A Project Based CS/IS-1 Course with an Active Learning Environment

Dr. Suvineetha Herath*, Dr. Ajantha Herath*, Mr. Mohammed A.R. Siddiqui*, Mr. Khuzaima AH. El-Jallad*
*ACMSIG Teaching & Learning Group
University of Bahrain
Kingdom of Bahrain

***Abstract*—High level programming languages use system defined data types and the user defined data types in computations. We have developed a project-based CS/IS-1 course to substitute the traditional lecture based classroom to help students design and use at least two user defined data types in their computations to solve real world problems. Abstract data types and basic programming constructs are introduced efficiently to the students in an active learning environment using games. To assess and evaluate the changes made we distributed the course module among our students and other instructors. This paper describes our experience in developing the project based active learning environment.**

Keywords-I/O; arithmetic expressions; if-else and switch conditional operations; for-while iterative computation; inheritance; polymorphism; recursion; searching and sorting algorithms; IDE.

I. INTRODUCTION

The objectives of this course are to introduce computer science fundamentals using C++, provide the ability to design algorithms to solve problems and implement those using test and document programs. The course focuses on software engineering process and techniques. Student outcomes at the end of the course are shown below:

They will be able to:

1) Design, implement, test and document application programs of simple to moderate complexity that use two or more user defined types and variables, classes or abstract data types, and one main program.

2) Apply basic programming constructs such as sequence, selection, iteration, functions in complete, syntactically correct programs.

3) Communicate both technical and non-technical aspects of student's work in formal and informal settings.

4) Develop programs in multiple stages; use stubs to test the system as a whole; use drivers to carry out unit testing for functional components and for data abstractions.

5) Maintain a record of time devoted to the component tasks in the completion of programming projects.

There is both local and national need for high-quality trained programmers with the ability to learn in a short period of time and stay current with the information technology advances. Colleges can help to meet this demand by redesigning their programming courses allowing more students to succeed at the entry level. Successful application of active learning with rapidly changing technologies in the learning process is a way to remove the difficulties at entry level. Such changes will help the students to improve their skills. Consequently, both the public and private sectors will benefit with the greater number of highly-skilled trained programmers.

We identified many deficiencies in teaching a traditional lecture based CS/IS-1 class and the importance of transforming the course into a project based e-learning environment. The lecture based environment is relatively passive. In addition, we want to connect this class to other classes such as computer architecture, digital design, operating systems, security, parallel computing, and importantly the needs of industry. An effective computer programming class could provide these links. From our initial planning input we took actions to transform the course to a project based one and then collected results. Feedback from the action stage and results are used to improve the plan. We continuously improved the quality of instruction and actions while using the feedback received from the students, and took action to transform the course suitable for e-learning. A course with the features outlined above could help our students develop design skills in several different languages before their graduation.

The following sections outline the details of research model, actions taken, course plan, goals achieved, difficulties encountered, course assessment, plan for future work and summary.

II. RESEARCH MODEL

Over the years we have reviewed many textbooks and papers, and attended workshops and conferences [1-5]. We continuously improved our offerings by making changes to the course, collecting data and evaluating them, and using the feedback for further improvements. There are many good books published on C++ programming. Malik authored a very good book for this class and published the seventh edition by Cengage learning [2]. Gaddis' C++ programming with objects first, published by Pearson is also another very good book with many supplements for this class [3]. Deitel's C++ programming, published by Pearson is also another very good book with many supplements for this class [4]. Horstmann has also written a good C++ Programming book published by Wiley. Because of the simplicity and the clustered sets of programming examples and problems presented in the book we selected Liang's C++ programming second edition published by Pearson as the main text for our class [1].

At the beginning of this course we introduced number systems and simple game designs using binary numbers. Thereafter, we included random number generators and simulations with basic programming constructs as open and close project laboratories. We have integrated security-related concepts into the course. At the end, we incorporated modules to help the students understand the importance of parallelism in enhancing performance of sorting algorithms and its benefits as an application programmer, a systems programmer, an algorithm designer, and a computer architect.

Topics presented in this course include the implementation of basic programming constructs such as I/O, arithmetic expressions, if-else and switch conditional operations, for-while iterative computation controls, simple functions, classes, inheritance, polymorphism, recursion, searching and sorting algorithms. This course includes twenty eight lecture modules, fifteen hands-on project activities. Several examples of classroom activities are given below.

