Problem 1 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int i = 2;
   int x[5] = {3, 1, 4, 1, 5};
   cout << max(2.1, i, i) << endl;                // (a) prints 2.1
   cout << min(x[2], x[3]) << endl;               // (b) prints 1
   doubleIt(i); cout << i << endl;                // (c) prints 4
   printIt(x, 3);                                 // (d) prints 314
   cout << sum(sum(2,6), sum(x[0],x[1])) << endl; // (e) prints 12
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
double max(double x, int y, int z)
```

(b) Title line for **min**.

Answer:

```
int min(int x, int y)
```

(c) Title line for **doubleIt**.

Answer:

```
void doubleIt(int &x)
```

(d) Title line for **printIt**.

Answer:

```
void printIt(int x[], int n)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 2 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int i = 3;
   int x[5] = {2, 7, 1, 8, 2};
   cout << min(i, 2.1, i) << endl;                // (a) prints 2.1
   cout << max(x[2], 3) << endl;                  // (b) prints 3
   cout << doubleIt(i) << endl;                   // (c) prints the following:  2 x 3
   cout << sum(sum(2,6,i), i, i) << endl;         // (d) prints 17
   sortIt(x, 3);                                  // (e) sorts array x by selection sort
   return 0;
}
```

(a) Title line for **min**.

Answer:

```
double min(int x, double y, int z)
```

(b) Title line for **max**.

Answer:

```
int max(int x, int y)
```

(c) Title line for **doubleIt**.

Answer:

```
string doubleIt(int x)
```

(d) Title line for **sum**.

Answer:

```
int sum(int x, int y, int z)
```

(e) Title line for **sortIt**.

Answer:

```
void sortIt(int x[], int n)
```

Problem 3 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int i = 2;
   double x[5] = {3, 1, 4, 1, 5};
   cout << max(4.1, x[i], i) << endl;               // (a) prints 4.1
   cout << min(x[2], x[3]) << endl;                 // (b) prints 1
   doubleIt(i); cout << i << endl;                  // (c) prints 4
   printIt(x, 3);                                   // (d) prints 314
   cout << sum(sum(2.1,6), sum(x[0],x[1])) << endl; // (e) prints 12.1
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
double max(double x, double y, int z)
```

(b) Title line for **min**.

Answer:

```
double min(double x, double y)
```

(c) Title line for **doubleIt**.

Answer:

```
void doubleIt(int &x)
```

(d) Title line for **printIt**.

Answer:

```
void printIt(double x[], int n)
```

(e) Title line for **sum**.

Answer:

```
double sum(double x, double y)
```

Problem 4 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   double i = 3;
   double x[5] = {2, 7, 1, 8, 2};
   cout << min(i, 2.1, i) << endl;               // (a) prints 2.1
   cout << max(x[2], 3.1) << endl;               // (b) prints 3.1
   cout << doubleIt(i) << endl;                  // (c) prints the following:  2 x 3
   cout << sum(sum(2.1,6,i), i, i) << endl;      // (d) prints 17.1
   sortIt(x, 3);                                 // (e) sorts array x by selection sort
   return 0;
}
```

(a) Title line for **min**.

Answer:

```
double min(double x, double y, double z)
```

(b) Title line for **max**.

Answer:

```
double max(double x, double y)
```

(c) Title line for **doubleIt**.

Answer:

```
string doubleIt(double x)
```

(d) Title line for **sum**.

Answer:

```
double sum(double x, double y, double z)
```

(e) Title line for **sortIt**.

Answer:

```
void sortIt(double x[], int n)
```

Problem 5 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int i = 2;
   int x[5] = {3, 1, 4, 1, 5};
   cout << add(i, i) << endl;               // (a) prints 4
   cout << numOdd(x, 5) << endl;           // (b) prints 4
   doubleIt(x[1]); cout << x[1] << endl;    // (c) prints 2
   cout << diff(diff(3,1), 1) << endl;      // (d) prints 1
   cout << percentage(i, x[2]) << endl;     // (e) prints 50%
   return 0;
}
```

(a) Title line for **add**.

Answer:

```
int add(int y, int z)
```

(b) Title line for **numOdd**.

Answer:

```
int numOdd(int x[], int y)
```

(c) Title line for **doubleIt**.

Answer:

```
void doubleIt(int &x)
```

(d) Title line for **diff**.

Answer:

```
int diff(int x, int y)
```

(e) Title line for **percentage**.

Answer:

```
string percentage(int x, int y)
```

Problem 6 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int i = 2;
   int x[5] = {3, 1, 4, 1, 5};
   cout << average(x, 5) << endl;             // (a) prints 2.8
   cout << max(i, i, 3) << endl;              // (b) prints 3
   cout << doubleIt(x[1]) << endl;            // (c) prints 2
   cout << total(total(3,1,1), 1, 1) << endl; // (d) prints 7
   percentage(i, x[2]);                       // (e) prints 50%
   return 0;
}
```

(a) Title line for **average**.

Answer:

```
double average(int y[], int cap)
```

(b) Title line for **max**.

Answer:

```
int max(int x, int y, int z)
```

(c) Title line for **doubleIt**.

Answer:

```
int doubleIt(int x)
```

(d) Title line for **total**.

Answer:

```
int total(int x, int y, int z)
```

(e) Title line for **percentage**.

Answer:

```
void percentage(int x, int y)
```

Problem 7 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   double i = 2.5;
   int x[5] = {3, 1, 4, 1, 5};
   cout << add(i, i) << endl;                // (a) prints 5.0
   if (oddSum(x, 5)) cout << "true" << endl; // (b) prints true
   doubleIt(i); cout << i << endl;           // (c) prints 5.0
   cout << diff(diff(3.0,i), i) << endl;     // (d) prints -2.0
   cout << percentage(x[1], x[2]) << endl;   // (e) prints 25%
   return 0;
}
```

(a) Title line for **add**.

Answer:

```
double add(double y, double z)
```

(b) Title line for **oddSum**.

Answer:

```
bool oddSum(int x[], int y)
```

(c) Title line for **doubleIt**.

Answer:

```
void doubleIt(double &x)
```

(d) Title line for **diff**.

Answer:

```
double diff(double x, double y)
```

(e) Title line for **percentage**.

Answer:

```
string percentage(int x, int y)
```

Problem 8 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   double i = 2; int n = 2;
   double x[5] = {3, 1, 4, 1, 5};
   cout << average(x, 5) << endl;             // (a) prints 2.8
   cout << max(i, i, 3.0) << endl;            // (b) prints 3.0
   cout << doubleIt(x[1]) << endl;            // (c) prints 2.0
   cout << ratio(ratio(3,1), n) << endl;      // (d) prints 1.5
   percentage(i, x[2]);                       // (e) prints 50.0%
   return 0;
}
```

(a) Title line for **average**.

Answer:

```
double average(double y[], int cap)
```

(b) Title line for **max**.

Answer:

```
double max(double x, double y, double z)
```

(c) Title line for **doubleIt**.

Answer:

```
double doubleIt(double x)
```

(d) Title line for **ratio**.

Answer:

```
double ratio(double x, int y)
```

(e) Title line for **percentage**.

Answer:

```
void percentage(double x, double y)
```

Problem 9 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out abc 123**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"An ", "easy ", "question ", ""};
    for (int i = 0; i <= 2; i++) cout << words[i];  cout << endl;      // line (a)
    for (int i = 0; i <= 2; i++) cout << words[i][i];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                         // line (c)
    cout << ++words[0][0] << endl;                                    // line (d)
    cout << argc << endl;                                             // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
An easy question
```

(b) What is the output at line (b)?

Answer:

```
Aae
```

(c) What is the output at line (c)?

Answer:

```
abc
```

(d) What is the output at line (d)?

Answer:

```
B
```

(e) What is the output at line (e)?

Answer:

```
3
```

Problem 10 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out 123**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"An ", "easy ", "question ", ""};
    for (int i = 2; i >= 0; i--) cout << words[i];  cout << endl;      // line (a)
    for (int i = 2; i >= 0; i--) cout << words[i][i+1];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                          // line (c)
    cout << words[0][0]++ << endl;                                     // line (d)
    cout << argc << endl;                                              // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
question easy An
```

(b) What is the output at line (b)?

Answer:

```
ssn
```

(c) What is the output at line (c)?

Answer:

```
123
```

(d) What is the output at line (d)?

Answer:

```
A
```

(e) What is the output at line (e)?

Answer:

```
2
```

Problem 11 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out xyz 987**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"Not ", "very ", "difficult ", ""};
    for (int i = 0; i <= 2; i++) cout << words[i];  cout << endl;      // line (a)
    for (int i = 0; i <= 2; i++) cout << words[i][i];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                          // line (c)
    cout << ++words[0][0] << endl;                                     // line (d)
    cout << argc << endl;                                              // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
Not very difficult
```

(b) What is the output at line (b)?
Answer:

```
Nef
```

(c) What is the output at line (c)?
Answer:

```
xyz
```

(d) What is the output at line (d)?
Answer:

```
0
```

(e) What is the output at line (e)?
Answer:

```
3
```

Problem 12 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out 007**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"Not ", "very ", "difficult ", ""};
    for (int i = 2; i >= 0; i--) cout << words[i];  cout << endl;      // line (a)
    for (int i = 2; i >= 0; i--) cout << words[i][i+1];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                          // line (c)
    cout << words[0][0]++ << endl;                                     // line (d)
    cout << argc << endl;                                              // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
difficult very Not
```

(b) What is the output at line (b)?
Answer:

```
fro
```

(c) What is the output at line (c)?
Answer:

```
007
```

(d) What is the output at line (d)?

Answer:

```
N
```

(e) What is the output at line (e)?

Answer:

```
2
```

Problem 13 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out a 1**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"CS111 ", "Queens ", "College ", ""};
    for (int i = 1; i <= 3; i++) cout << words[i];  cout << endl;      // line (a)
    for (int i = 0; i <= 2; i++) cout << words[i][i];  cout << endl;   // line (b)
    words[3] = argv[2];
    cout << words[3] << endl;                                         // line (c)
    cout << ++words[0][0] << endl;                                    // line (d)
    cout << argc << endl;                                             // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
Queens College
```

(b) What is the output at line (b)?

Answer:

```
Cul
```

(c) What is the output at line (c)?

Answer:

```
1
```

(d) What is the output at line (d)?

Answer:

```
D
```

(e) What is the output at line (e)?

Answer:

```
3
```

Problem 14 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out CS111**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"Queens ", "College ", "CUNY ", "NY"};
    for (int i = 3; i >= 0; i--) cout << words[i];  cout << endl;      // line (a)
    for (int i = 2; i >= 0; i--) cout << words[i][i+1];  cout << endl; // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                         // line (c)
    cout << words[0][0]++ << endl;                                    // line (d)
    cout << ++argc << endl;                                           // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
NYCUNY College Queens
```

(b) What is the output at line (b)?

Answer:

```
Ylu
```

(c) What is the output at line (c)?

Answer:

```
CS111
```

(d) What is the output at line (d)?

Answer:

```
Q
```

(e) What is the output at line (e)?

Answer:

```
3
```

Problem 15 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out out out**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"CS ", "QC ", "CUNY ", "EDU "};
    for (int i = 0; i <= 2; i++) cout << words[i];  cout << endl;      // line (a)
    for (int i = 0; i <= 2; i++) cout << words[i][i];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                         // line (c)
    cout << ++words[0][0] << endl;                                    // line (d)
    cout << argc++ << endl;                                           // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
CS QC CUNY
```

(b) What is the output at line (b)?
Answer:

```
CCN
```

(c) What is the output at line (c)?
Answer:

```
out
```

(d) What is the output at line (d)?
Answer:

```
D
```

(e) What is the output at line (e)?
Answer:

```
3
```

Problem 16 Consider the following C++ program. It is compiled to **a.out** and executed with the command **./a.out 007**.

