BLG252E – Object Oriented Programming
Final Exam Answers

QUESTION 1) [25 points]

a) [15 points]

```
#define MAX 20
template <class mytype>
class MyIterator {
  mytype data[MAX]; // Generic array
  int N;            // Number of elements in array
  int current;  // Index of current element

  public:
   MyIterator(mytype data_in[], int N_in) {
      N = N_in;
       for (int i=0; i<N; i++)  data[i] = data_in[i];
       current=0;
  }

   bool hasNext() {return (current < N);}
   mytype getNext() {return data[current++];}
};

class Complex {
  public:   int re, im;
};
```

b) [10 points]

```
int main() {
  string   s[] = {"Apple", "Orange", "Grape", "Cherry", "Mango"};
  int      a[] = {10,20,30,40};
  Complex  c[] = { {-2,4}, {5,-6}, {0,3} };

  MyIterator <string>   X(s, 5);
  MyIterator <int>       Y(a, 4);
  MyIterator <Complex>  Z(c, 3);

  while (X.hasNext() )  cout << X.getNext() << "  ";
  cout << endl;

  while (Y.hasNext() )  cout << Y.getNext() << "  ";
  cout << endl;

  while (Z.hasNext() ) {
    Complex tmp = Z.getNext();
    cout << "(" << tmp.re << "," << tmp.im << ")  ";
  }

  return 0; }
```

QUESTION 2) [25 points]

a) [20 points]

```
enum { LOCKED, VALIDATION, UNLOCKED, MAXSTATES };
enum { COIN, PUSH, VALID, INVALID, TIMEOUT, MAXEVENTS };

string states[MAXSTATES] = {"LOCKED", "VALIDATION", "UNLOCKED"};
string events[MAXEVENTS] = {"COIN", "PUSH", "VALID", "INVALID",
"TIMEOUT"};

class Turnstile {
  int current_state;
public:
  Turnstile() {current_state = LOCKED;}

 void process(int e)
{
  cout << "State = " << states[current_state]
         << "  Event = " << events[e];
    try {
    switch (current_state) {
      case LOCKED :
                switch (e) {
                case COIN : current_state = VALIDATION; break;
                case PUSH : break;
                default : throw (e);
                }
       break;
case VALIDATION :
                switch (e) {
                case VALID : current_state = UNLOCKED; break;
                case INVALID : current_state = LOCKED; break;
                default : throw (e);
                }
       break;

      case UNLOCKED :
                switch (e) {
                case TIMEOUT :
                case PUSH :
                      current_state = LOCKED; break;
                default : throw (e);
                }
   } // end of outer switch
   } // end of try block
   catch (int e) {
             cout << "\t(Event is not applicable!)";
   }
   cout << endl;
  }  // end of process function
};   // end of class
```

QUESTION 2-b) [5 points]

```
int main() {
  int event;
  Turnstile t;

  srand(time(NULL));
  while (true)  { // Infinite loop simulation
    event = rand()% MAXEVENTS;
    t.process(event);
  }
  return 0;
}
```

QUESTION 3-c) [10 points]

```
int main() {
  Collection K("ITU Library");
  char response;
  while (1) {
    cout << "Enter publication type (B=book, M=magazine, J=journal) or E to exit : ";
    cin>> response;
    if (response == 'E' || response == 'e') break;
    switch (response) {
              case 'B' : case 'b' :     K.add(new Book); break;
              case 'M' : case 'm' :  K.add(new Magazine); break;
              case 'J' : case 'j' :     K.add(new Journal); break;
              default : cout << "Invalid response\n";
    }
  }
   K.print();
  return 0;
}
```

QUESTION 3) [50 points]

b) [30 points]

```
#include <vector>
//**************************
class Publication {
string title;
float price;
public:
Publication() {
  cout << "Publication\n";
  cout<<"Enter title: "; cin>>title;
  cout<<"Enter price: "; cin>>price;
}

virtual void print(){
  cout<<"Title ="<<title
        <<" Price ="<<price<<endl;
}
};
//*************************
class Book : virtual public Publication
{
int isbn;
public:
Book() {
  cout << "Book\n";
  cout<<"Enter isbn number: ";
  cin>>isbn;
}

void print(){
  Publication::print();
  cout<<"Isbn ="<<isbn<<endl;
}
};
//**************************
```

```
//**************************
class Magazine : virtual public Publication {
char period;
public:
Magazine() {
  cout << "Magazine\n";
  cout<<"Enter period (W=weekly, M=monthly): ";
  cin>>period;
}

void print(){
  Publication::print();
  cout<<"period ="<<period<<endl;
}
};
//**************************
class Journal : virtual public Book,
                   virtual public Magazine {
string subject_name;
public:
Journal() {
  cout << "Journal\n";
  cout<<"Enter subject_name : ";
  cin>>subject_name;
  }

void print(){
  Book::print();
  Magazine::print();
  cout<<"subject_name ="
        <<subject_name<<endl;
}
};
//**************************
```

```
//**************************
class Collection {
string owner_name;
vector<Publication*> items;
public:
Collection(string s) {owner_name = s;}

void print(){
   cout<<"Collection owner ="
        <<owner_name<<endl;
   if (items.size()==0) {
     cout << "Warning : Collection is
              empty!\n";
     return;
    }

   for(int i = 0; i < items.size(); i++ ) {
    cout << items[i]->print();
    cout << "----------------\n";
   }
}

void add(Publication * pub) {
    items.push_back(pub);
}

};
//**************************
```