# Chapter 1
# Introduction to Data Structures

## Programming Exercises

32. Class gradeRecord is in the header file "grade.h" of the IG "include" directory.

```
class gradeRecord
{
   public:
      // constructor initializes the attributes
      gradeRecord(string id = "", int gunits = 0, int gpts = 0);

      // compute and return the student gpa
      double gpa();

      // output student id and grade information
      void writeGradeInfo();

      // update record to account for new course work
      void updateGradeInfo(int newunits, int newgpts);

   private:
      string studentID;    // student identification number
         int units;           // total units earned
         int gradepts;        // total grade points accumulated
};

// constructor. initialize studentID, units, gradepts.
// compute initial gpa
gradeRecord::gradeRecord(string ID, int gunits, int gpts):
   studentID(ID), units(gunits), gradepts(gpts)
{}

double gradeRecord::gpa()
{     if (units == 0)
      return 0.0;
   else
      return float(gradepts)/float(units);
}

void gradeRecord::writeGradeInfo()
{
   cout << "Student:  " << studentID << "  Units:  "
        << units << "  GradePts:  " << gradepts << "  GPA:  "
        << gpa() << endl;
}
```

```
// update record to account for new course work
void gradeRecord::updateGradeInfo(int newunits, int newgpts)
{
   // add in the new grade points and units
   units += newunits;
   gradepts += newgpts;
}
```

(a) The program PEX1_32A.CPP is in the IG software distribution.

```
#include <iostream>

#include "grade.h"

using namespace std;

int main()
{
   // current record for tMartin
   gradeRecord tMartin("716-29-4238", 20, 50);

   // output initial GPA
   cout << "Initial GPA = " << tMartin.gpa() << endl;

   // add 15 units and 40 grade points
   tMartin.updateGradeInfo(15, 40);

   // output the new record
   cout << "Update grade information:" << endl;
   tMartin.writeGradeInfo();

   return 0;
}
```

```
Run:

Initial GPA = 2.5
Update grade information:
Student:  716-29-4238  Units:  35  GradePts:  90  GPA:  2.57143
```

(b) The program PEX1_32B.CPP is in the IG software distribution.

```
#include <iostream>

#include "grade.h"

using namespace std;

int main()
{
   // current records for aDunn and bLange
   gradeRecord aDunn("455-27-9138", 14, 49),
               bLange("657-58-0331", 25, 50),
               saveRecord;
```

```
    int gradeptsPerUnit, i;


    // output GPA for both students
    cout << "aDunn GPA = " << aDunn.gpa() << endl;
    cout << "bLange GPA = " << bLange.gpa() << endl << endl;

    // add 15 units and 40 grade points
    aDunn.updateGradeInfo(4, 16);
    // output the new record for aDunn
    cout << "Update grade information for aDunn:" << endl;
    aDunn.writeGradeInfo();
    cout << endl;

    // initially, assume bLange averages C
    gradeptsPerUnit = 2;
    // loop 3 times
    for (i = 2; i >= 0;i--)
    {
       // save the record for bLange
       saveRecord = bLange;
       // update record assuming 15 units with 15*gradeptsPerUnit
       // grade points
       bLange.updateGradeInfo(15, 15*gradeptsPerUnit);
       // the average grade is char('A' + i), since
       //    'A'+2 = 'C', 'A'+1='B', and 'A'+0='A'
       cout << "bLang's GPA if average grade is " << char('A'+i)
            << ": " << bLange.gpa() << endl;
       // restore the original record
       bLange = saveRecord;
       // move to next higher grade
       gradeptsPerUnit++;
    }

    return 0;
}
```

```
Run:

aDunn GPA = 3.5
bLange GPA = 2

Update grade information for aDunn:
Student:  455-27-9138  Units:  18  GradePts:  65  GPA:  3.61111

bLang's GPA if average grade is C: 2
bLang's GPA if average grade is B: 2.375
bLang's GPA if average grade is A: 2.75
```