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

int main(int argc, char *argv[]) {
    string words[4] = {"Queens ", "College ", "Flushing ", "New York"};
    for (int i = 3; i >= 0; i--) cout << words[i];  cout << endl;      // line (a)
    for (int i = 3; i >= 0; i--) cout << words[i][i+1];  cout << endl;   // line (b)
    words[3] = argv[1];
    cout << words[3] << endl;                                          // line (c)
    cout << words[0][0]++ << endl;                                     // line (d)
    cout << --argc << endl;                                              // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
New YorkFlushing College Queens
```

(b) What is the output at line (b)?
Answer:

```
Yslu
```

(c) What is the output at line (c)?
Answer:

```
007
```

(d) What is the output at line (d)?
Answer:

```
Q
```

(e) What is the output at line (e)?

Answer:

```
1
```

Problem 17 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int a = 2, b = 3, c = 4;
   ifstream f;
   string s = "HELLO";  char t[] = "HELLO";
   f.open("testFile.txt");
 // (a) Tests whether a number is even, here Even!
   if (isEven(c)) cout << "Even!" << endl;
 // (b) Removes first and last chars from a string, here ELL
   cout << removeEnds(s) << endl;
 // (c) Prints first word in the input file
   cout << firstWord(f) << endl;
 // (d) Print last character of a C-string, here O
   cout << lastChar(t) << endl;
 // (e) Rotate a,b,c so as to print 3,4,2
   rotate(a, b, c);
   cout  << a << b << c << endl;
   return 0;
}
```

Answer:

(a)

```
bool isEven(int x) {
    return x % 2 == 0;
}
```

(b)

```
string removeEnds(string x) {
    return x.substr(1, x.length() - 2);
}
```

(c)

```
string firstWord(ifstream &file) {
    string x;
    file >> x;
    return x;
}
```

(d)

```
char lastChar(char x[]) {
   return x[strlen(x) - 1];
}
```

(e)

```
void rotate(int &x, int &y, int &z) {
   int temp = x;
   x = y;
   y = z;
   z = temp;
}
```

Problem 18 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int a = 23, b = 3, c = 4;
   ifstream f;
   string s = "HELLO";  char t[] = "HELLO";
   f.open("testFile.txt");
 // (a) Tests whether a number has 2 digits, here Yes!
   if (is2digit(a)) cout << "Yes!" << endl;
 // (b) Doubles a string, here HELLOHELLO
   cout << doubleIt(s) << endl;
 // (c) The number of words read from the input file before eof() is true
   cout << countWords(f) << endl;
 // (d) Print middle character of a C-string that has a middle, here L
   cout << midChar(t) << endl;
 // (e) Rotate a,b,c so as to print 4,23,3
   rotate(a, b, c);
   cout  << a << "," << b << "," << c << endl;
   return 0;
}
```

Answer:

(a)

```
bool is2digit(int x) {
    return (x > 9) && (x < 100);
}
```

(b)

```
string doubleIt(string x) {
    return x + x;
}
```

(c)

```
int countWords(ifstream &file) {
    string x;
    int count = 0;
    while (!file.eof()) {
       file >> x;
       count++;
    }
    return count;
}
```

(d)

```
char midChar(char x[]) {
   return x[(strlen(x) - 1)/2];
}
```

(e)

```
void rotate(int &x, int &y, int &z) {
   int temp = x;
   x = z;
   z = y;
   y = temp;
}
```

Problem 19 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int a = 2, b = 3, c = 4;
   ifstream f;
   string s = "HELLO";  char t[] = "HELLO";
   f.open("testFile.txt");
 // (a) Tests whether a number is seven, here No!
   if (!isSeven(c)) cout << "No!" << endl;
 // (b) Removes the last char from a string, here HELL
   cout << removeLast(s) << endl;
 // (c) Prints second word in the input file
   cout << secondWord(f) << endl;
 // (d) Print first character of a C-string, here H
   cout << firstChar(t) << endl;
 // (e) swap a with the biggest of a,b,c.  Here prints 4,3,2
   swapBig(a, b, c);
   cout  << a << b << c << endl;
   return 0;
}
```

Answer:

(a)

```
bool isSeven(int x) {
    return x == 7;
}
```

(b)

```
string removeLast(string x) {
    return x.substr(0, x.length() - 1);
}
```

(c)

```
string secondWord(ifstream &file) {
    string x;
```

```
    file >> x;
    file >> x;
    return x;
}
```

(d)

```
char firstChar(char x[]) {
   return x[0];
}
```

(e)

```
void swapBig(int &x, int &y, int &z) {
   int temp = x;
   if (x < y && z <= y) {
      x = y;
      y = temp;
   } else if (x < z) {
      x = z;
      z = temp;
   }
}
```

Problem 20 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int a = 123, b = 3, c = 4;
   ifstream f;
   string s = "HELLO";  char t[] = "HELLO";
   f.open("testFile.txt");
 // (a) Tests whether a number has 3 digits, here Yes!
   if (is3digit(a)) cout << "Yes!" << endl;
 // (b) Returns the part of a string before its midpoint, here HE
   cout << halfIt(s) << endl;
 // (c) The number of characters read from the input file before eof() is true
   cout << countChar(f) << endl;
 // (d) Print third character of a C-string that has a middle, here L
   cout << thirdChar(t) << endl;
 // (e) Replace a, b and c by their sum to print 130, 130, 130
   replace(a, b, c);
   cout  << a << "," << b << "," << c << endl;
   return 0;
}
```

Answer:

(a)

```
bool is3digit(int x) {
    return (x > 99) && (x < 1000);
}
```

(b)

```
string halfIt(string x) {
    return x.substr(0, x.length()/2);
}
```

(c)

```
int countChar(ifstream &file) {
    char x;
    int count = 0;
    while (!file.eof()) {
        x = file.get();
        count++;
    }
    return count;
}
```

(d)

```
char thirdChar(char x[]) {
   return x[2];
}
```

(e)

```
void replace(int &x, int &y, int &z) {
   x = x + y + z;
   y = x;
   z = x;
}
```

Problem 21 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   string s = "HELLO", t = "GOODBYE";
 // (a) Tests whether a string has 5 or more letters
   if (isLong(s)) cout << "Long!" << endl;
 // (b) Tests whether a string contains the letter E
   cout << hasE(s) << endl;
 // (c) Returns a string with just the first 4 characters
   cout << first4(t) << endl;
 // (d) Prints the last character at or before the middle of the string
   cout << middle(t) << endl;
 // (e) swaps them
   swap(s, t);
   cout  << s << " " << t << endl;
   return 0;
}
```

Answer:

(a)

```
bool isLong(string x) {
    return x.length() > 4;
}
```

(b)

```
bool hasE(string x) {
    return x.find("E") >= 0;
}
```

(c)

```
string first4(string x) {
    return x.substr(0,4);
}
```

(d)

```
char middle(string x) {
   return x[x.length()/2];
}
```

(e)

```
void swap(string &x, string &y) {
   string temp = x;
   x = y;
   y = temp;
}
```

Problem 22 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   string s = "HELLO", t = "GOODBYE";
 // (a) return number of characters
   cout << stringLength(s) << endl;
 // (b) Tests whether a string contains a target
   cout << contains(s, "HELL") << endl;
 // (c) Returns a string with just the last 4 characters
   cout << last4(t) << endl;
 // (d) Prints the first character
   cout << first(t) << endl;
 // (e) adds on the second string
   addOn(s, t);
   cout  << s << endl;
   return 0;
}
```

Answer:

(a)

```
int stringLength(string x) {
    x.length();
}
```

(b)

```
bool contains(string x, string target) {
    return x.find(target) >= 0;
}
```

(c)

```
string last4(string x) {
    return x.substr(x.length() - 4, 4);
}
```

(d)

```
char first(string x) {
   return x[0];
}
```

(e)

```
void addOn(string &x, string y) {
   x = x + y;
}
```

Problem 23 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   string s = "HELLO", t = "GOODBYE";
 // (a) Tests whether a string starts in upper case
   if (isUpper(s)) cout << "Upper Case!" << endl;
 // (b) Tests whether a string omits the letter E
   cout << hasNoE(s) << endl;
 // (c) Returns a string that drops the first character
   cout << dropFirst(t) << endl;
 // (d) Prints the last character
   cout << last(t) << endl;
 // (e) If t is shorter than s, swap the strings, otherwise do nothing
   sort(s, t);
   cout  << s << " " << t << endl;
   return 0;
}
```

Answer:

(a)

```
bool isUpper(string x) {
    'A' <= x[0] && x[0] <= 'Z';
}
```

(b)

```
bool hasNoE(string x) {
    return x.find("E") < 0;
}
```

(c)

```
string dropFirst(string x) {
    return x.substr(1);
}
```

(d)

```
char last(string x) {
   return x[x.length() - 1];
}
```

(e)

```
void sort(string &x, string &y) {
   if (x.length() <= y.length()) return;
   string temp = x;
   x = y;
   y = temp;
}
```

Problem 24 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   string s = "HELLO", t = "GOODBYE";
 // (a) Do two strings have the same number of characters?
   cout << sameLength(s, t) << endl;
 // (b) Tests whether a string contains a target
   cout << contains("HELL", s) << endl;
 // (c) Returns a string that drops the last character
   cout << dropLast(t) << endl;
 // (d) Prints the third character
   cout << third(t) << endl;
 // (e) Turns an upper case character to lower case
   lower(s[0]);
   cout  << s << endl;
   return 0;
}
```

Answer:

(a)

```
bool sameLength(string x, string y) {
    return x.length() == y.length();
}
```

(b)

```
bool contains(string target, string x) {
    return x.find(target) >= 0;
}
```

(c)

```
string dropLast(string x) {
    return x.substr(0, x.length() - 1);
}
```

(d)

```
char third(string x) {
   return x[2];
}
```

(e)

```
void lower(char &x) {
   if ('A' <= x && x <= 'Z') x = x + 'a' - 'A';
}
```

Problem 25 Write a function called *subtractAverage* that calculates the average of the entries in a 2-dimensional array (that is known to have 2 columns) and subtracts this average from every entry of the array.

For example, a program that uses the function *subtractAverage* follows.

```
int main() {
   double x[3][2] = {{1,3}, {1,3}, {1,3}} ;  // average is 2 here
   subtractAverage(x, 3, 2);
   cout << x[0][0] << " " << x[0][1] << endl;  // prints:  -1 1
   return 0;
}
```

Answer:

```
void subtractAverage(double x[][2], int rows, int cols) {
   double sum  = 0;
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++) sum += x[r][c];
   double average = sum / (rows * cols);
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++) x[r][c] -= average;

}
```

Problem 26 Write a function called *addMin* that calculates the minimum of the entries in a 2-dimensional array (that is known to have 2 columns) and adds this minimum to every entry of the array.

For example, a program that uses the function *addMin* follows.

```
int main() {
   int x[3][2] = {{1,3}, {1,3}, {1,3}} ;  // min is 1 here
   addMin(x, 3, 2);
   cout << x[0][0] << " " << x[0][1] << endl;  // prints:  2 4
   return 0;
}
```

Answer:

```
void addMin(int x[][2], int rows, int cols) {
   int min  = x[0][0];
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++)
         if (x[r][c] < min) min = x[r][c];
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++)
         x[r][c] += min;
}
```

Problem 27 Write a function called *subtractAverage* that calculates the average of the entries in an array and subtracts this average from every positive entry of the array.

For example, a program that uses the function *subtractAverage* follows.

```
int main() {
   double x[5] = {3, 1, 4, 1, 6} ;               // average is 3 here
   subtractAverage(x, 5);
   cout << x[0] << " " << x[1] << x[2] << endl;  // prints:  0 -2 1
   return 0;
}
```

Answer:

```
void subtractAverage(double x[], int capacity) {
   double sum  = 0;
   for (int r = 0; r < capacity; r++) sum += x[r];
   double average = sum / capacity;
   for (int r = 0; r < capacity; r++)
      if (x[r] > 0) x[r] -= average;

}
```

Problem 28 Write a function called *addMin* that calculates the minimum of the entries in an array and adds this minimum to every odd entry of the array.

For example, a program that uses the function *addMin* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5} ;  // min is 1 here
   addMin(x, 5);
   cout << x[0] << " " << x[1] << " " << x[2] << endl;  // prints:  4 2 4
   return 0;
}
```

Answer:

```
void addMin(int x[], int cols) {
   int min  = x[0];
   for (int c = 0; c < cols; c++)
      if (x[c] < min) min = x[c];
   for (int c = 0; c < cols; c++)
      if (x[c] % 2 == 1)
         x[c] += min;
}
```

Problem 29 Write a function called *minGap* that calculates the smallest gap between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *minGap* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5};
   cout << minGap(x, 5) << endl;   // prints 2 corresponding to the gap from 3 to 1.
   return 0;
}
```

Answer:

```
int minGap(int x[], int cap) {
   int ans = x[1] - x[0];
   if (ans < 0) ans = -ans;
   for (int i = 1; i < cap; i++) {
      if (x[i] > x[i-1]) {
         if ((x[i] - x[i-1]) < ans)
            ans = x[i] - x[i - 1];
      }  else {
         if ((x[i- 1] - x[i]) < ans)
            ans = x[i - 1] - x[i];
      }
   }
   return ans;
}
```

Problem 30 Write a function called *gapSum* that calculates the sum of the gaps between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *gapSum* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5};
   cout << gapSum(x, 5) << endl;   // prints 12
   // The gaps are 2, 3, 3, 4 and these add to 12
   return 0;
}
```

Answer:

```
int gapSum(int x[], int cap) {
   int ans = 0;
   for (int i = 1; i < cap; i++) {
      if (x[i] > x[i-1]) {
            ans = ans + x[i] - x[i - 1];
      }  else {
            ans = ans + x[i - 1] - x[i];
      }
   }
   return ans;
}
```

Problem 31 Write a function called *maxGap* that calculates the biggest gap between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *maxGap* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5};
   cout << maxGap(x, 5) << endl;   // prints 4 corresponding to the gap from 1 to 5.
   return 0;
}
```

Answer:

```
int maxGap(int x[], int cap) {
   int ans = 0;
   for (int i = 1; i < cap; i++) {
      if ((x[i] - x[i-1]) > ans)
         ans = x[i] - x[i - 1];
      if ((x[i- 1] - x[i]) > ans)
         ans = x[i - 1] - x[i];
   }
   return ans;
}
```

Problem 32 Write a function called *gapProd* that calculates the product of the gaps between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *gapProd* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5};
   cout << gapProd(x, 5) << endl;   // prints 72
   // The gaps are 2, 3, 3, 4 and these multiply to 72
   return 0;
}
```

Answer:

```
int gapProd(int x[], int cap) {
   int ans = 1;
   for (int i = 1; i < cap; i++) {
      if (x[i] > x[i-1]) {
            ans = ans * (x[i] - x[i - 1]);
      }  else {
            ans = ans * (x[i - 1] - x[i]);
      }
   }
   return ans;
}
```

Problem 33 Write a function called *roundOff* that returns the result of turning all digits (except the first) in a positive integer parameter to 0.