III. TRANSFORMATIONS-ACTIONS TAKEN

We have replaced the old text based programming style to Graphical User Interface (GUI). Students downloaded the free online and open software (FOSS) Code Blocks IDE from http://www.codeblocks.org/downloads/26/ to complete the project activities using Windows 7 operating system. [6]. The following paragraphs describe the projects assigned with sample solutions completed by the students from simple to moderate complexity.

A. Project 1:

The following code 1, depicts the first classroom project assigned to illustrate a simple vending machine. The goal of this particular activity is to introduce the concepts of cout, cin, << , >>, and simple arithmetic operations. Towards the end of the class this project is extended to demonstrate the design and the use of multiple classes.

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

int main()
{
   int juice_money, cent_85, cent_1;
   cout << "This juice machine accepts Dollar bills.\n";
   cout << "How many dollars will you insert?\n";
   cin >> juice_money;

   // Find the number of 85 cent juice cans to be dispensed.
   cent_85 = 100 * juice_money / 85;

   // Find the number of pennies to be dispensed.
   cent_1 = 100 * juice_money % 85;

   //Tell the user how many juice cans they will get.
   cout << "The juice vendor will dispence:\n";
   cout << "1) " << cent_85 << " 85 cent juice cans.\n";
   cout << "2) " << cent_1  << " Pennies.\n";
   return 0;
}
```

Code 1. I/O Project

B. Project 2:

In this project students design and implement a game based on binary number to decimal conversion that is able to produce a number from 1 to 31 and hence any individual's birthday [1]. First the binary number system is introduced with binary to decimal conversions. Students were allowed to discover that any decimal number from 1 to 31 can be represented with five bits and to identify the corresponding five groups. See Code 2.

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

int main()
{
   //  Initialize 6 input/output variables
   int bit1, bit2,bit3, bit4, bit5, day;

   cout << "I can guess your birthday.\n";
   cout << "Remember to put a 1 for yes and a 0 for no.\n";

   cout << "Is your birth day in set 1?\n";
   cout << "Set 1: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29,
31\n";
   cin >> bit1;

   cout << endl << "Is your birth day in set 2?\n";
   cout << "Set 2: 2, 3, 6, 7, 10, 11, 14, 15, 18, 19, 22, 23, 26, 27, 30,
31\n";
   cin >> bit2;

   cout << endl << "Is your birth day in set 3?\n";
   cout << "Set 3: 4, 5, 6, 7, 12, 13, 14, 15, 20, 21, 22, 23, 28, 29, 30,
31\n";
   cin >> bit3;

   cout << endl << "Is your birth day in set 4?\n";
   cout << "Set 4: 8, 9, 10, 11, 12, 13, 14, 15, 24, 25, 26, 27, 28, 29, 30,
31\n";
   cin >> bit4;

   cout << endl << "Is your birth day in set 5?\n";
   cout << "Set 5: 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
31\n";
   cin >> bit5;

   day = bit5*16 + bit4*8 + bit3*4 + bit2*2 + bit1;

  cout << endl << "Your birth day is on the " << day << " day of your
birth month.";
   return 0;
}
```

Code 2. Binary to Decimal Conversion game

C. Project 3:

Conditional if else switch statements are central to writing computer programs and aimed at helping students understand and solve intractable problems. In this project students were introduced to random number generation and asked to design a game that simulates a lottery drawing [1]. They implemented the game for one, two and three digit lottery numbers, with prizes awarded to the winning numbers.

The program first invites the user to enter the number corresponding to the version of the game they are playing. Once the user inputs the numbers for their lottery ticket, the computer generates a random number and compares to see a match. At the end of the program it displays the winning award.