33. The modifed gradeRecord class is in the header file "xgrade.h" of the IG "include" directory. The following is the implementation for replaceGrade().

```
void gradeRecord::replaceGrade(int oldUnits, int oldGradePts,
                               int newUnits, int newGradePts)
{
   // remove old units and grade points
   units -= oldUnits;
   gradepts -= oldGradePts;

   // add the new units and grade points
   units += newUnits;
   gradepts += newGradePts;
}
```

The program PEX1_33.CPP is in the IG software distribution.

```
#include <iostream>

#include "xgrade.h"

using namespace std;

int main()
{
   // current record for dHarris
   gradeRecord dHarris("558-29-4424", 100, 180);


   // output GPA for the student
   cout << "dHarris GPA = " << dHarris.gpa() << endl;

   // replace 4 units of D by 4 units of A
   dHarris.replaceGrade(4, 4, 4, 16);
   // output the new GPA for dHarris
   cout << "After replacing 4 units of D by 4 units of A:"
        << endl << "GPA = " << dHarris.gpa() << endl;

   // replace 8 units of F (0 grade points) by 8 units
   // of B (8*3 = 24 grade points)
   dHarris.replaceGrade(8, 0, 8, 24);
   cout << "dHarris substitutes a B for an F in two 4-unit classes:"
        << endl << "GPA = " << dHarris.gpa() << endl;

   return 0;
}
```

```
Run:

dHarris GPA = 1.8
After replacing 4 units of D by 4 units of A:
GPA = 1.92
dHarris substitutes a B for an F in two 4-unit classes:
GPA = 2.16
```

34. Class accumulator is in the header file "accum.h" of the IG "include" directory.

```
class accumulator
{
   public:
      accumulator(double value = 0.0);    // initialize total
      double getTotal();                  // return total
      void addValue(double value = 1.0);  // add to total

   private:
      // total accumulated by the object
      double total;
};

// initialize total
accumulator::accumulator(double value): total(value)
{}

// return the current total
double accumulator::getTotal()
{
   return total;
}

// add value to total
void accumulator::addValue(double value)
{
   total += value;
}
```

(a) The program PEX1_34A.CPP is in the IG software distribution.

```
#include <iostream>

#include "accum.h"

using namespace std;

int main()
{
   accumulator obj;

   cout << obj.getTotal() << endl;
   obj.addValue(3);
   obj.addValue(obj.getTotal()+3);
   cout << obj.getTotal() << endl;

   // using the constructor, create an object and
   // assign it to obj
   obj = accumulator(8);
   obj.addValue();
   cout << obj.getTotal() << endl;

   return 0;
}
```

```
Run:

0
9
9
```

(b) The program PEX1_34B.CPP is in the IG software distribution.

```
#include <iostream>

#include "accum.h"

using namespace std;

int main()
{
   // accumulate negative and positive totals
   accumulator negTotal, posTotal;
   // data comes from arr
   double arr[] = {4, -8, 6, 9, -2, 10, -14, 5};
   int arrSize = sizeof(arr)/sizeof(double);
   int i;

   for (i=0;i < arrSize;i++)
      if(arr[i] < 0)
         // accumulate the negative value
         negTotal.addValue(arr[i]);
      else
         // accumulate the positive value
         posTotal.addValue(arr[i]);

   // output the results
   cout << "Total of the negative numbers = "
        << negTotal.getTotal() << endl;
   cout << "Total of the positive numbers = "
        << posTotal.getTotal() << endl;

   return 0;
}
```

```
Run:

Total of the negative numbers = -24
Total of the positive numbers = 34
```

35. The program PEX1_35.CPP is in the IG software distribution.

```
#include <iostream>

#include "circle.h"
#include "d_rect.h"

using namespace std;
```

```
int main()
{
   double r;

   cout << "Enter the radius of the circle: ";
   cin >> r;

   // declare a circle of radius r
   circle circ(r);
   // rect circumscribes the circle
   rectangle rect(2*r, 2*r);

   cout << "The area between the circle and the circumscribed "
           "square is " << (rect.area() - circ.area()) << endl;

   return 0;
}
```

```
Run:

Enter the radius of the circle: 5
The area between the circle and the circumscribed square is 21.4602
```