For example, a program that uses the function *roundOff* follows.

```
int main() {
   cout << roundOff(19683) << endl;  // prints 10000
   cout << roundOff(2) << endl;      // prints 2
   return 0;
}
```

Answer:

```
int roundOff(int x) {
   if (x < 10) return x;
   return 10* roundOff(x/10);
}
```

Problem 34 Write a function called *allFirst* that returns the result of turning all digits in a positive integer parameter to match the first digit.

For example, a program that uses the function *allFirst* follows.

```
int main() {
   cout << allFirst(19683) << endl;  // prints 11111
   cout << allFirst(2048) << endl;   // prints 2222
   return 0;
}
```

Answer:

```
int allFirst(int x) {
   if (x < 10) return x;
   int y = allFirst(x/10);
   return 10*y + y%10;
}
```

Problem 35 Write a function called *firstDown* that returns the result of decreasing the first digit in a positive integer by 1.

For example, a program that uses the function *firstDown* follows.

```
int main() {
   cout << firstDown(2048) << endl;       // prints 1048
   cout << firstDown(19683) << endl;      // prints 9683
   return 0;
}
```

Answer:

```
int firstDown(int x) {
   if (x < 10) return x - 1;
   return 10* firstDown(x/10) + x % 10;
}
```

Problem 36 Write a function called *firstUp* that returns the result of increasing the first digit of the parameter by 1, unless this first digit is 9 in which case it is not changed.

For example, a program that uses the function *firstUp* follows.

```
int main() {
   cout << firstUp(19683) << endl;  // prints 29683
   cout << firstUp(95) << endl;     // prints 95
   return 0;
}
```

Answer:

```
int firstUp(int x) {
   if (x < 9) return x + 1;
   if (x == 9) return x;
   return 10* firstUp(x/10) + x % 10;
}
```

Problem 37 Write a function called *oddOne* that returns the result of turning all odd digits in a positive integer parameter to 1.

For example, a program that uses the function *oddOne* follows.

```
int main() {
   cout << oddOne(19683) << endl;  // prints 11681
   cout << oddOne(2) << endl;      // prints 2
   return 0;
}
```

Answer:

```
int oddOne(int x) {
   if (x == 0) return 0;
   if (x % 2 == 0) return 10 * oddOne(x/10) + x % 10;
   return 10* oddOne(x/10) + 1;
}
```

Problem 38 Write a function called *oddOneOut* that returns the result of removing the rightmost odd digit in a positive integer parameter.

For example, a program that uses the function *oddOneOut* follows.

```
int main() {
   cout << oddOneOut(19682) << endl;  // prints 1682
   cout << oddOneOut(2) << endl;      // prints 2
   return 0;
}
```

Answer:

```
int oddOneOut(int x) {
   if (x == 0) return 0;
   if (x % 2 == 1) return x/10;
   return 10 * oddOneOut(x/10) + x % 10;
}
```

Problem 39 Write a function called *eveNine* that returns the result of turning all even digits in a positive integer parameter to 9.

For example, a program that uses the function *eveNine* follows.

```
int main() {
   cout << eveNine(19683) << endl;  // prints 19993
   cout << eveNine(3) << endl;      // prints 3
   return 0;
}
```

Answer:

```
int eveNine(int x) {
   if (x == 0) return 0;
   if (x % 2 != 0) return 10 * eveNine(x/10) + x % 10;
   return 10* eveNine(x/10) + 9;
}
```

Problem 40 Write a function called *evenOut* that returns the result of removing the rightmost even digit in a positive integer parameter.

For example, a program that uses the function *evenOut* follows.

```
int main() {
   cout << evenOut(19683) << endl;  // prints 1963
   cout << evenOut(2) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int evenOut(int x) {
   if (x == 0) return 0;
   if (x % 2 == 0) return x/10;
   return 10 * evenOut(x/10) + x % 10;
}
```

Problem 41 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads the entries in a 2-dimensional array with 4 rows and 4 columns from the user.
2. It prints (all) rows that have the greatest sum.

Here is an example of how the program should work:

```
Give me the entries of a 4 x 4 array:
0 0 0 -1
1 2 3 4
1 1 1 1
2 3 3 2

Largest rows:
1 2 3 4
2 3 3 2
```

Answer:

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

int main() {
   int x[4][4];
   cout << "Give me the entries of a 4 x 4 array:" << endl;
   for (int i = 0; i < 4; i++)
      for (int j = 0; j < 4; j++) cin >> x[i][j];

   int sums[4] = {0, 0, 0 ,0 };
   for (int i = 0; i < 4; i++)
      for (int j = 0; j < 4; j++) sums[i] += x[i][j];

   int max = sums[0];
   for (int i = 1; i < 4; i++)
      if (sums[i] > max) max = sums[i];

   cout << "Largest rows\n";
   for (int i = 0; i < 4; i++) {
      if (sums[i] == max) {
         for (int j = 0; j < 4; j++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 42 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads the entries in a 2-dimensional array with 5 rows and 3 columns from the user.

2. It prints the last row that has an even sum.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 3 array:
0 0 0
1 2 3
1 1 1
3 3 3
1 1 1

Last row with even sum:
1 2 3
```

Answer:

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

int main() {
   int x[5][3];
   cout << "Give me the entries of a 5 x 3 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 3; j++) cin >> x[i][j];

   int sums[5] = {0, 0, 0, 0, 0};
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 3; j++) sums[i] += x[i][j];

   cout << "Last row with even sum: \n";
   for (int i = 4; i >= 0; i--) {
      if (sums[i] % 2 == 0) {
         for (int j = 0; j < 3; j++) cout << x[i][j] << " ";
         cout << endl;
         return 0;
      }
   }
   return 0;
}
```

Problem 43 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads the entries in a 2-dimensional array with 4 rows and 4 columns from the user.

2. It prints (all) columns that have the greatest sum.

Here is an example of how the program should work:

```
Give me the entries of a 4 x 4 array:
0 0 0 -1
1 2 3 4
1 1 1 1
2 3 3 2

Largest columns:
0 3 1 3
```

Answer:

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

int main() {
   int x[4][4];
   cout << "Give me the entries of a 4 x 4 array:" << endl;
   for (int i = 0; i < 4; i++)
      for (int j = 0; j < 4; j++) cin >> x[i][j];

   int sums[4] = {0, 0, 0 ,0 };
   for (int i = 0; i < 4; i++)
      for (int j = 0; j < 4; j++) sums[j] += x[i][j];

   int max = sums[0];
   for (int i = 1; i < 4; i++)
      if (sums[i] > max) max = sums[i];

   cout << "Largest columns\n";
   for (int j = 0; j < 4; j++) {
      if (sums[j] == max) {
         for (int i = 0; i < 4; i++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 44 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads the entries in a 2-dimensional array with 5 rows and 3 columns from the user.

2. It prints the last column that has an even sum.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 3 array:
0 0 0
1 2 3
1 1 1
3 3 3
1 2 0

Last column with even sum:
0 2 1 3 2
```

Answer:

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

int main() {
   int x[5][3];
   cout << "Give me the entries of a 5 x 3 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 3; j++) cin >> x[i][j];

   int sums[5] = {0, 0, 0, 0, 0};
   for (int i = 0; i < 5; i++)
```

```
      for (int j = 0; j < 3; j++) sums[j] += x[i][j];

   cout << "Last column  with even sum: \n";
   for (int i = 2; i >= 0; i--) {
      if (sums[i] % 2 == 0) {
         for (int j = 0; j < 5; j++) cout << x[j][i] << " ";
         cout << endl;
         return 0;
      }
   }
   return 0;
}
```

Problem 45 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads (from the user) the entries in a 2-dimensional array with 5 rows and 5 columns.

2. It prints (all) rows that have the property that entries increase as we move along their columns.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 5 array:
0 0 0 0 0
1 2 3 4 5
1 5 6 7 99
2 -1 3 4 5
5 4 3 2 1

Increasing rows:
1 2 3 4 5
1 5 6 7 99
```

Answer:

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

int main() {
   int x[5][5];
   cout << "Give me the entries of a 5 x 5 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 5; j++) cin >> x[i][j];

   cout << "Increasing rows\n";
   for (int i = 0; i < 5; i++) {
      bool ok = true;
      for (int j = 1; j < 5; j++)
         if (x[i][j] <= x[i][j - 1]) ok = false;
      if (ok) {
         for (int j = 0; j < 5; j++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 46 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads (from the user) the entries in a 2-dimensional array with 5 rows and 5 columns.

2. It prints (all) columns that have the property that entries increase as we move down their rows.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 5 array:
0 1 5  10  10
0 2 4  11  20
0 3 3   9  21
0 4 2  12  41
0 5 1  13  99

Increasing columns:
1 2 3 4 5
10 20 21 41 99
```

Answer:

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

int main() {
   int x[5][5];
   cout << "Give me the entries of a 5 x 5 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 5; j++) cin >> x[i][j];

   cout << "Increasing columns\n";
   for (int j = 0; j < 5; j++) {
      bool ok = true;
      for (int i = 1; i < 5; i++)
         if (x[i][j] <= x[i -1][j]) ok = false;
      if (ok) {
         for (int i = 0; i < 5; i++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 47 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads (from the user) the entries in a 2-dimensional array with 5 rows and 5 columns.

2. It prints (all) rows that have the property that entries decrease as we move along their columns.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 5 array:
0 0 0 0 0
1 2 3 4 5
501 5 306 107 99
2 -1 -3 -4 -5
5 4 3 2 1

Decreasing rows:
2 -1 -3 -4 -5
5 4 3 2 1
```

Answer:

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

int main() {
```

```
   int x[5][5];
   cout << "Give me the entries of a 5 x 5 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 5; j++) cin >> x[i][j];

   cout << "Decreasing rows\n";
   for (int i = 0; i < 5; i++) {
      bool ok = true;
      for (int j = 1; j < 5; j++)
         if (x[i][j] >= x[i][j - 1]) ok = false;
      if (ok) {
         for (int j = 0; j < 5; j++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 48 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It reads (from the user) the entries in a 2-dimensional array with 5 rows and 5 columns.

2. It prints (all) columns that have the property that entries decrease as we move down their rows.

Here is an example of how the program should work:

```
Give me the entries of a 5 x 5 array:
0 1 5  10  99
0 2 4  11  41
0 3 3   9  21
0 4 2  12  20
0 5 1  13  10

Decreasing columns:
5 4 3 2 1
99 41 21 20 10
```

Answer:

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

int main() {
   int x[5][5];
   cout << "Give me the entries of a 5 x 5 array:" << endl;
   for (int i = 0; i < 5; i++)
      for (int j = 0; j < 5; j++) cin >> x[i][j];

   cout << "Decreasing columns\n";
   for (int j = 0; j < 5; j++) {
      bool ok = true;
      for (int i = 1; i < 5; i++)
         if (x[i][j] >= x[i -1][j]) ok = false;
      if (ok) {
         for (int i = 0; i < 5; i++) cout << x[i][j] << " ";
         cout << endl;
      }
   }
}
```

Problem 49 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(a, 4) << endl;              // (a) prints: 358
   reverse(a, 4);                          // (b) prints: 358 265 315 314
   b = add(b, c);                          // (c) b becomes 5
   cout << difference(a[0], a[1]) << endl; // (d) prints: 1
   a[0] = sum(a[1], c);                    // (e) a[0] becomes 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int x[], int cap)
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 50 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(b, c);                             // (a) swaps values of b & c
   b = last(a, 4);                         // (b) b becomes 358
   c = add(a[1], a[0]);                    // (c) c becomes 629
   cout << max(a[1], 1) << endl;           // (d) prints: 314
   cout << max(a, capacity, 700) << endl;  // (e) prints 700
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **last**.

Answer:

```
int last(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **max**.

Answer:

```
int max(int x, int y)
```

(e) Title line for **max**.

Answer:

```
int max(int x[], int y, int z)
```

Problem 51 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(4, a) << endl;              // (a) prints: 358
   reverse(a, 4);                          // (b) a becomes 358,265,315,314
   b = add(b, b, c);                       // (c) b becomes 6
   cout << difference(a[1], 300) << endl;  // (d) prints: 15
   addOn(a[1], c);                         // (e) a[1] changes to 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int cap, int x[])
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y, int z)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **addOn**.

Answer:

```
void addOn(int &x, int y)
```

Problem 52 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(a[3], c);                          // (a) swaps values of a[3] & c
   b = first(a);                           // (b) b becomes 314
   a[3] = add(a[1], a[0]);                 // (c) a[3] becomes 629
   cout << min(a, capacity) << endl;       // (d) prints: 265
   printMin(a, capacity);                  // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **first**.

Answer:

```
int first(int x[])
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **min**.

Answer:

```
int min(int x[], int y)
```

(e) Title line for **printMin**.

Answer:

```
void printMin(int x[], int y)
```

Problem 53 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(a, 2, 2) << endl;                 // (a) prints: 358
   reverse(a, 2, 2);                             // (b) prints: 358 265 315 314
   b = add(b, c);                                // (c) b becomes 5
   cout << difference(a[0][0], a[0][1]) << endl; // (d) prints: 1
   a[0][0] = sum(a[0][1], c);                    // (e) a[0][0] becomes 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int x[][2], int r, int c)
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 54 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, rows = 2, cols = 2;

   swap(b, c);                                // (a) swaps values of b & c
   b = last(a, rows, cols);                   // (b) b becomes 358
   c = add(a[0][1], a[0][0]);                 // (c) c becomes 629
   cout << max(a[0][1], 1) << endl;           // (d) prints: 315
   cout << max(a, rows, cols, 700) << endl;   // (e) prints 700
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **last**.