In the one digit game, a simple if/then statement matches the random number generated by the program for the lottery to the users guess. The random number is generated by the rand() mod 10 to ensure one digit. See Code 3.

```
#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;

int main()
{
   srand (time(0));
   int lottery = rand() % 10, guess, prize =1;
   cout << "Would you like to play the lottery and win?\n";
   cout << "Please enter any number from 0 to 9 to find out if you
win!!\n";
   cin >> guess;
   if (lottery == guess)
         prize = 1000;
   else prize = 0;
   if (prize >0)
      cout << "Congratulations you have won " << prize << " dollars\n"
<<"The correct number was " <<lottery<< endl;
   else cout << "Sorry you have not won this time.\n" <<"The correct
number was " <<lottery<<". Please try again.";
   return 0;
}
```

Code 3. One digit game program

In the two digit game, the two numbers need to be separated from one another. There are four different winning outcomes and each will be determined by an if/then statement. The random integer, generated by the rand() mod 100 ensures two digits, from 00 to 99. The division and modulus operators are used to separate the first and second digits. See Code 4.

```
#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;

int main()
{
   srand (time(0));
   int lottery = rand() % 100, firstdig, seconddig, guess, firstguess,
secondguess;
   int prize = 1;
   firstdig=lottery/10;
   seconddig=lottery % 10;
   cout << "Would you like to play the lottery and win?\n";
   cout << "Please enter any number from 00 to 99 to find out if you
win!!\n";
   cin >> guess;
   firstguess=guess/10;
   secondguess=guess % 10;
   if ((firstdig == firstguess) && (seconddig == secondguess))
         prize = 10000;
   else if ((firstdig == firstguess) && (seconddig != secondguess))
         prize = 5000;
   else if ((firstdig != firstguess) && (seconddig == secondguess))
         prize = 5000;
   else if ((firstdig == secondguess) && (seconddig == firstguess))
         prize = 7500;
   else prize = 0;

   if (prize >0)
      cout << "Congratulations you have won " << prize << " dollars\n"
<<"The correct number was " <<lottery<< endl;
   else cout << "Sorry you have not won this time.\n" <<"The correct
number was " <<lottery<<". Please try again.";
   return 0;
}
```

Code 4. Two digit game program

In the three digit game, the rand() mod 1000 ensures three digits, ranging from 000 to 999. The largest amount will be awarded to the users who guess all three numbers in the correct order. In this program we limited the number of matches to produce eight potential outcomes, having one number right in any of the three locations (three outcomes), having two of the numbers correct and in the proper location (three more outcomes), having all three numbers correct but in the wrong order, and having all three numbers correct in the correct order. See Code 5.

```
#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;

int main()
{
   srand (time(0));
   int lottery = rand() % 1000, firstdig, seconddig, thirddig, guess,
firstguess, secondguess, thirdguess;
   int prize = 1;
   firstdig=lottery/100;
   seconddig=lottery % 100 / 10;
   thirddig=lottery % 100 % 10;
   cout << "Would you like to play the lottery and win?\n";
   cout << "Please enter any number from 000 to 999 to find out if you
win!!\n";
   cin >> guess;
   firstguess=guess/100;
   secondguess=guess % 100 / 10;
   thirdguess=guess % 100 % 10;
   if ((firstdig == firstguess) && (seconddig == secondguess) &&
(thirddig == thirdguess))
         prize = 100000;
   else if ((firstdig == firstguess) && (seconddig == secondguess) &&
(thirddig != thirdguess))
         prize = 4000;
   else if ((firstdig == firstguess) && (seconddig != secondguess) &&
(thirddig == thirdguess))
         prize = 4000;
   else if ((firstdig != firstguess) && (seconddig == secondguess) &&
(thirddig == thirdguess))
         prize = 4000;
   else if (((firstdig == secondguess) || (firstdig == thirdguess)) &&
((seconddig == firstguess) || (seconddig == thirdguess)) && ((thirddig ==
secondguess) || (thirddig == firstguess)))
         prize = 25000;
   else if ((firstdig == firstguess) && (seconddig != secondguess) &&
(thirddig != thirdguess))
         prize = 1000;
   else if ((firstdig != firstguess) && (seconddig == secondguess) &&
(thirddig != thirdguess))
         prize = 1000;
   else if ((firstdig != firstguess) && (seconddig != secondguess) &&
(thirddig == thirdguess))
         prize = 1000;
   if (prize >0)
      cout << "Congratulations you have won " << prize << " dollars\n"
<<"The correct number was " <<lottery<< endl;
   else cout << "Sorry you have not won this time.\n" <<"The correct
number was " <<lottery<<". Please try again.";
   return 0;
}
```

Code 5. Three digit game program

D. Project 4:

In this project students simulate a rock, paper, scissors game [1]. The program asks the user to choose one of the options to play the game. After that, the computer will determine its choice and decides the outcome of the game. A number mod 3 generates three possibilities 0, 1 or 2, with each number being assigned a value: rock, paper or scissors.

