ANSWER 1) [15 points]

```
Main begins
Constructor is called: Object a2
Constructor is called: Object a3


Function begins
Constructor is called: Object a1
Function ends


Destructor is called: Object a1
Destructor is called: Object a3
Main ends


Destructor is called: Object a2
```

ANSWER 2) [35 points]

a) [15 points]

```
class Consumer
{
public:
   int ID;
   string Name;
   int CompanyNum;
   int Consumption;
   Consumer() {}
   Consumer(int, string, int, int);
   void Display();
};

Consumer::Consumer(int id, string nm,
                   int cn, int co)
{
        ID = id;
        Name = nm;
        CompanyNum = cn;
        Consumption = co;
}

void Consumer::Display()
{
    cout << left << setw(5) << ID
         << "  " << setw(13)
         << Name << "  "
         << Consumption << endl;
}
```

b) [20 points]

```
#include <iostream>
#include <iomanip>
#include <fstream> //File stream
using namespace std;


//Globals
const int C = 4;    //Number of companies
const int N = 100; //Max number of consumers


string companies[C] = {"ALCATEL", "FOXCONN",
                       "GATEWAY", "HUAWEI"};
int main () {
  Consumer liste[N];
  int i=0;  //Counter for records in file
  int id;
  string nm;
  int cn;
  int co;

  ifstream  dosya;
  dosya.open("CONSUMERS.TXT");

  if (! dosya.is_open()) {
      cout << "File can not be opened \n";
      return 0;
  }

  while (! dosya.eof() ) {
    dosya >> id >> nm >> cn >> co;
    liste[i] = Consumer(id, nm, cn, co);
    if (dosya.eof()) break;
    i++;
  }
  dosya.close();


  //Calculate Company averages
  for (int j=0; j<C; j++)   {
    int sum = 0, counter = 0;
    cout << "COMPANY NAME : " << companies[j] << endl;
    cout << "ID     ConsumerName   Consumption" << endl;


    for (int k=0; k<i; k++) {
       if (liste[k].CompanyNum == j+1 ) {
              sum += liste[k].Consumption;
              counter++;
              liste[k].Display();
          }
    } //End k


    cout << "COMPANY CONSUMPTION AVERAGE = "
         << (float)sum/counter << endl;
  } //End j
}  //End of main
```

ANSWER 3) [50 points]

a) [15 points]

```
class Point
{
public:
   int x, y;
   Point(int a, int b) : x(a), y(b) {}
};

class Rectangle
{
public:
   Point  P1;    // Left-lower coordinates
   Point  P2;    // Right-upper coordinates
   // Constructor
   Rectangle(Point n1, Point n2) : P1(n1), P2(n2) {};
   int Area();
   void Print();
};

int  Rectangle :: Area()
{
  int alan = abs(P1.x - P2.x) * abs(P1.y - P2.y);
  return alan;
}

void  Rectangle :: Print()
{
  cout << "Left-lower coordinates : (" << P1.x
       << " , "  << P1.y << ")" << endl;
  cout << "Right-upper coordinates : (" << P2.x
       << " , "  << P2.y << ")" << endl;
  cout << "Area = " << Area() << endl;
}
```

b) [10 points]

```
// Union operator
int  operator| (Rectangle R1, Rectangle R2)
{
        int area1 = R1.Area();
        int area2 = R2.Area();
        int ia = R1 & R2; //Call intersection
        return (area1 + area2 - ia);
} //End of function
```

c) [10 points]

```
// Intersection operator
int operator& (Rectangle R1, Rectangle R2)
{
        // Calculate coordinates of intersection points.
        int left_x   = max(R1.P1.x, R2.P1.x);
        int lower_y  = max(R1.P1.y, R2.P1.y);
        int right_x  = min(R1.P2.x, R2.P2.x);
        int upper_y  = min(R1.P2.y, R2.P2.y);


        // Check whether there is intersection.
        if (left_x < right_x && lower_y < upper_y)
        {
           // Define an intersection rectangle.
           Rectangle R( Point(left_x, lower_y) ,
                        Point(right_x, upper_y) );
           return R.Area();
        }


        cout << "No intersection." << endl;
        return 0;
} //End of function
```

d) [15 points]

```
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
        Rectangle  R1(Point(0, 0), Point(20, 10) );
        Rectangle  R2(Point(5, 4), Point(25, 15) );


        cout << "Rectangle R1:" << endl;
        R1.Print();


        cout << "Rectangle R2:" << endl;
        R2.Print();


        int  intersection_area = R1 & R2;
        cout << "Area of Intersection (R1 & R2) = "
             << intersection_area << endl;


        int  union_area = R1 | R2;
        cout << "Area of Union (R1 | R2) = "
             << union_area << endl;
} //End of main
```