Answer:

```
int last(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **max**.

Answer:

```
int max(int x, int y)
```

(e) Title line for **max**.

Answer:

```
int max(int x[][2], int r, int c, int z)
```

Problem 55 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(2, 2, a) << endl;              // (a) prints: 358
   reverse(a, 2, 2);                          // (b) a becomes 358,265,315,314
   b = add(b, b, c);                          // (c) b becomes 6
   cout << difference(a[0][1], 300) << endl;  // (d) prints: 15
   addOn(a[0][1], c);                         // (e) a[0][1] changes to 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int r, int c, int x[][2])
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y, int z)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **addOn**.

Answer:

```
void addOn(int &x, int y)
```

Problem 56 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, row = 2, col = 2;

   swap(a[1][1], c);                          // (a) swaps values of a[1][1] & c
   b = first(a);                              // (b) b becomes 314
   a[1][1] = add(a[0][1], a[0][0]);           // (c) a[1][1] becomes 629
   cout << min(a, row, col) << endl;          // (d) prints: 265
   printMin(a, row, col);                     // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap**.
Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **first**.
Answer:

```
int first(int x[][2])
```

(c) Title line for **add**.
Answer:

```
int add(int x, int y)
```

(d) Title line for **min**.
Answer:

```
int min(int x[][2], int r, int c)
```

(e) Title line for **printMin**.
Answer:

```
void printMin(int x[][2], int r, int c)
```

Problem 57 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the difference (ignoring sign), here 1
   cout << absoluteDifference(7,8) << endl;
 // (b) Prints random integer in range from b to c, assume b < c
   cout << random(b, c) << endl;
 // (c) Print square root of sum of squares of arguments, here 5.0
   cout << hyp(3, 4) << endl;
 // (d) Print array backwards, here 1413
   backPrint(a, 4);
 // (e) Print the last digit, assume argument is positive.  Here 3.
   lastDigit(19683);
   return 0;
}
```

Answer:
(a)

```
int absoluteDifference(int x, int y) {
    if (x < y) return y - x;
    return x - y;
}
```

(b)

```
int random(int x, int y) {
    return rand() % (y - x + 1) + x;
}
```

(c)

```
double hyp(int x, int y) {
    return sqrt(x * x + y * y);
}
```

(d)

```
void backPrint(int x[], int cap) {
   for (int i = cap - 1; i >= 0; i--)
       cout << x[i];
   cout << endl;
}
```

(e)

```
void lastDigit(int x) {
   cout << x % 10;
}
```

Problem 58 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the max, here 8
   cout << max(7,8) << endl;
 // (b) Swaps values
   swap(b, c);
 // (c) Print ratio, here 0.75
   cout << ratio(3, 4) << endl;
 // (d) Print number of even entries, here 1
   cout << countEven(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   firstDigit(19683);
   return 0;
}
```

Answer:

(a)

```
int max(int x, int y) {
    if (x < y) return y;
    return x;
}
```

(b)

```
void swap(int &x, int &y) {
   int temp = x;
   x = y;
   y = temp;
}
```

(c)

```
double ratio(int x, int y) {
    return ((double) x) / y;
}
```

(d)

```
int countEven(int x[], int cap) {
   int ans = 0;
   for (int i = 0; i < cap; i++)
      if (x[i] % 2 == 0) ans++;
   return ans;
}
```

(e)

```
void firstDigit(int x) {
    while (x > 10) x = x / 10;
    cout << x;
}
```

Problem 59 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the absolute value (ignore sign), here 7
   cout << absolute(-7) << endl;
 // (b) Prints a random id number with the given length, here 007 may be printed
   random(3);
 // (c) Prints the ratio as a percentage, here 12.5% for 1/8
   cout << percentage(1, 8) << "%" << endl;
 // (d) Print every second entry of the array here 34
   skipPrint(a, 4);
 // (e) Print the last two digit, assume argument is at least 10.  Here 83.
   lastTwoDigits(19683);
   return 0;
}
```

Answer:

(a)

```
int absolute(int x) {
    if (x < 0) return - x;
    return x;
}
```

(b)

```
void random(int x) {
    for (int i = 0; i < x; i++)
       cout << rand() % 10;
    cout << endl;
}
```

(c)

```
double percentage(int x, int y) {
    return 100.0 * x / y;
}
```

(d)

```
void skipPrint(int x[], int cap) {
   for (int i = 0; i < cap; i += 2)
       cout << x[i];
   cout << endl;
}
```

(e)

```
void lastTwoDigits(int x) {
   cout << x % 100;
}
```

Problem 60 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Print the number of odd arguments, here 1
   cout << numberOdd(7,8) << endl;
 // (b) Reorder arguments so that they increase, here swap them
   sort(c, b);
 // (c) Print closest integer here 4
   cout << closest(3.75) << endl;
 // (d) Print maximum entry, here 4
   cout << max(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   cout << firstDigit(19683) << endl;
   return 0;
}
```

Answer:

(a)

```
int numberOdd(int x, int y) {
    return x % 2 + y % 2;
}
```

(b)

```
void sort(int &x, int &y) {
   if (x <= y) return;
   int temp = x;
   x = y;
   y = temp;
}
```

(c)

```
int closest(double x) {
    return (int) (x + 0.5);
}
```

(d)

```
int max(int x[], int cap) {
   int ans = x[0];
   for (int i = 0; i < cap; i++)
      if (x[i] > ans) ans = x[i];
   return ans;
}
```

(e)

```
int firstDigit(int x) {
    while (x > 10) x = x / 10;
    return x;
}
```

Problem 61 Write a function called *numEven* that returns the number of even digits in a positive integer parameter.

For example, a program that uses the function *numEven* follows.

```
int main() {
   cout << numEven(23) << endl;       // prints 1
   cout << numEven(1212) << endl;     // prints 2
   cout << numEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int numEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return numEven(x/10) + 1;
   return numEven(x/10);
}
```

Problem 62 Write a function called *lastEven* that returns the last even digit in a positive integer parameter. It should return 0 if there are no even digits.

For example, a program that uses the function *lastEven* follows.

```
int main() {
   cout << lastEven(23) << endl;       // prints 2
   cout << lastEven(1214) << endl;     // prints 4
   cout << lastEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int lastEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return x % 10;
   return lastEven(x / 10);
}
```

Problem 63 Write a function called *sumEven* that returns the sum of the even digits in a positive integer parameter.

For example, a program that uses the function *sumEven* follows.

```
int main() {
   cout << sumEven(23) << endl;       // prints 2
   cout << sumEven(1212) << endl;     // prints 4
   cout << sumEven(777) << endl;      // prints 0, because there are none
   return 0;
}
```

Answer:

```
int sumEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return x%10 + sumEven(x/10);
   return sumEven(x/10);
}
```

Problem 64 Write a function called *lastOdd* that returns the last odd digit in a positive integer parameter. It should return 0 if there are no odd digits.

For example, a program that uses the function *lastOdd* follows.

```
int main() {
   cout << lastOdd(23) << endl;       // prints 3
   cout << lastOdd(1254) << endl;     // prints 5
   cout << lastOdd(666) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int lastOdd(int x) {
   if (x <= 0) return 0;
   if (x % 2 != 0) return x%10;
   return lastOdd(x/10);
}
```

Problem 65 Write a function called *firstEven* that returns the first even digit in a positive integer parameter. It should return -1 if there are no even digits.

For example, a program that uses the function *firstEven* follows.

```
int main() {
   cout << firstEven(23) << endl;       // prints 2
   cout << firstEven(1416) << endl;     // prints 4
   cout << firstEven(777) << endl;      // prints -1
   return 0;
}
```

Answer:

```
int firstEven(int x) {
   if (x <= 0) return -1;
   if (firstEven(x/10) >= 0) return firstEven(x/10);
   if (x % 2 == 0) return x % 10;
   return -1;
}
```

Problem 66 Write a function called *evenLessOdd* that returns the sum of the even valued digit minus the sum of the odd valued digits in a positive integer parameter.

For example, a program that uses the function *evenLessOdd* follows.

```
int main() {
   cout << evenLessOdd(43) << endl;       // prints 1
   cout << evenLessOdd(9876) << endl;     // prints -2
   cout << evenLessOdd(777) << endl;      // prints -21
   return 0;
}
```

Answer:

```
int evenLessOdd(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return evenLessOdd(x/10) + x % 10;
   return evenLessOdd(x/10) - x % 10;
}
```

Problem 67 Write a function called *firstOdd* that returns the first odd digit in a positive integer parameter. It should return -1 if there are no odd digits.

For example, a program that uses the function *firstOdd* follows.

```
int main() {
   cout << firstOdd(21) << endl;       // prints 1
   cout << firstOdd(3456) << endl;     // prints 3
   cout << firstOdd(666) << endl;      // prints -1
   return 0;
}
```

Answer:

```
int firstOdd(int x) {
   if (x <= 0) return -1;
   if (firstOdd(x/10) >= 0) return firstOdd(x/10);
   if (x % 2 == 1) return x % 10;
   return -1;
}
```

Problem 68 Write a function called *oddLessEven* that returns the sum of the odd valued digits minus the sum of the even valued digits in a positive integer parameter.

For example, a program that uses the function *oddLessEven* follows.

```
int main() {
   cout << oddLessEven(23) << endl;       // prints 1
   cout << oddLessEven(1234) << endl;     // prints -2
   cout << oddLessEven(777) << endl;      // prints 21
   return 0;
}
```

Answer:

```
int oddLessEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 1) return x % 10 + oddLessEven(x/10);
   return oddLessEven(x/10) - x % 10;
}
```

Problem 69 Consider the following C++ program.

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

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[2][3] = {{1,2,3}, {3,4,5}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;      // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,1,2) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
4
```

(b) What is the output at line (b)?

Answer:

```
1
4
```

(c) What is the output at line (c)?

Answer:

```
5
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
6
```

Problem 70 Consider the following C++ program.

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

int up(int a[][3], int x, int y) {
    if (y < 2) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{3,2,1}, {0,3,6}}, a = 0;
    cout << x[a][a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][2 - i] << endl;    // line (b)
    cout << x[x[x[0][2]][0]][0] << endl;                        // line (c)
    up(x,1,1);                                                  // line (d)
    cout << up(x,1,2) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3
```

(b) What is the output at line (b)?

Answer:

```
1
3
```

(c) What is the output at line (c)?

Answer:

```
3
```

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

```
6
6
```

Problem 71 Consider the following C++ program.

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

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 1) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}
```

```
int main() {
    int x[2][3] = {{0,1,2}, {4,5,6}}, a = 0;
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;      // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                      // line (c)
    cout << up(x,1,1) << endl;                                // line (d)
    up(x,1,2);                                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
5
```

(b) What is the output at line (b)?

Answer:

```
0
5
```

(c) What is the output at line (c)?

Answer:

```
1
```

(d) What is the output at line (d)?

Answer:

```
5
6
```

(e) What is the output at line (e)?

Answer:

Problem 72 Consider the following C++ program.

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

int up(int a[][3], int x, int y) {
    if (y < 2) return a[1-x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{2,1,0}, {0,4,8}}, a = 0;
    cout << x[a][2*a] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;        // line (b)
    cout << x[0][x[x[0][1]][0]]<< endl;                         // line (c)
    up(x,1,2);                                                  // line (d)
    cout << up(x,1,1) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
2
```

(b) What is the output at line (b)?

Answer:

```
2
4
```

(c) What is the output at line (c)?

Answer:

```
2
```

(d) What is the output at line (d)?

Answer:

```
8
```

(e) What is the output at line (e)?

Answer:

```
0
```

Problem 73 Consider the following C++ program.

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

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[3][2] = {{1,2}, {3,3}, {4,5}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,2,1) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3
```

(b) What is the output at line (b)?

Answer:

```
1
3
```

(c) What is the output at line (c)?
Answer:

```
5
```

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

```
6
```

Problem 74 Consider the following C++ program.

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

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{3,2},{4,5},{0,1}}, a = 0;
    cout << x[a][a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;    // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                        // line (c)
    up(x,1,1);                                                  // line (d)
    cout << up(x,2,1) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
3
```

(b) What is the output at line (b)?
Answer:

```
0
5
```

(c) What is the output at line (c)?
Answer:

```
No output:
Code error, sorry
```

(d) What is the output at line (d)?
Answer:

```
5
```

(e) What is the output at line (e)?

Answer:

```
1
1
```

Problem 75 Consider the following C++ program.

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

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[3][2] = {{0,1}, {3,4}, {5,7}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;      // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,2,1) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
4
```

(b) What is the output at line (b)?

Answer:

```
0
4
```

(c) What is the output at line (c)?

Answer:

```
3
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
8
```

Problem 76 Consider the following C++ program.