A switch will take the computer's choice and compare it to the user's choice using an if/then statement. If the two choices are the same, the game will end up tied. .

The else part will give one of the two alternative scenarios for the game in another if statement and the result of that outcome, with a final else being the result of the only other potential outcome for the scenario, since only three exist for each possibility. See Code 6.

```
#include <iostream>
    #include <ctime>
    #include <cstdlib>
    using namespace std;

    int main()
    {
       srand (time(0));
       int number = rand() % 3;
       string playerguess, cpuoption;
       cout << "Would you like to play rock/paper/scissors?\n";
       cout << "Please choose whether you would like the rock, paper, or
scissors.\n";
       cin >> playerguess;
       if (number == 0)
          cpuoption = "rock";
       else if (number == 1)
          cpuoption = "paper";
       else if (number == 2)
          cpuoption = "scissors";
       switch (number)
       {
          case 0:  if (playerguess == cpuoption)
                      cout << "You have tied.  Please play again.\n";
                      else if (playerguess == "paper" )
                      cout << "You have won.  Paper covers rock.\n";
                      else cout << "You have lost.  Rock breaks scissors.\n";
break;
          case 1: if (playerguess == cpuoption)
                      cout << "You have tied.  Please play again.\n";
                      else if (playerguess == "scissors" )
                      cout << "You have won.  Scissors cuts paper.\n";
                      else cout << "You have lost.  Paper covers rock.\n"; break;
          case 2:  if (playerguess == cpuoption)
                         cout << "You have tied.  Please play again.\n";
                         else if (playerguess == "rock" )
                         cout << "You have won.  Rock breaks scissors.\n";
                         else cout << "You have lost.  Scissors cut paper.\n";
break;
       }
       return 0;
    }
```

Code 6. Rock, Paper, Scissor

E. Project 5:

In this project students use at least three functions to design and implement a game that simulates the actions of an ATM machine, where the user initially has no money, then makes an initial deposit, and then is able to withdraw money.

The program asks the user to insert any number of quarters, dimes, nickels and pennies. Then the program tells the user the available balance. The same process will be repeated for the withdrawal, and the final balance will be displayed. There are three main functions in this program. First, is the initialization of the variables, setting the values to zero, as if this is the first time that the account is filled with money, or another programmer defines a different value other than zero that would be available for withdrawal. The second function is a deposit function to add up the coins entered into the purse that would give the user a total amount of money deposited. The third function enables withdrawal from the account. If the withdrawal exceeds the account balance the program will state that there are insufficient funds. See Code 7.

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

    void insert(int quarters, int dimes, int nickels, int pennies, double& total);
    void withdrawl(int quarters, int dimes, int nickels, int pennies, double&
total);
    void initialize (int& quarters, int& dimes, int& nickels, int& pennies,
double& total);

    int main()
    {
       int quarters, dimes, nickels, pennies;
       double total;
       initialize(quarters, dimes, nickels, pennies, total);
       insert(quarters, dimes, nickels, pennies, total);
       withdrawl(quarters, dimes, nickels, pennies, total);
       return 0;
    }

    void insert(int quarters, int dimes, int nickels, int pennies, double& total)
    {

       cout << "Please insert how many quarters you are inserting.\n";
       cin >> quarters;
       cout << "Please insert how many dimes you are inserting.\n";
       cin >> dimes;
       cout << "Please insert how many nickels you are inserting.\n";
       cin >> nickels;
       cout << "Please insert how many pennies you are inserting.\n";
       cin >> pennies;
       total = quarters*.25+dimes * .10 + nickels * .05 + pennies * .01;
    }

    void withdrawl(int quarters, int dimes, int nickels, int pennies, double&
total)
    {

       cout << "You have $" << total << " that you can withdraw\n";
       cout << "How many quarters do you want to take out?\n";
       cin >> quarters;
       cout << "How many dimes do you want to take out?\n";
       cin >> dimes;
       cout << "How many nickels do you want to take out?\n";
       cin >> nickels;
       cout << "How many pennies do you want to take out?\n";
       cin >> pennies;
       total = total - (quarters*.25+dimes*.10+nickels*.05+pennies*.01);
       if (total>0)
          cout << "Please take your change.  Your new balance is $" <<total;
       else cout <<"You do not have enough funds.  Please try again.";
    }
```

