```cpp
#include<iostream>
#include<string>
using namespace std;

int VariableNumberOfObject = 3;
const int DefaultNumberOfObject = 3;
string SmallestObjects[DefaultNumberOfObject];
string LargestObjects[DefaultNumberOfObject];

const string ReadObjectNameNotification = "Enter the name of the object: ";
const string  ReadNumOfObjectNotification = "Enter the number of the objects ";
const string  SmallestObjectNotification = "\n\tSmallest object's Names : \n";
const string  LargestObjectNotification = "\n\t Largest object's Names : \n";

string GetObjectPrompt(int number) {
    return ReadObjectNameNotification + to_string(number);
}

string ReadObjectName(const string& message) {

    string _objectName = "";
    cout << message << endl;
    getline(cin, _objectName);

    return _objectName;
}

void ReadSmallestObjectNames() {
    cout << "\nReading Smallest Objects\n";
    for (int i = 0; i < DefaultNumberOfObject; i++) {
        SmallestObjects[i] = ReadObjectName(GetObjectPrompt(i + 1));
    }

}

void ReadLargestObjectNames() {

    cout << "\nReading Largest Objects\n";
    for (int i = 0; i < DefaultNumberOfObject; i++) {
        LargestObjects[i] = ReadObjectName(GetObjectPrompt(i+1));
    }

}

void PrintSmallestObjectNames() {
```

```
46
47      cout << SmallestObjectNotification;
48
49      for (int i = 0; i < DefaultNumberOfObject; i++) {
50      cout<< "\t" << SmallestObjects[i]<<"\n";
51      }
52
53  }
54
55  void PrintLargestObjectNames() {
56
57      cout << LargestObjectNotification;
58
59      for (int i = 0; i < DefaultNumberOfObject; i++) {
60          cout <<"\t"<< LargestObjects[i] << "\n";
61      }
62      cout << endl;
63  }
64
65
66  /*
67  *                                                                              ↵
      ------------------------------------------------------------------------- ↵
      ---------------------------------------
68  Problem Description:
69
70   In the project file, there is an image titled "image1" that illustrates   ↵
       two containers.
71   The first container is intended to store the names of the smallest        ↵
       objects, while the second container is meant for the largest objects.
72   Your task is to develop a program that reads and categorizes the names of ↵
        the smallest and largest objects,
73   correctly storing them in their respective containers.
74  ------------------------------------------------------------------------- ↵
      ---------------------------------------
75  Requirements:
76  Read the names of largest objects
77  Read the names of the smallest objects
78  Print the names of largest objects
79  print the names of smallest object
80
81  ------------------------------------------------------------------------- ↵
      -------------------------
82  Problem Analysis-:
83  Inputs:
84  -three smallest object's name
85  -three largest object's name
86
87  Outputs:
```

```
-printing the smallest and largest object's names

Constraints: unknown

Constant Values:
DefaultNumberOfObject =3
 ReadObjectNameNotification = "Enter the name of the object: ";
 ReadNumOfObjectNotification = "Enter the number of the objects ";
 SmallestObjectNotification = "\n\tSmallest object's Names :\n";
 LargestObjectNotification = "\n\t Largest object's Names :\n";


Decomposing problem:

Function To Read Only object name
Function to read the samllest names
function to read the largest
function to get a numbered format
function to print the largest objects
function to print the smallest objects

Design Phase:
Data Structures :
-SmallestObjects[DefaultNumberOfObject]
-LargestObjects[DefaultNumberOfObject];

Aglorithm Name: unknown
Time Complexity: O(1)
Space Complexity:
Solution Technique:

*/
int main() {
    ReadLargestObjectNames();
    ReadSmallestObjectNames();
    PrintLargestObjectNames();
    PrintSmallestObjectNames();
    return 0;
}
```