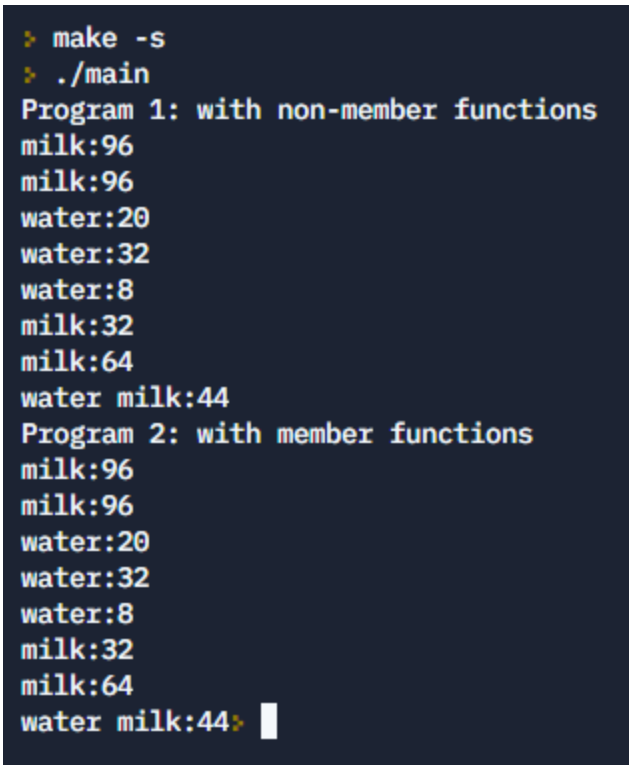

Main.cpp

```
// Kenry Yu
// Olena Bilinska
// Diego Garcia
// Demo 5:10PM
#include "Can.h"
#include "Can1.h"
#include <iostream>
using namespace std;

int main() {
  // Program 1
  cout << "Program 1: with non-member functions" << endl;
  Can c1("water", 12);
  Can c2("water", 20);
  Can c3 = c2 + c1; // c3nowhas32ouncesofwater
```

```
Can c4 = c2 - c1; // c4nowhas8ouncesofwater
Can c5("milk", 32);
Can c6("milk", 64);
Can c7;      // this will produce a can of air with 0 ounces using default
          // constructor
c7 = c1 + c5; // c7 will have 46 ounces of "mixed" the contents of c1 and c5
          // were not the same
c1 = c5 + c6; // c1 will now have 96 ounces of milk
cout << c1;   // output -> milk:96
cout << endl;
cout << c1 << '\n'
    << c2 << '\n'
    << c3 << '\n'
    << c4 << '\n'
    << c5 << '\n'
    << c6 << '\n'
    << c7; // prints all output on the same line

// Program 2
cout << "\nProgram 2: with member functions" << endl;
Can1 ac1("water", 12);
Can1 ac2("water", 20);
Can1 ac3 = ac2 + ac1; // c3 now has 32ounces of water
Can1 ac4 = ac2 - ac1; // c4 now has 8ounces of water
Can1 ac5("milk", 32);
Can1 ac6("milk", 64);
Can1 ac7;       // this will produce a can of air with 0 ounces using default
             // constructor
ac7 = ac1 + ac5; // c7 will have 46 ounces of "mixed" the contents of c1 and
```

```
                // c5 were not the same
  ac1 = ac5 + ac6; // c1 will now have 96 ounces of milk
  cout << ac1;     // output -> milk:96
  cout << endl;
  cout << ac1 << '\n'
      << ac2 << '\n'
      << ac3 << '\n'
      << ac4 << '\n'
      << ac5 << '\n'
      << ac6 << '\n'
      << ac7; // prints all output on the same line
  return 0;
}
```

```
Can.h // With non-member functions
#include <string>
#include <iostream>
using namespace std;
// non-member function
class Can {
private:
  string liquid;
  float ounces;

public:
  Can() : liquid("Empty"), ounces(0){};
  Can(string liq, float oz) : liquid(liq), ounces(oz){};
  friend Can operator+(Can, Can);
  friend Can operator-(Can, Can);
  friend ostream& operator<<(ostream&output, const Can&c);
};
```

Can.cpp

```
#include "Can.h"
using namespace std;
// non-member function
ostream &operator<<(ostream &out, const Can &right) {
  out << right.liquid << ":" << right.ounces;
  return out;
}

Can operator+(Can left, Can right) {
  Can temp;
  if (left.liquid == right.liquid)
    temp.liquid = left.liquid;
  else
    temp.liquid = left.liquid + " " + right.liquid;
  temp.ounces = left.ounces + right.ounces;
  return temp;
}

Can operator-(Can left, Can right) {
  Can temp;
  temp.liquid = left.liquid;
  temp.ounces = left.ounces - right.ounces;
  if (temp.ounces < 0)
    temp.ounces = 0;
  return temp;
}
```

```
Can1.h // The one with member functions
#include <string>
#include <iostream>
using namespace std;
// member function
class Can1 {
private:
  string liquid;
  float ounces;

public:
  Can1() : liquid("Empty"), ounces(0){};
  Can1(string liq, float oz) : liquid(liq), ounces(oz){};
  Can1 operator+(Can1);
  Can1 operator-(Can1);
  friend ostream& operator<<(ostream&output, const Can1&c);
};
```

Can1.cpp

```
#include "Can1.h"
using namespace std;
// member function
ostream &operator<<(ostream &out, const Can1 &right) {
  out << right.liquid << ":" << right.ounces;
  return out;
}

Can1 Can1::operator+(Can1 c2) {
  Can1 temp;
  if (this->liquid == c2.liquid)
    temp.liquid = this->liquid;
  else
    temp.liquid = this->liquid + " " + c2.liquid;
  temp.ounces = this->ounces + c2.ounces;
  return temp;
}

Can1 Can1::operator-(Can1 c2) {
  Can1 temp;
  temp.liquid = this->liquid;
  temp.ounces = this->ounces - c2.ounces;
  if (temp.ounces < 0)
    temp.ounces = 0;
  return temp;
}
```