36. (a) The modified rectangle is in the header file "xrect.h" of the IG "include" directory. The following is the implementation for diagonal() using inline code.

```
// return the diagonal of the rectangle
double diagonal()
{
   return sqrt(length*length + width*width);
}
```

(b) The program PEX1_36B.CPP is in the IG software distribution.

```
#include <iostream>

#include "xrect.h"

using namespace std;

int main()
{
   double s;

   cout << "Enter the length of a side for the clear glass: ";
   cin >> s;

   // clear glass is s x s square
   rectangle innerSq(s,s);
   // compute the diagonal of the square, which is the length
   // of the frame sides
   double lengthFrame = innerSq.diagonal();
   // declare the frame
   rectangle frame(lengthFrame,lengthFrame);
   double clearGlassCost, stainedGlassCost;
```

```
   // clear glass cost = area of inner square * $.40
   clearGlassCost = innerSq.area() * .40;
   // clear glass cost = area outside inner square and inside frame
   // * $.75
   stainedGlassCost = (frame.area() - innerSq.area()) * .75;
   cout << "The cost of the window is $"
        << clearGlassCost + stainedGlassCost << endl;

   return 0;
}
```

```
Run:

Enter the length of a side for the clear glass: 5
The cost of the window is $28.75
```

(c) The program PEX1_36C.CPP is in the IG software distribution.

```
#include <iostream>

#include "xrect.h"
#include "circle.h"

using namespace std;

int main()
{  double r;

   cout << "Enter the radius of the inner circle: ";
   cin >> r;

   // declare the inner circle
   circle innerCircle(r);
   // inner square is 2r x 2r
   rectangle innerSquare(2*r, 2*r);
   // radius of outer circle is 1/2 diagonal of inner square
   circle outerCircle(innerSquare.diagonal()/2);

   // output the required results
   cout << "Perimeter outer circle/perimeter inner square = "
        << outerCircle.circumference()/innerSquare.perimeter()
        << endl << endl;

   cout << "Area inner square/area inner circle = "
        << innerSquare.area()/innerCircle.area()
        << endl;

   return 0;
}
```

```
Run 1:

Enter the radius of the inner circle: 2

Perimeter outer circle/perimeter inner square = 1.11072
Area inner square/area inner circle = 1.27324
```

```
Run 2:

Enter the radius of the inner circle: 100

Perimeter outer circle/perimeter inner square = 1.11072
Area inner square/area inner circle = 1.27324
```

Some simple geometry gives the following results:

$$\frac{\text{perimeter outer circle}}{\text{perimeter inner squre}} = \frac{\pi\sqrt{2}}{4} = 1.1107207\ldots \qquad \frac{\text{area inner square}}{\text{area inner circle}} = \frac{4}{\pi} = 1.2732395\ldots$$

37. The program PEX1_37.CPP is in the IG software distribution.

```
#include <iostream>

#include "d_random.h"

using namespace std;

int main()
{
   // 10-integer array initialized to 0
   int count[10] = {0};
   randomNumber rnd;
   int i, value;

   // test to see if unit's digit of random numbers produced
   // by randomNumber are uniformly distributed
   for (i=0;i < 1000000;i++)
   {
      // generate a random integer in range 0 to 99,999
      value = rnd.random(100000);
      // extract unit's digit and increment corresponding
      // element of count
      count[value % 10]++;
   }

   // output the results. they should all be near 0.1
   for (i=0;i < 10;i++)
      cout << i << ": " << count[i]/1000000.0 << endl;

   return 0;
}
```

```
Run:

0: 0.100508
1: 0.100332
2: 0.100481
3: 0.099408
4: 0.099743
5: 0.099868
6: 0.099627
7: 0.099932
8: 0.100036
9: 0.100065
```