```
    void initialize (int& quarters, int& dimes, int& nickels, int& pennies,
double& total)
    {
       quarters = 0, dimes = 0, nickels =0, pennies = 0;
       total = 0;
    }
```

Code 7. ATM Machine

F. Project 6:

In this project students define two classes and design and implement a program that simulates a juice vending machine. The program asks the user to select the juice choice from the vending machine [2]. Then the program asks for the proper amount of money for the kind of juice they selected.

The program gives the user two chances to input at least the proper amount of money before returning their money and displaying a statement asking them to try again, or outputs a thank you statement for the selection. The main menu for the vending machine then reappears after the juice has been paid for and asks the user again if he/she would like to make a selection, repeating the process until the user exits the program.

Although the system doesn't output the number of items available for each juice, the program tracks the number of juice cans, and if it runs out of one it displays a message that the item as sold out. The user defines the cashRegisterType and the dispenserType classes. The cashRegisterType acts as the user interface for the program while the dispenserType banks the information internally as the user doesn't need to know this information. See Code 8.

```
    #include <iostream>
    #include <iomanip>

    using namespace std;

    class CashRegisterType
    {
       private:
          int cashOnHand;
       public:
          CashRegisterType(int cashIn = 500); //constructor
          int getCurrentBalance() const;
          void acceptAmount(int amountIn);
    };

    CashRegisterType::CashRegisterType(int cashIn)
    : cashOnHand(cashIn)
    {

    }

    int CashRegisterType::getCurrentBalance() const
    {
       return cashOnHand;
    }

    void CashRegisterType::acceptAmount(int cashIn)
    {
       cashOnHand += cashIn;
    }

    class DispenserType
    {
       private:
          int numberOfItems;
          int cost;
       public:
          DispenserType(int noOfItems = 50, int costIn = 50);
          int getCost() const;
          void makeSale();
    };

    DispenserType::DispenserType(int noOfItems, int costIn)
    : numberOfItems(noOfItems), cost(costIn)
    {

    }

    int DispenserType::getCost() const
    {
       return cost;
    }

    void DispenserType::makeSale()
    {
       numberOfItems--;
    }

    int main()
    {

       CashRegisterType reg;
       DispenserType orange;
       DispenserType apple(50, 75);
       DispenserType mango(50, 65);
       DispenserType banana(50, 25);
       DispenserType juices(50, 125);
       int option;

       cout << "Cash in Register: $" << reg.getCurrentBalance() / 100.0 <<
endl;
       //show menu, sell items
       do
       {
          cout << "Select an Item";
          cout << "1. Orange " << orange.getCost() << " Cents." << endl;
          cout << "2. Apple " << apple.getCost() << " Cents." << endl;
          cout << "3. Mango " << mango.getCost() << " Cents." << endl;
          cout << "4. Banana " << banana.getCost() << " Cents." << endl;
          cout << "5. Juice " << juices.getCost() << " Cents." << endl;
          cout << "6. Quit." << endl;
          cout << "Enter Option: ";
          cin >> option;

          if ( option == 1 )
          {
             cout << "You bought Orange for " << orange.getCost() << "
Cents." << endl;
             reg.acceptAmount(orange.getCost());
          }
          else if ( option == 2 )
          {
             cout << "You bought Apple for " << apple.getCost() << " Cents."
<< endl;
             reg.acceptAmount(apple.getCost());
          }
          if ( option == 3 )
          {
             cout << "You bought Mango for " << mango.getCost() << "
Cents." << endl;
             reg.acceptAmount(mango.getCost());
          }
          if ( option == 4 )
          {
             cout << "You bought Banana for " << banana.getCost() << "
Cents." << endl;
             reg.acceptAmount(banana.getCost());
```