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

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{2,3},{0,4},{1,5}}, a = 0;
    cout << x[a][a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;    // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                        // line (c)
    up(x,1,1);                                                  // line (d)
    cout << up(x,2,1) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
2
```

(b) What is the output at line (b)?

Answer:

```
1
4
```

(c) What is the output at line (c)?

Answer:

```
2
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
5
5
```

Problem 77 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 21.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical left edge and a horizontal bottom edge. Odd numbered rows of the triangle are made from the letter A and even numbered rows with the letter B, as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 21:   9
A
BB
AAA
BBBB
AAAAA
BBBBBB
AAAAAAA
BBBBBBBB
AAAAAAAAA
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 21:";
   cin >> n;

   if (n < 1 || n > 21) return 0;
   for (int r = 0; r < n; r++) {
      for (int c = 0; c <= r; c++)
         cout << (char) ('A' + r % 2);
      cout << endl;
   }
}
```

Problem 78 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 23.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical right edge and a horizontal top edge. Odd numbered rows of the triangle are made from the letter x and even numbered rows with the letter y, as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 23:   5
xxxxx
 yyyy
  xxx
   yy
    x
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 23:";
   cin >> n;

   if (n < 1 || n > 23) return 0;
   for (int r = 0; r < n; r++) {
```

```
      for (int c = 0; c < n; c++)
         if (c < r) cout << " ";
         else cout << (char) ('x' + r % 2);
      cout << endl;
   }
}
```

Problem 79 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 16.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical left edge and a horizontal bottom edge. Odd numbered columns of the triangle are made from the letter A and even numbered columns with the letter B, as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 16:   6
A
AB
ABA
ABAB
ABABA
ABABAB
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 16:";
   cin >> n;

   if (n < 1 || n > 16) return 0;
   for (int r = 0; r < n; r++) {
      for (int c = 0; c <= r; c++)
         cout << (char) ('A' + c % 2);
      cout << endl;
   }
}
```

Problem 80 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 18.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical right edge and a horizontal top edge. Odd numbered columns of the triangle are made from the letter x and even numbered columns with the letter y, as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 18:   5
xyxyx
 yxyx
  xyx
   yx
    x
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 18:";
   cin >> n;

   if (n < 1 || n > 18) return 0;
   for (int r = 0; r < n; r++) {
      for (int c = 0; c < n; c++)
         if (c < r) cout << " ";
         else cout << (char) ('x' + c % 2);
      cout << endl;
   }
}
```

Problem 81 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   cout << firstLetter("Freddy") << endl; // prints: F
   sort(a, 4);                            // prints: 159 265 314 358
   oddElements(a, 4);                     // prints: odd: 159 265
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **sqrt**.

Answer:

```
string sqrt(string x)
```

(b) Title line for **firstLetter**.

Answer:

```
char firstLetter(string x)
```

(c) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(d) Title line for **oddElements**.

Answer:

```
void oddElements(int a[], int capacity)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 82 Consider the following C++ program.

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

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
5
```

(b) line b:

```
15
```

(c) line c:

```
71
```

(d) line d:

```
37
```

(e) line e:

```
93
27
```

Problem 83 Write a function called *addThrees* that inserts a 3 after each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 13331333
   cout << addThrees(777) << endl;      // prints 737373
   return 0;
}
```

Answer:

```
int addThrees(int x) {
   if (x == 0) return 0;
   return 100 * addThrees(x / 10) + 10 * (x % 10) + 3;
}
```

Problem 84 Write a C++ function called *halfs* that divides each element of a 2-dimensional array (with two columns) by 2.

It should be possible to use your function in the following program.

```
main() {
   double data[2][2] = {{1, 2}, {3, 4}};
   halfs (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 1.5 2.0
}
```

Answer:

```
void halfs(double d[][2], int r, int c) {
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         d[i][j] = d[i][j] / 2;
      }
}
```

Problem 85 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 1, y = 10, z = 19;
   double b[5] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   b[1] = divide(z, y);                 // (a) sets b[1] to quotient 2
   reset(d[1][1], x);                   // (b) replaces d[1][1] by value of x
   cout << bigRow(d, 2, 2);             // (c) prints biggest row: 3 4
   printAll(b, 3);                      // (d) prints array: 1.9 2.3 3.0
   cout << add(d[0][0], b[2]) << endl;  // (e) prints the sum 4
   return 0;
}
```

(a) Title line for **divide**.

Answer:

```
double divide(int z, int y)
```

(b) Title line for **reset**.

Answer:

```
void reset(int &x, int y)
```

(c) Title line for **bigRow**.

Answer:

```
string bigRow(int d[][2], int r, int c)
```

(d) Title line for **printAll**.

Answer:

```
void printAll(double b[], int cap)
```

(e) Title line for **add**.

Answer:

```
int add(int x, double y)
```

Problem 86 Consider the following C++ program.

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

string fun(int x) {
   string ans = "9876543210";
   if (x <= 10) return "0";
   if ((x <= 30) || (x > 10000)) return ans.substr(x % 10);
   if ((x >= 0) && (x < 100)) return "x+1";
   return ans.substr(x%4, x%4);
}

int nuf(int &x) {
  cout << x << endl;
  x = x * x - 3;
  return x;
}

int main() {
    int x = 2;
    cout << fun(23) << endl;    // line (a)
    cout << fun(233) << endl;   // line (b)
    cout << fun(2333) << endl;  // line (c)
    nuf(x);                    // line (d)
    cout << nuf(x) << endl;    // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
6543210
```

(b) What is the output at line (b)?

Answer:

```
8
```

(c) What is the output at line (c)?

Answer:

```
8
```

(d) What is the output at line (d)?

Answer:

```
2
```

(e) What is the output at line (e)?

Answer:

```
1
-2
```

Problem 87 Write a function called *smallRow* that calculates and returns the smallest possible sum of entries of any row in a 2-dimensional array.

For example, a program that uses the function *smallRow* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << smallRow (x, 2, 3) << endl;
     // from the 2-d array x that has size 2 x 3, find the smallest row sum
     // output will be 8 since row #0 contains 3, 1 and 4 is smallest.
   return 0;
}
```

Answer:

```
int smallRow(int x[][3], int r, int c) {
   int ans;
   for (int row = 0; row < r; row++) {
      int sum = 0;
      for (int col = 0; col < c; col++)
         sum += x[row][col];
      if (row == 0 || sum < ans) ans = sum;
   }
   return ans;
}
```

Problem 88 Write a function called *bond* that changes any sequence of digits 006 to 007 in a positive integer parameter.

For example, a program that uses the function *bond* follows.

```
int main() {
   cout << bond(4006) << endl;            // prints 4007
   cout << bond(4006006) << endl;         // prints 4007007
   cout << bond(106) << endl;             // prints 106
   cout << bond(1006) + 1 << endl;        // prints 1008
   return 0;
}
```

Answer:

```
int bond(int x) {
   if (x <= 0) return 0;
   if (x % 1000 == 6) return 1000 * bond(x / 1000) + 7;
   return 10 * bond(x / 10) + x % 10;
}
```

Problem 89 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 24.
2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical right edge and a horizontal top edge. The right edge is formed from the letter A, next to it is a vertical line formed from the letter B, then one formed from the letter C and so on. The top edge is also formed from the letter A, just below it is a line formed from the letter B and so on as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 24:   8
AAAAAAAA
 BBBBBBA
  CCCCBA
   DDCBA
    DCBA
     CBA
      BA
       A
```

Answer:

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

int main() {
   int n;
   char picture[24][24];
   cout << "Give me an integer between 1 and 24:";
   cin >> n;

   if (n < 1 || n > 24) {
      cout << "Illegal." << endl;
      return 0;
   }
   int mid = (n + 1) / 2;

   char letter = 'A';
   for (int step = 0; step < mid; step++) {
      for (int r = step; r < n - step; r++)
         for (int c = step; c < n - step; c++)
            picture[r][c] = letter;
      letter++;
   }

   for (int r = 0; r <n; r++) {
      for (int c = 0; c < n; c++)
         if (r <= c) cout << picture[r][c];
         else cout << " ";
      cout << endl;
   }
}
```

Problem 90 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   sqrt("FFrreedd");                    // prints: Fred
   firstLetter("Freddy");               // prints: F
   sort(a, 4);                          // prints: 159 265 314 358
   cout << oddElements(a, 4);           // prints: odd: 159 265
```

```
   swap(a[1], a[2]);                      // swaps elements
   return 0;
}
```

(a) Title line for **sqrt**.
Answer:

```
void sqrt(string x)
```

(b) Title line for **firstLetter**.
Answer:

```
void firstLetter(string x)
```

(c) Title line for **sort**.
Answer:

```
void sort(int a[], int capacity)
```

(d) Title line for **oddElements**.
Answer:

```
string oddElements(int a[], int capacity)
```

(e) Title line for **swap**.
Answer:

```
void swap(int &x, int &y)
```

Problem 91 Consider the following C++ program.

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

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
4
```

(b) line b:

```
24
```

(c) line c:

```
62
```

(d) line d:

```
46
```

(e) line e:

```
84
32
```

Problem 92 Write a function called *addThrees* that inserts a 3 before each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 31333133
   cout << addThrees(777) << endl;      // prints 373737
   return 0;
}
```

Answer:

```
int addThrees(int x) {
   if (x == 0) return 0;
   return 100 * addThrees(x / 10) + 30 + x % 10;
}
```

Problem 93 Write a C++ function called *roots* that replaces each element of an array by its root. It should be possible to use your function in the following program.

```
main() {
   double data[3] = {1.0, 4.0, 9.0};
   roots (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 2 3
}
```

Answer:

```
#include <cmath>

void roots(double d[], int cap) {
   for (int i = 0; i < cap; i++)
      d[i] = sqrt(d[i]);
}
```

Problem 94 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   x = diffTwo(y, b[0]);                  // (a) sets x to approx difference 1
   swap(d[1][1], x);                      // (b) swaps x with value of d[1][1]
   cout << biggest(d, 2, 2);              // (c) prints biggest row: 3 4
   printThree(b);                         // (d) prints three entries: 1.9 2.3 3.0
   summit(b[2], d[0][0]);                 // (e) prints the sum 4
   return 0;
}
```

(a) Title line for **diffTwo**.

Answer:

```
int diffTwo(int y, double z)
```

(b) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(c) Title line for **biggest**.

Answer:

```
string biggest(int d[][2], int r, int c)
```

(d) Title line for **printThree**.

Answer:

```
void printThree(double b[])
```

(e) Title line for **summit**.

Answer:

```
void summit(double x, int y)
```

Problem 95 Consider the following C++ program.

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

string fun(int x) {
   string ans = "0123456789";
   if (x <= 0) return "0";
   if ((x <= 10) || (x > 10000)) return ans.substr(x % 10);
   if ((x >= 0) && (x < 100)) return "x+1";
   return ans.substr(x%4, x%4);
}

int nuf(int &x) {
  cout << x << endl;
  x = x * x;
  return x - 6;
}
```

```
int main() {
    int x = 4;
    cout << fun(3) << endl;    // line (a)
    cout << fun(32) << endl;   // line (b)
    cout << fun(323) << endl;  // line (c)
    nuf(x);                    // line (d)
    cout << nuf(x) << endl;    // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3456789
```

(b) What is the output at line (b)?

Answer:

```
x+1
```

(c) What is the output at line (c)?

Answer:

```
345
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
16
250
```

Problem 96 Write a function called *smallCol* that calculates and returns the smallest possible sum of entries of any column in a 2-dimensional array.

For example, a program that uses the function *smallCol* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << smallCol (x, 2, 3) << endl;
     // from the 2-d array x that has size 2 x 3, find the smallest col sum
     // output will be 4  since col #0 contains 3 and 1 is smallest.
   return 0;
}
```

Answer:

```
int smallCol(int x[][3], int r, int c) {
   int ans;
   for (int col = 0; col < c; col++) {
      int sum = 0;
      for (int row = 0; row < r; row++)
         sum += x[row][col];
      if (col == 0 || sum < ans) ans = sum;
   }
   return ans;
}
```

Problem 97 Write a function called *bond* that inserts a digit 0 before any digit pair 07 in a positive integer parameter.

For example, a program that uses the function *bond* follows.

```
int main() {
   cout << bond(407) << endl;        // prints 4007
   cout << bond(401) << endl;        // prints 401
   cout << bond(40707) << endl;      // prints 4007007
   cout << bond(107) + 1 << endl;    // prints 1008
   return 0;
}
```

Answer:

```
int bond(int x) {
   if (x <= 0) return 0;
   if (x % 100 == 7) return 1000 * bond(x / 10) + 7;
   return 10 * bond(x / 10) + x % 10;
}
```

Problem 98 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 23.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical right edge and a horizontal bottom edge. The right edge is formed from the letter A, next to it is a vertical line formed from the letter B, then one formed from the letter C and so on. The bottom edge is also formed from the letter A, just above it is a line formed from the letter B and so on as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 23:   9
        A
       BA
      CBA
     DCBA
    EDCBA
   DDDCBA
  CCCCCBA
 BBBBBBBA
AAAAAAAAA
```

Answer:

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

int main() {
   int n;
   char picture[23][23];
   cout << "Give me an integer between 1 and 23:";
   cin >> n;

   if (n < 1 || n > 23) {
      cout << "Illegal." << endl;
      return 0;
   }
   int mid = (n + 1) / 2;
```

```
   char letter = 'A';
   for (int step = 0; step < mid; step++) {
      for (int r = step; r < n - step; r++)
         for (int c = step; c < n - step; c++)
            picture[r][c] = letter;
      letter++;
   }

   for (int r = 0; r <n; r++) {
      for (int c = 0; c < n; c++)
         if ((r + c) >= (n - 1)) cout << picture[r][c];
         else cout << " ";
      cout << endl;
   }
}
```

Problem 99 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << firstLetter("Freddy") << endl; // prints: F
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   oddElements(a, 4);                     // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

```
char firstLetter(string x)
```

(b) Title line for **sqrt**.

Answer:

```
string sqrt(string x)
```

(c) Title line for **oddElements**.

Answer:

```
void oddElements(int a[], int capacity)
```

(d) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 100 Consider the following C++ program.

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

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
3
```

(b) line b:

```
13
```

(c) line c:

```
51
```

(d) line d:

```
35
```

(e) line e:

```
73
21
```

Problem 101 Write a function called *addTwos* that inserts a 2 after each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 32
   cout << addTwos(1212) << endl;     // prints 12221222
   cout << addTwos(777) << endl;      // prints 727272
   return 0;
}
```

Answer:

```
int addTwos(int x) {
   if (x == 0) return 0;
   return 100 * addTwos(x / 10) + 10 * (x % 10) + 2;
}
```

Problem 102 Write a C++ function called *squares* that replaces each element of a 2-dimensional array (with two columns) by its square.

It should be possible to use your function in the following program.