38. The program PEX1_38.CPP is in the IG software distribution.

```
#include <iostream>

#include "d_random.h"

using namespace std;

// toss n coins and return the number of heads. use
// the random number generator rnd for the simulation
int countHeads(int n, randomNumber& rnd);

const int NUMBERTOSSES = 5000000;

// toss n coins NUMBERTOSSES times and determine the
// empirical probability of tossing m heads
double empiricalHeadToss (int m, int n);

int main()
{
   int n, m;

   cout << "Enter the number of coins: ";
   cin  >> n;
   cout << "Enter the number of heads: ";
   cin >> m;

   cout << "The empirical probability is "
        << empiricalHeadToss(m, n) << endl;

   return 0;
}

/* see Written Exercise 1-20(a) for implementation of countHeads() */

/* see Written Exercise 1-20(b) for implementation of
   empiricalHeadToss() */
```

```
Run 1:

Enter the number of coins: 5
Enter the number of heads: 3
The empirical probability is 0.312767

Run 2:

Enter the number of coins: 5
Enter the number of heads: 1
The empirical probability is 0.156282
```

39. Class playHiLow is in the header file "hilo.h" of the IG "include" directory.

    The program PEX1_39.CPP is in the IG software distribution.

```
#include <iostream>

#include "hilo.h"

using namespace std;

int main()
{
   // guess holds player input
   int guess, result;
   // number of tries. used in loop control
   int numTries;
   // specifies whether player wins or loses. used in
   // loop control
   bool success = false;
   // object used to play the game
   playHiLow player;

   // a player always has at least one try
   numTries = 1;
   // continue until player wins or has made 10 tries
   while (success == false && numTries <= 10)
   {
      // input player's number
      cout << "Guess: ";
      cin >> guess;

      // indicate whether the guess is higher or lower
      // than the target. if the player guesses the
      // number, set success to true and cause loop
      // termination
      result = player.makeGuess(guess);
      if (result < 0)
         cout << "LOWER" << endl;
      else if (result > 0)
         cout << "HIGHER" << endl;
      else
         success = true;
```

```
      // determine number of the next guess
      numTries++;
   }

   // output result of the game
   if (success)
      cout << "You win!" << endl;
   else
   {
      cout << "Sorry. ";
      player.writeTarget();
      cout << endl;
   }

   return 0;
}
```

```
Run:

Guess: 500
HIGHER
Guess: 750
LOWER
Guess: 625
LOWER
Guess: 565
LOWER
Guess: 530
LOWER
Guess: 515
HIGHER
Guess: 522
HIGHER
Guess: 526
HIGHER
Guess: 528
You win!
```

40. The program PEX1_40.CPP is in the IG software distribution.

```
#include <iostream>
#include <string>

using namespace std;

// return the string strA + ch + strB
string linkNames(const string& strA, const string& strB, char ch);

int main()
{
   string algs = "Algorithms ",
          dataStruc = " Data Structures ",
          programs = " Programs",
          title;
```

```
   cout << linkNames(linkNames(algs, dataStruc, '+'), programs, '=')
        << endl;

   return 0;
}

string linkNames(const string& strA, const string& strB, char ch)
{ return strA + ch + strB; }
```

```
Run:

Algorithms + Data Structures = Programs
```

41. The program PEX1_41.CPP is in the IG software distribution.

```
#include <iostream>
#include <string>

using namespace std;

// return the number vowels in the string
int numberVowels(const string& s);

int main()
{  string word[] = {"Arizona", "walk", "government", "C++", "beach"};
   int wordSize = sizeof(word)/sizeof(string), i;

   for (i=0;i < wordSize;i++)
      cout << word[i] << ": " << numberVowels(word[i])
           << " vowels" << endl;

   return 0;
}

int numberVowels(const string& s)
{  int i, nvowels = 0;
   // put all the vowels in a string
   string vowels = "aAeEiIoOuU";

   // cycle through the characters of s
   for (i=0;i < s.length();i++)
      // see if s[i] is in vowels. if so, increment nvowels
      if (vowels.find_first_of(s[i],0) != -1)
         nvowels++;

   return nvowels;
}
```

```
Run:

Arizona: 4 vowels
walk: 1 vowels
government: 3 vowels
C++: 0 vowels
beach: 2 vowels
```

