# EHB354E - MIDTERM EXAM SOLUTIONS

## ANSWER 1) [40 points]

### 1a ) [25 points]

```
#include <iostream>
using namespace std;
#define SEMBOL "*"

class Rectangle
{         protected:
          int vlength, hlength;

          public:
          Rectangle (int vl, int hl); //constructor
          void display();
};
//------------------------------------------------------------------------------
Rectangle :: Rectangle (int vl, int hl)
{
   if (vl <= 0 ||  hl <= 0  || hl > 80 ||  vl > 40 )
   {
     cout << "INVALID LENGTHS IN CONSTRUCTOR :  ";
     cout << "(Vertical=" << vl << " ,
                Horizontal=" << hl << ") \n";
     vlength = 1;   hlength = 1;
 }
 else
 {
   vlength = vl;   hlength = hl;
 }
}
//------------------------------------------------------------------------------
void Rectangle::display()
{
   // Print the lengths info on screen.
    cout << "\nRECTANGLE ";
    cout << "(Vertical=" << vlength << " ,
           Horizontal=" << hlength << ") \n";

   // Print the rectangle as non-filled frame.
    for (int i=1; i<= vlength; i++)
    {
      cout << SEMBOL;

      for (int j=2; j <= hlength-1; j++)
          if (i > 1 && i < vlength)  cout << " ";
          else   cout << SEMBOL;

      if (hlength > 1)   cout << SEMBOL;
      cout << endl;
   }
    cout << "\n\n";
}
```

```
class FilledRectangle : public Rectangle
{
          public:
          FilledRectangle (int vlength, int hlength)
               : Rectangle (vlength, hlength) {}
          void display();
};
//-----------------------------------------------------------
void FilledRectangle::display()
{
   // Print the lengths info on screen.
    cout << "\nFILLED RECTANGLE ";
    cout << "(Vertical=" << vlength << " ,
          Horizontal=" << hlength << ") \n";

   // Print the rectangle as solid-filled.
    for (int i=1; i<= vlength; i++)
    {
          for (int j=1; j <= hlength; j++)
              cout << SEMBOL;
          cout << "\n";
    }
    cout << "\n\n";
}

//-----------------------------------------------------------
```

### 1b ) [15 points]

```
int main()
{
Rectangle  R1 (10, 10);
Rectangle  R2 (5, 20);
Rectangle  R3 (0, 0);
Rectangle  R4 (1, 10);
R1.display();
R2.display();
R3.display();
R4.display();

FilledRectangle  FR1 (15, 8);
FilledRectangle  FR2 (6, 13);
FilledRectangle  FR3 (-1, -1);
FR1.display();
FR2.display();
FR3.display();

cout << "\nPROGRAM FINISHED.\n";
}
```

**ANSWER 2)** [35 points]

**2a )** [15 points]

```
#include <iostream>
using namespace std;
#define N 10

Collection :: Collection()
{
        for (int i=0; i < N; i++)
           liste[i] = "";
}
//-------------------------------------------------------------------
void  Collection :: operator+ (string newitem)
{
        for (int i=0; i < N; i++)
        {
                if ( liste[i] == "" )
                {
                  liste[i] = newitem;
                  return;
                }
        }
}
//-------------------------------------------------------------------
bool  Collection :: operator== (Collection  other)
{
        for (int i=0; i < N; i++)
                if ( this->liste[i] != other.liste[i] )
                  return false;

   return true;
}
```

**2b )** [10 points]

```
// Nonmember friend function
void  operator<<  (ostream& cihaz, Collection  col)
{
        cout << "Items in collection : ";
        for (int i=0; i < N; i++)
        {
                if ( col.liste[i] != "" )
                  cihaz << col.liste[i] << "  ";
        }
   cihaz << endl;
}


//---------------------------------------------------------------------
```

**2c )** [10 points]

```
int main()
{
  Collection C1, C2;

  C1+"Apple";
  C1+"Orange";
  C1+"Grape";

  C2+"Apple";
  C2+"Kiwi";

  cout << C1;
  cout << C2;

  if (C1 == C2)
    cout << "Collections equal\n";
  else
    cout << "Collections not equal\n";
}
```

**ANSWER 3)** [25 points]

```
#include <iostream>
#include <fstream>
#include <cstring> // strcpy,  strcat
using namespace std;

int main (int argc, char * argv [] )
{
 try
 {
   if ( argc < 3)
     throw ("Error: At least two filenames required");

   ofstream  outputdosya ("output.txt", ios::out);
   if ( ! outputdosya )
     throw "Error : Output file could not be opened";

// Read input files contents, and write to output file.
   for (int i = 1; i < argc; i++)
   {
    ifstream  inputdosya;
    inputdosya.open( argv[i] );

    if (! inputdosya.is_open() ) {
      char mesaj[50];
      strcpy (mesaj, "Error : Input file ");
      strcat (mesaj, argv[i] );
      strcat (mesaj, " could not be opened");
      throw mesaj;
    }

    char satir[100];
    while ( ! inputdosya.eof() )
    {
      inputdosya.getline (satir, 100);
      outputdosya << satir << endl;
    }

    inputdosya.close();
    cout << "Appended file : " << argv[i] << endl;
   } // end of for loop

   outputdosya.close();
   cout << "Program finished successfully.\n";
 } // end of try block

 catch (char const * msg)
 {
   cout << msg << endl;
   cout << "Program finished with throw error.\n";
 }

} // end of main
```