
const int FALSE = 0;

/**
 * Definition of Product Data.
 */
typedef struct {
    long id;
} Product;

/**
 * Definition of sales data.
 */
typedef struct {
    Product *productItems;
    long count;
} Sales;

/**
 * Define the product frequencies.
 */
typedef struct {
    Product product;
    long frequency;
} ProductFrequency;

/**
 * Product Statistics.
 */
struct ProductStats {
    ProductFrequency *productFrequency;
    long entryCount;
};

/**
 * Finds the max occurrence of the product.
 */
ProductFrequency *maxOccurrence(ProductStats *);
ProductStats *accumulateProductFrequency(Sales *);
int isAlreadyCounted(ProductStats *, Product);
#endif //POPULAR_ARITHMETIC_H
```

```
---------------
arithmetic.cpp
---------------
/* ********************************************** */
/* This file is a part of popular. */
/* Created by santa on 2022-03-27. */
/* Everest Engineering College. */
/* Sanepa - 2, Lalitpur. */
/* https://www.eemc.com.np */
/* ********************************************** */

#include <malloc.h>
#include "arithmetic.h"

/**
 * Checks if the sales product is already present or not,
 * if so, it returns the positive index, otherwise it
 * returns -1.
 */
int isAlreadyCounted(ProductStats *productStats, Product product) {
    int found = -1;
    for (int index = 0; index < productStats->entryCount; index++) {
        if (productStats->productFrequency->product.id == product.id) {
            found = index;
            break;
        }
    }
    return found;
}

/**
 * Calculates the product frequency table.
 */
ProductStats *accumulateProductFrequency(Sales *sales) {
    ProductStats *productStats = (ProductStats *) malloc(sizeof(ProductStats));
    productStats->productFrequency = (ProductFrequency *)
malloc(sizeof(ProductFrequency) * MAX_SIZE);
    productStats->entryCount = 0;
    for (int index = 0; index < sales->count; index++) {
        int isFound = isAlreadyCounted(productStats, sales-
>productItems[index]);
        if (isFound >= 0) {
            productStats->productFrequency[isFound].frequency++;
        } else {
            (productStats->productFrequency + productStats->entryCount)->product
= sales->productItems[index];
            (productStats->productFrequency + productStats->entryCount++)-
>frequency = 1L;
        }
    }
    return productStats;
}

/**
 * Calculates the most frequent item from the sales data.
 * @param sales
```

```
 * @return productItem
 */
ProductFrequency *maxOccurrence(ProductStats *productStats) {
    int max = 0;
    ProductFrequency *productFrequency = NULL;
    for (int index = 0; index < productStats->entryCount; index++) {
        if (productStats->productFrequency[index].frequency > max) {
            productFrequency = productStats->productFrequency;
        }
    }
    return productFrequency;
}
```

```
----------------
console_utils.h
----------------

/* ************************************************ */
/* This file is a part of popular. */
/* Created by santa on 2022-03-27. */
/* Everest Engineering College. */
/* Sanepa - 2, Lalitpur. */
/* https://www.eemc.com.np */
/* ************************************************ */

#include "arithmetic.h"

#ifndef POPULAR_CONSOLE_UTILS_H
#define POPULAR_CONSOLE_UTILS_H

void displayProduct(ProductFrequency *);

#endif //POPULAR_CONSOLE_UTILS_H

------------------
console_utils.cpp
------------------
/* ************************************************ */
/* This file is a part of popular. */
/* Created by santa on 2022-03-27. */
/* Everest Engineering College. */
/* Sanepa - 2, Lalitpur. */
/* https://www.eemc.com.np */
/* ************************************************ */

#include <stdio.h>
#include "arithmetic.h"

/**
 * Displays product information to the console.
 */
void displayProduct(ProductFrequency *productFrequency) {
    fprintf(stdout, "\n*******************************");
    fprintf(stdout, "\nProduct Information: ");
    fprintf(stdout, "\n\tID              : %ld", productFrequency->product.id);
    fprintf(stdout, "\n\tFrequency Count : %ld", productFrequency->frequency);
    fprintf(stdout, "\n*******************************");
}


-----------------------------
Program Output:
-----------------------------
Total Stats: 7
*******************************
Product Information:
       ID              : 1
       Frequency Count : 4
*******************************
```