```
            }
            if ( option == 5 )
            {
               cout << "You bought Juice for " << juices.getCost() << " Cents."
<< endl;
               reg.acceptAmount(juices.getCost());
            }
         }while(option != 6);

         cout << endl;
         cout << "Cash in Register: $" << reg.getCurrentBalance() / 100.0 <<
endl;

         cin.ignore(100, '\n');
         system("pause");
         return 0;
      }
```

Code 8. Mutiple Classes

IV. Transformations, Assessment, Evaluation and Feedback

With the help of Integrated Development Environment (IDE) software, students learned how to edit-compile-run programs in their first class hour. Also, we observed that the number of assignments completed by the students both during the course and in the class hour is much more compared to a typical lecture-based programming class. The IDE helped students as designers to quickly compare alternative solutions for problems and test for correctness. Student learning was evaluated using many different methods. The background knowledge and preconception checks were performed in the form of a simple questionnaire/worksheet that the students will fill in prior to working on the project assignments. Each student explained the sample programs given in each chapter of the book to others, compiled and executed to check the correctness. The students were asked to explain the concepts they have learned and the programs so that the instructor can measure student learning. Students completed one programming assignment daily and submitted one project report per week. In larger projects, students worked together in groups. Each member turned in an evaluation of his/her own learning experiences gained by being part of a team. To reinforce the learning, a test was scheduled after the completion of each module. Excellent students performed well in all levels and had complete understanding of the subject matter. Very good students were strong in many areas but weak in some. Average students showed weaknesses in some levels. Poor students could not perform well in many areas. Classroom opinion polls and course-related self-confidence surveys were also performed to receive the feedback.

Student progress towards attaining each Student Outcome described in section 1 of this paper is measured in stages. For each stage, students complete a pre-defined set of activities in the course. Data collection processes to assess and evaluate each student outcome include exam questions, group activities, project presentations, focus groups, industrial advisory committee meetings, and other processes that are relevant and appropriate to the program. Student performance is evaluated in the course by the assessment data collection, data analysis, review and recommendations, and course outcomes. Student and Exit surveys are also used to collect data. Data collection processes for each outcome include quizzes, tests, final exams, homework, lab reports, oral presentations, and project work. The undergraduate committee and the program Chair analyze the data. The program faculty makes recommendations thereafter. The undergraduate committee approves the recommendations. The program faculty and the Chair implement those recommendations. The faculty retreat and department meeting times are used to discuss these changes.

Classroom activities were designed to reflect student-centered design and analysis processes. In general, pre project assignments helped the students explore and create on their own. They synthesized the classroom instructions with other resources to produce algorithms and then to test and to debug. In the classroom, each student provided with a computer to extend the concepts they learned in the pre project assignment. Less challenging design problems that can be solved within a given period of time were assigned as in-class closed-project assignments. A post-project assignment helped the students to analyze the use of in-class activities. More challenging and time consuming problems were assigned as post laboratories. After completing each project, students submitted a report discussing their experience. First, each student worked alone as the sole developer of the programs. Towards the end of the semester two to four students were allowed to work in a team to design, construct and complete the projects. The group was responsible for developing definitions and specification of a larger software product to solve a problem as their final project.

V. Summary

This paper described the project-based CS/IS-1 course we have developed to substitute the traditional lecture based classroom to help students design and use at least two user defined data types in a program. First we have outlined the expected student outcomes, thereafter the transformations made, data collected for evaluation and continuous improvements. For many years, we have been experimenting with methods to improve the quality of teaching programming languages for undergraduate computing students. Our goal has been and continues to be to help them become good programmers in a relatively short period of time with both theoretical understanding and practical skills so that they can enter and make an effective contribution to the profession. Traditionally, the programming lectures have been presented to a less than enthusiastic student body in a relatively passive classroom environment. In general, this lecture based instructional process consists of multiple copying stages: the instructor first copies programs from a textbook to his note book, then the instructor displays the projects onto a whiteboard, thereafter the students copy programs into their note books and then edit on a computer to compile and run after the class. Moreover, each instructor allocates considerable amount of his/her time to prepare or update the same course material in each offering. Designing a project based course with learning-by-doing modules and making it available for all the instructors on-line. It reduces the course preparation time for instructors, reduces multiple copying steps in the learning process, strengthen the abilities and increase the enthusiasm of both traditional undergraduate students as well as the adult learners. Growth of any