```
main() {
   int data[2][2] = {{1, 2}, {3, 4}};
   squares (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 9 16
}
```

Answer:

```
void squares(int d[][2], int r, int c) {
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         d[i][j] = d[i][j] * d[i][j];
      }
}
```

Problem 103 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   cout << twoD(y, b[0]) << endl;         // (a) prints difference: 0.9
   y = addUp(d[1][1], y);                 // (b) sets y to sum 4 + 1
   cout << firstElt(d, 2, 2);             // (c) prints last element: 1
   b[2] = av(b, 3);                       // (d) sets as average
   setOne(b[2], d[0][0]);                 // (e) sets both to 1
   return 0;
}
```

(a) Title line for **twoD**.

Answer:

```
double twoD(int a, double b)
```

(b) Title line for **addUp**.

Answer:

```
int addUp(int x, int y)
```

(c) Title line for **firstElt**.

Answer:

```
int firstElt(int d[][2], int r, int c)
```

(d) Title line for **av**.

Answer:

```
double av(double b[], int cap)
```

(e) Title line for **setOne**.

Answer:

```
void setOne(double &x, int &y)
```

Problem 104 Consider the following C++ program.

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

string fun(int x) {
   string ans = "0123456789";
   if (x <= 10) return "0";
   if ((x <= 30) || (x > 10000)) return ans.substr(x % 10);
   if ((x >= 0) && (x < 100)) return "x+1";
   return ans.substr(x%4, x%4);
}

int nuf(int &x) {
  cout << x << endl;
  x = x * x;
  return x;
}

int main() {
    int x = 2;
    cout << fun(2) << endl;    // line (a)
    cout << fun(22) << endl;   // line (b)
    cout << fun(222) << endl;  // line (c)
    nuf(x);                    // line (d)
    cout << nuf(x) << endl;    // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
0
```

(b) What is the output at line (b)?

Answer:

```
23456789
```

(c) What is the output at line (c)?

Answer:

```
23
```

(d) What is the output at line (d)?

Answer:

```
2
```

(e) What is the output at line (e)?

Answer:

```
4
16
```

Problem 105 Write a function called *bigRow* that calculates and returns the biigest possible sum of entries of any row in a 2-dimensional array.

For example, a program that uses the function *bigRow* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << bigRow (x, 2, 3) << endl;
     // from the 2-d array x that has size 2 x 3, find the biggest row sum
     // output will be 15 since row #1 contains 1, 5 and 9 is biggest.
   return 0;
}
```

Answer:

```
int bigRow(int x[][3], int r, int c) {
   int ans;
   for (int row = 0; row < r; row++) {
      int sum = 0;
      for (int col = 0; col < c; col++)
         sum += x[row][col];
      if (row == 0 || sum > ans) ans = sum;
   }
   return ans;
}
```

Problem 106 Write a function called *bond* that the insert the digit 7 after any pair of zero digits in a positive integer parameter.

For example, a program that uses the function *bond* follows.

```
int main() {
   cout << bond(400) << endl;        // prints 4007
   cout << bond(401) << endl;        // prints 41
   cout << bond(4007) << endl;       // prints 40077
   cout << bond(400) + 1 << endl;    // prints 4008
   return 0;
}
```

Answer:

```
int bond(int x) {
   if (x <= 0) return 0;
   if (x % 100 == 0) return 1000 * bond(x / 100) + 7;
   return 10 * bond(x / 10) + x % 10;
}
```

Problem 107 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 22.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical left edge and a horizontal top edge. The left edge is formed from the letter A, next to it is a vertical line formed from the letter B, then one formed from the letter C and so on. The top edge is also formed from the letter A, just below it is a line formed from the letter B and so on as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 22:   8
AAAAAAAA
ABBBBBB
ABCCCC
ABCDD
ABCD
ABC
AB
A
```

Answer:

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

int main() {
   int n;
   char picture[22][22];
   cout << "Give me an integer between 1 and 22:";
   cin >> n;

   if (n < 1 || n > 22) {
      cout << "Illegal." << endl;
      return 0;
   }
   int mid = (n + 1) / 2;

   char letter = 'A';
   for (int step = 0; step < mid; step++) {
      for (int r = step; r < n - step; r++)
         for (int c = step; c < n - step; c++)
            picture[r][c] = letter;
      letter++;
   }

   for (int r = 0; r <n; r++) {
      for (int c = 0; c < n; c++)
         if ((r + c) < n) cout << picture[r][c];
         else cout << " ";
      cout << endl;
   }
}
```

Problem 108 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   firstLetter("Freddy");                 // prints: F
   sqrt("FFrreedd");                      // prints: Fred
   cout << oddElements(a, 4);             // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   swap(a[1], a[2]);                      // swaps elements
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

```
void firstLetter(string x)
```

(b) Title line for **sqrt**.
Answer:

```
void sqrt(string x)
```

(c) Title line for **oddElements**.
Answer:

```
string oddElements(int a[], int capacity)
```

(d) Title line for **sort**.
Answer:

```
void sort(int a[], int capacity)
```

(e) Title line for **swap**.
Answer:

```
void swap(int &x, int &y)
```

Problem 109 Consider the following C++ program.

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

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
2
```

(b) line b:

```
22
```

(c) line c:

```
42
```

(d) line d:

```
44
```

(e) line e:

```
64
24
```

Problem 110 Write a function called *addTwos* that inserts a 2 before each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 23
   cout << addTwos(1212) << endl;     // prints 21222122
   cout << addTwos(777) << endl;      // prints 272727
   return 0;
}
```

Answer:

```
int addTwos(int x) {
   if (x == 0) return 0;
   return 100 * addTwos(x / 10) + 20 + x % 10;
}
```

Problem 111 Write a C++ function called *cubes* that replaces each element of an array by its cube. It should be possible to use your function in the following program.

```
main() {
   int data[3] = {1, 2, 3};
   cubes (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 8 27
}
```

Answer:

```
void cubes(int d[], int cap) {
   for (int i = 0; i < cap; i++)
      d[i] = d[i] * d[i] * d[i];
}
```

Problem 112 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double x = 0, y = 1, z = 2;
   int b[3] = {1, 2, 3};
   double d[2][2] = {{1.9,2},{3.9,4}};

   cout << add3(b[0], y, d[0][0]) << endl;// (a) prints sum: 3.9
   y = addUp(d[1][1], x) + 1;             // (b) sets y to sum 4.0 + 0 + 1
   cout << col(d, 2, 2, 0);               // (c) prints column 0 as: 1.9,3.9
   b[0] = min(b, 3);                      // (d) sets as min element
   decrease(b[2], d[0][0]);               // (e) decreases both by 1
   return 0;
}
```

(a) Title line for **add3**.
Answer:

```
double add3(int a, double b, double c)
```

(b) Title line for **addUp**.
Answer:

```
double addUp(double x, double y)
```

(c) Title line for **col**.
Answer:

```
string col(double d[][2], int r, int c, int colNumber)
```

(d) Title line for **min**.
Answer:

```
int min(int b[], int cap)
```

(e) Title line for **decrease**.
Answer:

```
void decrease(int &x, double &y)
```

Problem 113 Consider the following C++ program.

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

string fun(int x) {
   string ans = "0123456789";
   if (x <= 10) return "0";
   if ((x <= 30) || (x > 10000)) return ans.substr(x % 10);
   if ((x >= 0) && (x < 100)) return "x+1";
   return ans.substr(x%4, x%4);
}

int nuf(int &x) {
  cout << x << endl;
  x = x * x;
  return x;
}

int main() {
    int x = 4;
    cout << fun(3) << endl;    // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << fun(333) << endl;  // line (c)
    nuf(x);                    // line (d)
    cout << nuf(x) << endl;    // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
0
```

(b) What is the output at line (b)?

Answer:

```
x+1
```

(c) What is the output at line (c)?

Answer:

```
1
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
16
256
```

Problem 114 Write a function called *bigCol* that calculates and returns the biigest possible sum of entries of any column in a 2-dimensional array.

For example, a program that uses the function *bigCol* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << bigCol (x, 2, 3) << endl;
     // from the 2-d array x that has size 2 x 3, find the biggest col sum
     // output will be 13 since col #2 contains 4 and 9 is biggest.
   return 0;
}
```

Answer:

```
int bigCol(int x[][3], int r, int c) {
   int ans;
   for (int col = 0; col < c; col++) {
      int sum = 0;
      for (int row = 0; row < r; row++)
         sum += x[row][col];
      if (col == 0 || sum > ans) ans = sum;
   }
   return ans;
}
```

Problem 115 Write a function called *bond* that the insert the digits 07 after each digit 0 in a positive integer parameter.

For example, a program that uses the function *bond* follows.

```
int main() {
   cout << bond(40) << endl;        // prints 4007
   cout << bond(41) << endl;        // prints 41
   cout << bond(400) << endl;       // prints 4007007
   cout << bond(10) + 1 << endl;    // prints 1008
   return 0;
}
```

Answer:

```
int bond(int x) {
   if (x <= 0) return 0;
   if (x % 10 == 0) return 1000 * bond(x / 10) + 7;
   return 10 * bond(x / 10) + x % 10;
}
```

Problem 116 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 21.

2. It terminates if the user supplies an illegal value for n.

3. It prints out a triangular picture with n rows like the one shown in the example (below). The triangle has a vertical left edge and a horizontal bottom edge. The left edge is formed from the letter A, next to it is a vertical line formed from the letter B, then one formed from the letter C and so on. The bottom edge is also formed from the letter A, just above it is a line formed from the letter B and so on as in the example.

Here is an example of how the program should work:

```
Give me an integer between 1 and 21:   9
A
AB
ABC
ABCD
ABCDE
ABCDDD
ABCCCCC
ABBBBBBB
AAAAAAAAA
```

Answer:

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

int main() {
   int n;
   char picture[21][21];
   cout << "Give me an integer between 1 and 21:";
   cin >> n;

   if (n < 1 || n > 21) {
      cout << "Illegal." << endl;
      return 0;
   }
   int mid = (n + 1) / 2;

   char letter = 'A';
   for (int step = 0; step < mid; step++) {
      for (int r = step; r < n - step; r++)
         for (int c = step; c < n - step; c++)
            picture[r][c] = letter;
      letter++;
   }

   for (int r = 0; r <n; r++) {
      for (int c = 0; c < n; c++)
         if (r >= c) cout << picture[r][c];
```

```
         else cout << " ";
      cout << endl;
   }
}
```

Problem 117 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   undouble(11223344);                    // prints: 1234
   firstDigit(65536);                     // prints: Six
   printSorted(a, 4);                     // prints: Freddy Jack Kelly Max
   cout << join(a[1], a[3]) << endl;      // prints: MaxJack
   randomWords(a, 4);                     // assigns new random values to array
   return 0;
}
```

(a) Title line for **undouble**.

Answer:

```
void undouble(int x)
```

(b) Title line for **firstDigit**.

Answer:

```
void firstDigit(int x)
```

(c) Title line for **printSorted**.

Answer:

```
void printSorted(string a[], int capacity)
```

(d) Title line for **join**.

Answer:

```
string join(string x, string y)
```

(e) Title line for **randomWords**.

Answer:

```
void randomWords(string a[], int capacity)
```

Problem 118 Consider the following C++ program.

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

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}
```

```
int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
5
```

(b) line b:

```
62
```

(c) line c:

```
02
```

(d) line d:

```
17
```

(e) line e:

```
73
21
```

Problem 119 Write a function called *killTwos* that deletes all digits that are multiples of 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(2400) << endl;     // prints 0, because no digits are left
   return 0;
}
```

Answer:

```
int killTwos(int x) {
   if (x == 0) return 0;
   if ((x % 10) % 2 == 0) return killTwos(x / 10);
   return 10 * killTwos(x / 10) + x % 10;
}
```

Problem 120 Write a C++ function called *numOdd* that returns the number of odd elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 11s are odd).

```
main() {
   int data[2][4] = {{11, 12, 14, 0}, {32, 12, 132, 11}};
   int x;
   x = numOdd (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The number of odds is: " << x  << endl;
}
```

Answer:

```
int numOdd(int d[][4], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         if ((d[i][j] % 2) != 0) count++;
      }
   return count;
}
```

Problem 121 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   int b[3] = {1, 2, 3};
   double d[2][2] = {{1.9,2},{3.9,4}};

   cout << sum3(b[0], y, d[0][0]) << endl;// (a) prints sum: 3.9
   y = addUp(x, d[1][1]) + 1;             // (b) sets y to sum 0 + 4.0 + 1
   cout << col0(d, 2, 2);                 // (c) prints column as: 1.9,3.9
   b[0] = max(b, 3);                      // (d) sets as max element
   increase(b[2], d[0][0]);               // (e) increases both by 1
   return 0;
}
```

(a) Title line for **sum3**.

Answer:

```
double sum3(int a, int b, double c)
```

(b) Title line for **addUp**.

Answer:

```
int addUp(int x, double y)
```

(c) Title line for **col0**.

Answer:

```
string col0(double d[][2], int r, int c)
```

(d) Title line for **max**.

Answer:

```
int max(int b[], int cap)
```

(e) Title line for **increase**.

Answer:

```
void increase(int &x, double &y)
```

Problem 122 Consider the following C++ program.

