Distributed System Lab
Assignment - 8

Ashish Verma

20204041

CS - A

```
#include "Quiz.hh"
#include <iostream>

/** Name is defined in the server */
#define SERVER_NAME "Quiz"

Quiz::QuizServer_ptr service_server;

using namespace std;

void insert_question(const char* sentence, int numAnswers,
Quiz::Answer** answers, int numCorrectAnswers, CORBA::Char*
correctAnswers);
void create_questions();

int main(int argc, char ** argv)
{
  try {
 //--------------------------------------------------------------------
 // Initialize ORB object.
 //--------------------------------------------------------------------
CORBA::ORB_ptr orb = CORBA::ORB_init(argc, argv);

//--------------------------------------------------------------------
// Resolve service
//--------------------------------------------------------------------
service_server = 0;

 try {
```

```
//-----------------------------------------------------------------------
// Bind ORB object to name service object.
// (Reference to Name service root context.)
//-----------------------------------------------------------------------
CORBA::Object_var ns_obj = orb-
>resolve_initial_references("NameService");

if (!CORBA::is_nil(ns_obj)) {
//-----------------------------------------------------------------------
// Bind ORB object to name service object.
// (Reference to Name service root context.)
//-----------------------------------------------------------------------
CosNaming::NamingContext_ptr nc =
CosNaming::NamingContext::_narrow(ns_obj);
//-----------------------------------------------------------------------
// The "name text" put forth by CORBA server in name service.
// This same name ("MyServerName") is used by the CORBA server when
// binding to the name server (CosNaming::Name).
//-----------------------------------------------------------------------
CosNaming::Name name;
name.length(1);
name[0].id = CORBA::string_dup(SERVER_NAME);
name[0].kind = CORBA::string_dup("");

//-----------------------------------------------------------------------
// Resolve "name text" identifier to an object reference.
//-----------------------------------------------------------------------
CORBA::Object_ptr obj = nc->resolve(name);

if (!CORBA::is_nil(obj)) {
service_server = Quiz::QuizServer::_narrow(obj);
}
}
} catch (CosNaming::NamingContext::NotFound &) {
cerr << "Caught corba not found" << endl;
 } catch (CosNaming::NamingContext::InvalidName &) {
cerr << "Caught corba invalid name" << endl;
} catch (CosNaming::NamingContext::CannotProceed &) {
cerr << "Caught corba cannot proceed" << endl;
}

//-----------------------------------------------------------------------
// Do stuff
//-----------------------------------------------------------------------
if (!CORBA::is_nil(service_server)) {
 cout << "QuizClient client is running ..." << endl;
```

```
 orb->register_value_factory("IDL:Quiz/Answer:1.0", new
Quiz::Answer_init());

                        create_questions();

                        //
                        // get random question
                        //
                        orb-
>register_value_factory("IDL:Quiz/Question:1.0", new
Quiz::Question_init());

                        Quiz::Question* received_question = new
OBV_Quiz::Question();
                        service_server->getQuestion(received_question);
                        const char* received_question_sentence =
received_question->sentence();
                        CORBA::Long received_question_id =
received_question->id();
                        Quiz::Question::AnswerSeq
received_question_answers = received_question->answers();
                        int numAnswers =
received_question_answers.length();

                        cout << "Received Question: id=" <<
received_question_id << ", sentence=" << received_question_sentence <<
endl;

                        for(int i = 0; i < numAnswers; i++) {
                            if(received_question_answers[i]) {
                                cout << "\t" <<
received_question_answers[i]->id() << ": " <<
received_question_answers[i]->sentence() << endl;
                            }
                        }
        }

        //---------------------------------------------------------------
-----------
        // Destroy OBR
        //---------------------------------------------------------------
-----------
        orb->destroy();

    } catch (CORBA::UNKNOWN) {
        cerr << "Caught CORBA exception: unknown exception" << endl;
    }
}
```

```
void insert_question(const char* sentence, int numAnswers,
Quiz::Answer** answers, int numCorrectAnswers, CORBA::Char*
correctAnswers)
{
        Quiz::Question::AnswerSeq* answersSeq = new
OBV_Quiz::Question::AnswerSeq(numAnswers, numAnswers, answers, 1);
        Quiz::CompleteQuestion::CharSeq* correctAnswersSeq = new
OBV_Quiz::CompleteQuestion::CharSeq(numCorrectAnswers,
numCorrectAnswers, correctAnswers, 1);
        Quiz::CompleteQuestion* new_question = new
OBV_Quiz::CompleteQuestion(0, sentence, *answersSeq,
*correctAnswersSeq);

    CORBA::Long question_received_id = service_server-
>insertQuestion(new_question);
        cout << "send question and received id " <<
question_received_id << endl;
}

void create_questions()
{
    // create first question
        const char* question_sentence = "It applies to a software layer
that provides a programming abstraction as well as masking the
heterogeneity of the underlying networks, hardware, operating systems
and programming languages. What is it?";
        Quiz::Answer** question0_answers = new Quiz::Answer*[3];
        question0_answers[0] = new OBV_Quiz::Answer('a',
"Hetereogenity");
        question0_answers[1] = new OBV_Quiz::Answer('b', "Middleware");
        question0_answers[2] = new OBV_Quiz::Answer('c', "Opennes");
        CORBA::Char question0_correctAnswers[] = {'b'};
        insert_question(question_sentence, 3, question0_answers, 1,
question0_correctAnswers);

        // create second question
        question_sentence = "It refers to a running program (a process)
on a networked computer that accepts requests from programs running on
other computers to perform a service and responds appropriately.";
        Quiz::Answer** question1_answers = new Quiz::Answer*[3];
        question1_answers[0] = new OBV_Quiz::Answer('a', "Server");
        question1_answers[1] = new OBV_Quiz::Answer('b', "Middleware");
        question1_answers[2] = new OBV_Quiz::Answer('c', "Client");
        CORBA::Char question1_correctAnswers[] = {'a'};
        insert_question(question_sentence, 3, question1_answers, 1,
question1_correctAnswers);
```