undergraduate computing program will largely depend on the strength of the introductory programming language courses in the curriculum. Learning takes place through creation, interaction, inspiration and effort. Thus the performance of the students can be improved by converting the traditional passive classroom into an active hands-on learning environment.

We have provided the students an efficient, rigorous and engaging learning environment with necessary tools and training to become proficient in the C++ programming language in a relatively short period of time. Also, we have enhanced the quality of the graduates by helping them to understand basic programming constructs with hands-on skills, integration and team-work. Our approaches taught the students on how to explore a programming language on their own and learn while interacting with a machine loaded with necessary tools. Our active learning environment strengthened student abilities and increase the enthusiasm of both traditional and adult learners. Our active learning modules consist of a complete programming solution to problems, partially completed programs for a given problem and tasks with no programs. We introduced one programming concept of the language at a time. Classroom activities include discussions, reading, writing, modifying and presenting programming solutions to the class. Students enjoyed program design, implementation and error correction in an active classroom.

Within our geographic area we will have opportunities to test our designs which could possibly extend to other colleges, faculty and students. We would like to compare the student retention in a project based active learning programming classes with lecture-based classes. We will continue assessing, evaluating and improving the course material continuously through faculty and student feedback for the next couple semesters. Also, we will continue to share the experience gained from this experiment with the rest of the C++ programming community.

REFERENCES

[1] Y.D. Liang, Introduction to Programming with C++, 2nd Edition, Prentice Hall, 2011

[2] D.S. Malik., C++ Programming From Problem Analysis to Program Design, 5th Edition, Cengage Learning, 2012

[3] T.Gaddis,J.Walters and G.Muganda Starting out with C++, 7th Edition, Addison Wesley,2010

[4] P.Dietel and P Dietel, C++ How to Program, 7th Edition, Addison Wesley, 2010

[5] Horstmannn, C++ For Everyone, 2nd Edition, , Wiley Publishing, ©2011

[6] http://www.codeblocks.org/downloads/26/

AUTHORS PROFILES

The first author of this paper, Dr. Suvineetha Hearth, is a faculty member of University Bahrain. She taught Information Systems, Business and Computer Science departments of Dubuque University, Richard Stockton College, and Atlantic Cape Community College in New Jersey, USA. She also worked as a teaching and research assistant at the Gifu University, and a visiting instructor at Gifu Technical College, in Japan. She received her PhD in Public Policies from the Gifu University, Japan in 1998 as a Rotary Yoneyama graduate Scholar. Also, she continued the coursework required for a second PhD in International Law at Asahi University in Japan. As an Attorney-at-Law, she serve for the Herath Foundation as the Vice President of Legal Affairs and Reforms. Her research interests include security policies, protocols for security, access control, authentication, integrity, Biometrics, Information Assurance and e-learning. She is a Life member of Sri Lanka Bar Association and senior member of the Institute of Electrical and Electronic Engineers (IEEE W) and a active member of the Association of Computer Machinery (ACM). Currently, she is leading the ACM GULF Region Teaching and Learning group and organizing IEEE Computer Society in Bahrain.

Dr. Ajantha Hearth earned his PhD from the Gifu University, Japan, in 1997. His research interests include e-commerce protocols; secure network protocols, computer forensics and algorithm transformations to cryptographic hardware. He worked as the Professor at the University of Fiji's Department of Computer Science and Information Technology in 2011. At present he is teaching at the University of Bahrain. In 1988 he received the prestigious Japanese Government Monbusho research scholarship award. In 2007 he received the Outstanding Research Award for Commitment to Excellence in Computer Forensics and Development of Student Leaders and Researchers from the IEEE–Region 2 AIAA USA. He is a senior member of the IEEE and member of ACM. In 1986, Herath brothers established the Herath Foundation to help financially needy but talented students and awarded more than 7000 scholarships to continue their higher education. He functions as the senior vice president of the Herath Foundation.

Mr. Mohammed A.R. Siddiqui is a faculty of University Bahrain. He received his MSc and B.Sc. from Osmania University, India. In 1999,,he joined UOB and also work as an adjunct faculty at Polytechnic College in Bahrain. His research interests include Visual Programming, Knowledge Base System, Database Management Systems E - Commerce and E-Learning. He is an a member of the Association of Computer Machinery (ACM.) since 2010 and an active member of ACM Gulf region Learning and Research group.

Mr. Khuzaima A.H El-Jallad is a faculty member of University Bahrain 2003. He received his MSc from Towson State University 1984, USA and B.Sc. 1982 from Point Park University, Pittsburgh,. USA. Over the past fifteen years he was working in many projects related to development of Instructional Technology and ICTs though the uses of Digital media. He is a member of the Association of Computer Machinery (ACM.) and an active member of ACM Gulf Region learning and Research group. His research interests are Programming Languages, Elearning, Quality Assurance, Multimedia, Ecommerce and Human Computer Interface.