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

string fun(int x) {
   string ans = "012345";
   if (x <= 0) return "";
   if ((x >= 30) && (x < 1000)) return ans.substr(x % 5);
   if ((x >= 0) || (x < 100)) return "xyz";
   return ans;
}

int up(int &x) {
  x += 3;
  cout << x << endl;
  return x - 1;
}

int main() {
    int x = 7;
    cout << fun(0) << endl;    // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << fun(3003) << endl; // line (c)
    up(x);                     // line (d)
    cout << up(x) << endl;     // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

```
345
```

(c) What is the output at line (c)?

Answer:

```
xyz
```

(d) What is the output at line (d)?

Answer:

```
10
```

(e) What is the output at line (e)?

Answer:

```
13
12
```

Problem 123 Write a function called *rowProd* that calculates and returns the product of the entries of a specified row of a 2-dimensional array.

For example, a program that uses the function *rowProd* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << rowProd (x, 2, 3, 1) << endl;
     // from the 2-d array x that has size 2 x 3, find the product of row 1
     // output will be 45 since row #1 contains 1, 5 and 9.
   return 0;
}
```

Answer:

```
int rowProd(int x[][3], int r, int c, int row) {
   int ans = 1;
   for (int j = 0; j < c; j++) ans *= x[row][j];
   return ans;
}
```

Problem 124 Write a function called *numOdd* that the returns the number of digits in a positive integer parameter that are odd.

For example, a program that uses the function *numOdd* follows.

```
int main() {
   cout << numOdd(777) << endl;        // prints 3
   cout << numOdd(747) << endl;        // prints 2
   cout << numOdd(42) << endl;         // prints 0
   return 0;
}
```

Answer:

```
int numOdd(int x) {
   if (x <= 0) return 0;
   if (x % 2 != 0) return numOdd(x / 10) + 1;
   return numOdd(x / 10);
}
```

Problem 125 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an odd integer n that is between 1 and 19.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n characters in the first row). Reading from the right, along each row the characters to be used is the sequence of uppercase letters A, B, C, ..., and so on.

Here is an example of how the program should work:

```
Give me an odd integer between 1 and 19:   7
GFEDCBA
 EDCBA
  CBA
   A
```

Answer:

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

int main() {
   int n;
   cout << "Give me an odd integer between 1 and 19:";
   cin >> n;

   while (n < 1 || n > 19 || (n % 2) != 1) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }

   int mid = n / 2;

   for (int r = mid; r >= 0; r--) {
      char out = 'A' + 2 * r;
      for (int c = 0; c < n; c++)
         if ((c >= mid - r) && (c <= mid + r)) {
            cout << out;
            out--;
         }
         else cout << " ";
      cout << endl;
   }
}
```

Problem 126 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << undouble(11223344);            // prints: 1234
   cout << firstDigit(65536) << endl;     // prints: Six
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   cout << halfString(a[0]) << endl;      // prints: Fre
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **undouble**.

Answer:

```
int undouble(int x)
```

(b) Title line for **firstDigit**.

Answer:

```
string firstDigit(int x)
```

(c) Title line for **sort**.

Answer:

```
void sort(string a[], int capacity)
```

(d) Title line for **halfString**.

Answer:

```
string halfString(string x)
```

(e) Title line for **randomWord**.

Answer:

```
string randomWord()
```

Problem 127 Consider the following C++ program.

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

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
4
```

(b) line b:

```
52
```

(c) line c:

```
12
```

(d) line d:

```
26
```

(e) line e:

```
63
18
```

Problem 128 Write a function called *twos* that deletes all digits that are not multiples of 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

```
int twos(int x) {
   if (x == 0) return 0;
   if ((x % 10) % 2 != 0) return twos(x / 10);
   return 10*twos(x / 10) + x % 10;
}
```

Problem 129 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][4] = {{11, 12, 11, 5}, {6, 3, 12, 13}};
   int x;
   x = range (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

```
int range(int d[][4], int r, int c) {
   int max = d[0][0];
   int min = d[0][0];
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
          if (d[i][j] < min) min = d[i][j];
          if (d[i][j] > max) max = d[i][j];
      }
   return max - min;
}
```

Problem 130 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   cout << twoD(b[0], y) << endl;         // (a) prints difference: 0.9
   y = addUp(x, d[1][1]);                 // (b) sets y to sum 0 + 4
   cout << lastElt(d, 2, 2);              // (c) prints last element: 4
   b[0] = average(b, 3);                  // (d) sets as average
   setZero(b[2], d[0][0]);                // (e) sets both to 0
   return 0;
}
```

(a) Title line for **twoD**.
Answer:

```
double twoD(double a, int b)
```

(b) Title line for **addUp**.
Answer:

```
int addUp(int x, int y)
```

(c) Title line for **lastElt**.
Answer:

```
int lastElt(int d[][2], int r, int c)
```

(d) Title line for **average**.
Answer:

```
double average(double b[], int cap)
```

(e) Title line for **setZero**.
Answer:

```
void setZero(double &x, int &y)
```

Problem 131 Consider the following C++ program.

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

string fun(int x) {
   string ans = "9876543210";
   if (x <= 0) return "5";
   if ((x >= 30) && (x < 1000)) return ans.substr(x % 10);
   if ((x >= 0) || (x < 100)) return "1+x";
   return ans + ans;
}

int up(int &x) {
  x++;
  cout << x << endl;
  return x - 2;
}

int main() {
    int x = 2;
    cout << fun(0) << endl;    // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << fun(3003) << endl; // line (c)
    up(x);                     // line (d)
    cout << up(x) << endl;     // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
5
```

(b) What is the output at line (b)?

Answer:

```
6543210
```

(c) What is the output at line (c)?

Answer:

```
1+x
```

(d) What is the output at line (d)?

Answer:

```
3
```

(e) What is the output at line (e)?

Answer:

```
4
2
```

Problem 132 Write a function called *colProd* that calculates and returns the product of the entries of a specified column in a 2-dimensional array.

For example, a program that uses the function *colProd* follows.

```
int main() {
   int x[2][3] = {{3, 2, 4}, {1, 5, 9}};
   cout << colProd (x, 2, 3, 1) << endl;
     // from the 2-d array x that has size 2 x 3, find the product of column 1
     // output will be 10 since col #1 contains 2 and 5.
   return 0;
}
```

Answer:

```
int colProd(int x[][3], int r, int c, int col) {
   int ans = 1;
   for (int i = 0; i < r; i++) ans *= x[i][col];
   return ans;
}
```

Problem 133 Write a function called *numBig* that the returns the number of digits in a positive integer parameter that are greater than or equal to 7.

For example, a program that uses the function *numBig* follows.

```
int main() {
   cout << numBig(777) << endl;        // prints 3
   cout << numBig(747) << endl;        // prints 2
   cout << numBig(41) << endl;         // prints 0
   return 0;
}
```

Answer:

```
int numBig(int x) {
   if (x <= 0) return 0;
   if (x % 10 >= 7) return numBig(x / 10) + 1;
   return numBig(x / 10);
}
```

Problem 134 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an odd integer n that is between 1 and 23.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n characters in the last row). Reading from the right, along each row the characters to be used is the sequence of uppercase letters A, B, C, ..., and so on.

Here is an example of how the program should work:

```
Give me an odd integer between 1 and 23:   7
   A
  CBA
 EDCBA
GFEDCBA
```

Answer:

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

int main() {
   int n;
   cout << "Give me an odd integer between 1 and 23:";
   cin >> n;

   while (n < 1 || n > 23 || (n % 2) != 1) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }

   int mid = n / 2;

   for (int r = 0; r <= mid; r++) {
      char out = 'A' + 2 * r;
      for (int c = 0; c < n; c++)
         if ((c >= mid - r) && (c <= mid + r)) {
            cout << out;
            out--;
         }
         else cout << " ";
      cout << endl;
   }
}
```

Problem 135 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   firstDigit(65536);                    // prints: Six
   undouble(11223344);                   // prints: 1234
```

```
   cout << join(a[1], a[3]) << endl;      // prints: MaxJack
   printSorted(a, 4);                     // prints: Freddy Jack Kelly Max
   randomWords(a, 4);                     // assigns new random values to array
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

```
void firstDigit(int x)
```

(b) Title line for **undouble**.

Answer:

```
void undouble(int x);
```

(c) Title line for **join**.

Answer:

```
string join(string x, string y)
```

(d) Title line for **printSorted**.

Answer:

```
void printSorted(string a[], int capacity)
```

(e) Title line for **randomWords**.

Answer:

```
void randomWords(string a[], int capacity)
```

Problem 136 Consider the following C++ program.

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

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
3
```

(b) line b:

```
42
```

(c) line c:

```
02
```

(d) line d:

```
15
```

(e) line e:

```
53
15
```

Problem 137 Write a function called *killTwos* that deletes all digits that are equal to 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(222) << endl;      // prints 0, because no digits are left
   return 0;
}
```

Answer:

```
int killTwos(int x) {
   if (x == 0) return 0;
   if (x % 10 == 2) return killTwos(x / 10);
   return 10 * killTwos(x / 10) + x % 10;
}
```

Problem 138 Write a C++ function called *numEven* that returns the number of even elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 12s are even).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = numEven (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The number of evens is: " << x  << endl;
}
```

Answer:

```
int numEven(int d[][3], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         if ((d[i][j] % 2) == 0) count++;
      }
   return count;
}
```

Problem 139 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   x = diffTwo(b[0], y);                    // (a) sets x to approx difference 1
   swap(x, d[1][1]);                        // (b) swaps x with value of d[1][1]
   cout << biggest(d, 2, 2);                // (c) prints biggest row: 3 4
   printTwo(b);                             // (d) prints two entries: 1.9 2.3
   cout << summit(b[2], d[0][0]) << endl;   // (e) prints the sum 4
   return 0;
}
```

(a) Title line for **diffTwo**.

Answer:

```
int diffTwo(double z, int y)
```

(b) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(c) Title line for **biggest**.

Answer:

```
string biggest(int d[][2], int r, int c)
```

(d) Title line for **printTwo**.

Answer:

```
void printTwo(double b[])
```

(e) Title line for **summit**.

Answer:

```
double summit(double x, int y)
```

Problem 140 Consider the following C++ program.

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

string fun(int x) {
   string ans = "0123456789";
   if (x <= 0) return "4";
   if ((x >= 30) && (x < 1000)) return ans.substr(x % 7);
   if ((x >= 0) || (x < 100)) return "x11";
   return ans;
}

int up(int &x) {
  x--;
```

```
  cout << x << endl;
  return x - 1;
}

int main() {
    int x = 5;
    cout << fun(0) << endl;    // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << fun(3003) << endl; // line (c)
    up(x);                     // line (d)
    cout << up(x) << endl;     // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
4
```

(b) What is the output at line (b)?

Answer:

```
56789
```

(c) What is the output at line (c)?

Answer:

```
x11
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
3
2
```

Problem 141 The following C++ program has errors at the lines marked a,b,c,d, and e. For each answer write a single line of C++ that fixes the errors in the corresponding line.

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

void main(double x, string s[]) {    // line a

    ofstream f;
    f.open("outputFile");
    if (f == 0) return f;            // line b

    while (1 = 1) {                  // line c

       x -- 1;                       // line d

       if (x < 0) return 0;
```

```
       cout <<  s[x] endl;           // line e

    }

    return 0;
}
```

(a) Correct line (a):

Answer:

```
int main(int x, char  *y[]) {        // line a
```

(b) Correct line (b):

Answer:

```
    if (f == 0) return 0;            // line b
```

(c) Correct line (c):

Answer:

```
    while (1 == 1) {                 // line c
```

(d) Correct line (d):

Answer:

```
       x -= 1;                       // line d
```

(e) Correct line (e):

Answer:

```
       cout <<  y[x] << endl;        // line e
```

Problem 142 Write a function called *rowSum* that calculates and returns the sum of the entries of a specified row of a 2-dimensional array.

For example, a program that uses the function *rowSum* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << rowSum (x, 2, 3, 1) << endl;
     // from the 2-d array x that has size 2 x 3, find the sum of row 1
     // output will be 15 since row #1 contains 1, 5 and 9.
   return 0;
}
```

Answer:

```
int rowSum(int x[][3], int r, int c, int row) {
   int ans = 0;
   for (int j = 0; j < c; j++) ans += x[row][j];
   return ans;
}
```

Problem 143 Write a function called *numEven* that the returns the number of digits in a positive integer parameter that are even.

For example, a program that uses the function *numEven* follows.

```
int main() {
   cout << numEven(444) << endl;        // prints 3
   cout << numEven(414) << endl;        // prints 2
   cout << numEven(91) << endl;         // prints 0
   return 0;
}
```

Answer:

```
int numEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return numEven(x / 10) + 1;
   return numEven(x / 10);
}
```

Problem 144 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an odd integer n that is between 1 and 25.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n characters in the first row). Along each row the characters to be used is the sequence of uppercase letters A, B, C, ..., and so on.

Here is an example of how the program should work:

```
Give me an odd integer between 1 and 25:   7
ABCDEFG
 ABCDE
  ABC
   A
```

Answer:

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

int main() {
   int n;
   cout << "Give me an odd integer between 1 and 25:";
   cin >> n;

   while (n < 1 || n > 25 || (n % 2) != 1) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }

   int mid = n / 2;

   for (int r = mid; r >= 0; r--) {
      char out = 'A';
      for (int c = 0; c < n; c++)
         if ((c >= mid - r) && (c <= mid + r)) {
            cout << out;
            out++;
         }
         else cout << " ";
      cout << endl;
   }
}
```

Problem 145 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << firstDigit(65536) << endl;     // prints: Six
   cout << undouble(11223344);            // prints: 1234
   cout << halfString(a[0]) << endl;      // prints: Fre
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

```
string firstDigit(int x)
```

(b) Title line for **undouble**.