42. The program PEX1_42.CPP is in the IG software distribution.

```
#include <iostream>
#include <string>

using namespace std;

// if strA < strB return strA + strB; otherwise, return
// strB + strA
string newString(const string& strA, const string& strB);

int main()
{  string A = "String", B = "cat";

   cout << newString(A,B) << endl;
   cout << newString(A,"C++") << endl;

   return 0;
}

string newString(const string& strA, const string& strB)
{  if (strA < strB)
      return strA + strB;
   else
      return strB + strA;
}
```

```
Run:

Stringcat
C++String
```

43. The program PEX1_43.CPP is in the IG software distribution.

```
#include <iostream>
#include <string>

using namespace std;

// insert strB into strA at location pos
void strInsert(string& strA, const string& strB, int pos);

// find all occurrences of strB in strA and replace each one by strC
void strReplace (string& strA, const string& strB, const string& strC);

int main()
{  string line, rep;

   cout << "Enter a line of text: ";
   getline(cin, line);
   cout << "Enter a replacement pattern: ";
   cin >> rep;

   // replace all occurrences of "<pat>" in line by rep
   strReplace(line, "<pat>", rep);
   strInsert(line, "+++", line.length()/2);
```

```
   cout << "Modified string:" << endl << line << endl;

   return 0;
}

/* implementations for strInsert() and strReplace() given in
   the solution to Written Exercise 1-27 */
```

```
Run:

Enter a line of text: Alfred the snake h<pat>ed because he m<pat>ed his <pat>y.

Enter a replacement pattern: iss
Modified string:
Alfred the snake hissed be+++cause he missed his Missy.
```

44. The program PEX1_44.CPP is in the IG software distribution.

```
#include <iostream>
#include <string>

using namespace std;

int main()
{  string word;
   char c;

   // input from cin until end-of-file
   while (true)
   {  // input a word
      cin >> word;

      // if EOF, break
      if (!cin)
         break;

      // get word's first character
      c = word[0];
      // see if the 1st character is a vowel
      if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')
         // yes. append "ay" to word
         word += "ay";
      else
      {  // 1st character is a consonant. remove 1st character,
         // append it to the word, and then append "ay"
         word.erase(0,1);
         word += c;
         word += "ay";
      }
      cout << "Pig Latin: " << word << endl;
   }

   return 0;
}
```

```
Run:

this is simple
Pig Latin: histay
Pig Latin: isay
Pig Latin: implesay
```

45. The program PEX1_45.CPP is in the IG software distribution.

```
#include <iostream>

#include "d_time24.h"

using namespace std;

// return true if t1 is later than t2; otherwise,
// return false
bool isLater(const time24& t1, const time24& t2);

int main()
{  // clock is 9:00 AM
   time24 clock(9,0), fivePM(17,0);
   int minutes;

   while (!isLater(clock,fivePM))
   {  // output the current time
      cout << "Time: ";
      clock.writeTime();
      cout << endl;

      // enter a time in minutes and increment clock
      cout << "Enter an increment in minutes: ";
      cin >> minutes;
      clock.addTime(minutes);
   }
   cout << "Final time: ";
   clock.writeTime();
   cout << endl;

   return 0;
}

bool isLater(const time24& t1, const time24& t2)
{  // compare the times in minutes
   return (t1.getHour()*60 + t1.getMinute()) >
          (t2.getHour()*60 + t2.getMinute());
}
```

```
Run:

Time:  9:00
Enter an increment in minutes: 300
Time: 14:00
Enter an increment in minutes: 150
Time: 16:30
```

```
Enter an increment in minutes: 25
Time: 16:55
Enter an increment in minutes: 3
Time: 16:58
Enter an increment in minutes: 2
Time: 17:00
Enter an increment in minutes: 5
Final time: 17:05
```