Answer:

```
int undouble(int x)
```

(c) Title line for **halfString**.

Answer:

```
string halfString(string x)
```

(d) Title line for **sort**.

Answer:

```
void sort(string a[], int capacity)
```

(e) Title line for **randomWord**.

Answer:

```
string randomWord()
```

Problem 146 Consider the following C++ program.

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

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
2
```

(b) line b:

```
32
```

(c) line c:

```
12
```

(d) line d:

```
24
```

(e) line e:

```
43
12
```

Problem 147 Write a function called *twos* that deletes all digits that are not equal to 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

```
int twos(int x) {
   if (x == 0) return 0;
   if (x % 10 != 2) return twos(x / 10);
   return 10*twos(x / 10) + 2;
}
```

Problem 148 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = range (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

```
int range(int d[][3], int r, int c) {
   int max = d[0][0];
   int min = d[0][0];
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
          if (d[i][j] < min) min = d[i][j];
          if (d[i][j] > max) max = d[i][j];
      }
   return max - min;
}
```

Problem 149 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[5] = {1.9, 2.3, 3.0};
   int d[2][2] = {{1,2},{3,4}};

   x = subtract(z, y);                  // (a) sets x to difference 1
   reset(x, d[1][1]);                   // (b) replaces x by value of d[1][1]
   bigRow(d, 2, 2);                     // (c) prints biggest row: 3 4
   cout << printAll(b, 3) << endl;      // (d) prints array: 1.9 2.3 3.0
   cout << add(b[2], d[0][0]) << endl;  // (e) prints the sum 4
   return 0;
}
```

(a) Title line for **subtract**.

Answer:

```
int subtract(int z, int y)
```

(b) Title line for **reset**.

Answer:

```
void reset(int &x, int y)
```

(c) Title line for **bigRow**.

Answer:

```
void bigRow(int d[][2], int r, int c)
```

(d) Title line for **printAll**.

Answer:

```
string printAll(double b[], int cap)
```

(e) Title line for **add**.

Answer:

```
double add(double x, int y)
```

Problem 150 Consider the following C++ program.

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

string fun(int x) {
   string ans = "0123456789";
   if (x <= 0) return "0";
   if ((x >= 30) && (x < 1000)) return ans.substr(x % 10);
   if ((x >= 0) || (x < 100)) return "x+1";
   return ans + ans;
}

int up(int &x) {
  x++;
  cout << x << endl;
  return x;
}

int main() {
    int x = 4;
    cout << fun(0) << endl;    // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << fun(3003) << endl; // line (c)
    up(x);                     // line (d)
    cout << up(x) << endl;     // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
0
```

(b) What is the output at line (b)?

Answer:

```
3456789
```

(c) What is the output at line (c)?

Answer:

```
x+1
```

(d) What is the output at line (d)?

Answer:

```
5
```

(e) What is the output at line (e)?

Answer:

```
6
6
```

Problem 151 The following C++ program has errors at the lines marked a,b,c,d, and e. For each answer write a single line of C++ that fixes the errors in the corresponding line.

```
#include <iostream>
#include <fstream>
```

```
using namespace std;

int main(int x, string y[]) {        // line a

    while (0 < x < 5) {              // line b

       cout >> y[x - 1] >> end;      // line c

       x =- 1;                       // line d

    }
    ifstream f;
    f.open("inputFile");
    if (f = 0) return -1;            // line e

    return 0;
}
```

(a) Correct line (a):

Answer:

```
int main(int x, char  *y[]) {        // line a
```

(b) Correct line (b):

Answer:

```
    while (0 < x && x < 5) {         // line b
```

(c) Correct line (c):

Answer:

```
       cout << y[x - 1] << endl;     // line c
```

(d) Correct line (d):

Answer:

```
       x -= 1;                       // line d
```

(e) Correct line (e):

Answer:

```
    if (f == 0) return -1;           // line e
```

Problem 152 Write a function called *colSum* that calculates and returns the sum of the entries of a specified column in a 2-dimensional array.

For example, a program that uses the function *colSum* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << colSum (x, 2, 3, 1) << endl;
     // from the 2-d array x that has size 2 x 3, find the sum of column 1
     // output will be 6 since col #1 contains 1 and 5.
   return 0;
}
```

Answer:

```
int colSum(int x[][3], int r, int c, int col) {
   int ans = 0;
   for (int i = 0; i < r; i++) ans += x[i][col];
   return ans;
}
```

Problem 153 Write a function called *num4* that the returns the number of digits in a positive integer parameter that are equal to 4.

For example, a program that uses the function *num4* follows.

```
int main() {
   cout << num4(444) << endl;        // prints 3
   cout << num4(414) << endl;        // prints 2
   cout << num4(81) << endl;         // prints 0
   return 0;
}
```

Answer:

```
int num4(int x) {
   if (x <= 0) return 0;
   if (x % 10 == 4) return num4(x / 10) + 1;
   return num4(x / 10);
}
```

Problem 154 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an odd integer n that is between 1 and 21.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n characters in the last row). Along each row the characters to be used is the sequence of uppercase letters A, B, C, ..., and so on.

Here is an example of how the program should work:

```
Give me an odd integer between 1 and 21:   7
   A
  ABC
 ABCDE
ABCDEFG
```

Answer:

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

int main() {
   int n;
   cout << "Give me an odd integer between 1 and 21:";
   cin >> n;

   while (n < 1 || n > 21 || (n % 2) != 1) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }

   int mid = n / 2;
```

```
   for (int r = 0; r <= mid; r++) {
      char out = 'A';
      for (int c = 0; c < n; c++)
         if ((c >= mid - r) && (c <= mid + r)) {
            cout << out;
            out++;
         }
         else cout << " ";
      cout << endl;
   }
}
```

Problem 155 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,5}};
   string s= "Hello";
   string t;

   cout << average(a, 5) << endl;           // prints the average:  2.8
   t = reverse(s);  cout << t << endl;      // prints:  olleH
   reverseRows(x, 2, 3);                    // prints:  2 4 5, 0 1 3
   if (hasRepeat(a, 5)) cout << "Has repeat" << endl;
                                            // prints:  Has repeat
   t = entries(a, 5); cout << t << endl;    // prints:  3,1,4,1,5
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int a[], int cap)
```

(b) Title line for **reverse**

Answer:

```
string reverse(string s)
```

(c) Title line for **reverseRows**

Answer:

```
void reverseRows(int x[][3], int r, int c)
```

(d) Title line for **hasRepeat**

Answer:

```
bool hasRepeat(int a[], int cap)
```

(e) Title line for **entries**

Answer:

```
string entries(int a[], int cap)
```

Problem 156 Consider the following C++ program.

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

char f(string s, int n) {
   if (n >= s.length()) return 'A';
   return s[n];
}

int mystery (int x) {
   if (x > 5) return 0;
   cout << -x;
   return x;
}

int main () {
   cout << f("Hello", 20) << endl;     //line A
   cout << f("Hello", 1)  << endl;     //line B
   cout << mystery(19683) << endl;     //line C
   cout << mystery(2) << endl;         //line D
   mystery(-5);                        //line E
   cout << endl;
   return 0;
}
```

(a) What is the output at line A?

Answer:

```
A
```

(b) What is the output at line B?

Answer:

```
e
```

(c) What is the output at line C?

Answer:

```
0
```

(d) What is the output at line D?

Answer:

```
-22
```

(e) What is the output at line E?

Answer:

```
5
```

Problem 157 Write a function called *extraOne* that places an initial 1 at the start of an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *extraOne* follows.

```
int main() {
   int x = extraOne(729);
   cout << x << endl;    // prints   1729
   return 0;
}
```

Answer:

```
int extraOne(int x) {
   if (x < 10) return 10 + x;
   return 10 * extraOne(x / 10) + x % 10;
}
```

Problem 158 Write a function called *dropDimension* that copies the entries from a 2-dimensional array row by row as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,1,4}, {1,5,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

```
void dropDimension(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++) {
          x[i] = y[r][c];
          i++;
      }
}
```

Problem 159 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,8}};
   string s= "Hello";
   string t;

   cout << average(x, 2, 3) << endl;        // prints the average:  3.0
   t = doubleIt(s);  cout << t << endl;     // prints:  HelloHello
   reverseCols(x, 2, 3);                    // prints:  3 0 1, 8 4 2
   if (isPositive(a[0])) cout << "Positive" << endl;
                                            // prints:  Positive
   cout << midEntry(a, 5) << endl;          // prints:  4
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int x[][3], int r, int c)
```

(b) Title line for **doubleIt**
Answer:

```
string doubleIt(string s)
```

(c) Title line for **reverseCols**
Answer:

```
void reverseCols(int x[][3], int r, int c)
```

(d) Title line for **isPositive**
Answer:

```
bool isPositive(int x)
```

(e) Title line for **midEntry**
Answer:

```
int midEntry(int a[], int cap)
```

Problem 160 Consider the following C++ program.

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

string f(string s, int n) {
   if (n >= s.length()) return "XYZ";
   return s.substr(n);
}

int mystery (int x) {
   if (x > 5) return 0;
   return x;
}

int main () {
   cout << mystery(19683) << endl;     //line A
   cout << mystery(2) << endl;         //line B
   cout << f("Hello", 20) << endl;     //line C
   cout << f("Hello", 1)  << endl;     //line D
   mystery(-5);                        //line E
   return 0;
}
```

(a) What is the output at line A?
Answer:

```
0
```

(b) What is the output at line B?
Answer:

```
2
```

(c) What is the output at line C?
Answer:

```
XYZ
```

(d) What is the output at line D?

Answer:

```
ello
```

(e) What is the output at line E?

Answer:

Problem 161 Write a function called *doubleEight* that places an extra digit 8 after the last 8 in an integer parameter. If there is no 8 present, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleEight* follows.

```
int main() {
   int x = doubleEight(19683);
   cout << x << endl;                       // prints   196883
   cout << doubleEight(271828) << endl;     // prints   2718288
   cout << doubleEight(314159) << endl;     // prints   314159
   return 0;
}
```

Answer:

```
int doubleEight(int x) {
   if (x % 10 == 8) return 10 * x + 8;
   if (x < 10) return x;
   return 10 * doubleEight(x / 10) + x % 10;
}
```

Problem 162 Write a function called *dropDimension* that copies the entries from a 2-dimensional array column by column as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,4,5}, {1,1,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

```
void dropDimension(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int c = 0; c < cols; c++)
      for (int r = 0; r < rows; r++) {
          x[i] = y[r][c];
          i++;
      }
}
```

Problem 163 Write a function called *extraTwo* that inserts an extra digit 2 as the second digit of an integer parameter. (Assume that the input parameter is positive.)

For example, a program that uses the function *extraTwo* follows.

```
int main() {
   int x = extraTwo(79);
   cout << x << endl;             // prints   729
   cout << extraTwo(1) << endl;   // prints   12
   return 0;
}
```

Answer:

```
int extraTwo(int x) {
   if (x < 10) return 10 * x + 2;
   return 10 * extraTwo(x / 10) + x % 10;
}
```

Problem 164 Write a function called *fill2D* that fills the entries of a 2-dimensional array column by column from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 345   is printed
     // 119
   return 0;
}
```

Answer:

```
void fill2D(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int c = 0; c < cols; c++)
      for (int r = 0; r < rows; r++) {
          y[r][c] = x[i];
          i++;
      }
}
```

Problem 165 Write a function called *doubleFour* that places an extra copy of the 4th digit right after that digit in an integer parameter. If there is no 4th digit, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleFour* follows.

```
int main() {
   int x = doubleFour(19683);
   cout << x << endl;             // prints    196883
```

```
   cout << doubleFour(271828);    // prints   2718828
   cout << doubleFour(314159);    // prints   3141159
   return 0;
}
```

Answer:

```
int doubleFour(int x) {
   if (x < 1000) return x;
   if (x < 10000) return 10 * x + x % 10;
   return 10 * doubleFour(x / 10) + x % 10;
}
```

Problem 166 Write a function called $fill2D$ that fills the entries of a 2-dimensional array row by row from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 314   is printed
     // 159
   return 0;
}
```

Answer: Answer:

```
void fill2D(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++) {
          y[r][c] = x[i];
          i++;
      }
}
```

Problem 167 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double b[5] = {1.9, 2.3, 3.0, 4.4, 5.7};
   double d = 3.1415926;
   int x = 2;
   cout << decimalPart(b[1]) << endl;   // (a) prints 0.3
   medianPosition(b, 5);                // (b) prints 2, the index of the median
   swap1(d, b[1]);                      // (c) swaps b[1] with d
   swap2(b, 3, x);                      // (d) swaps entry b[3] with b[x]
   cout << sqrt(d) << endl;             // (e) prints the square root of d
   return 0;
}
```

(a) Title line for **decimalPart** as called at the line marked (a).
Answer:

```
double decimalPart(double x)
```

(b) Title line for **medianPosition** as called at the line marked (b).
Answer:

```
void medianPosition(double x[], int cap)
```

(c) Title line for **swap1** as called at the line marked (c).
Answer:

```
void swap1(double &x, double &y)
```

(d) Title line for **swap2** as called at the line marked (d).
Answer:

```
void swap2(double x[], int y, int z)
```

(e) Title line for **sqrt** as called at the line marked (e).
Answer:

```
double sqrt(double x)
```

Problem 168 Consider the following C++ program.

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

string fun(int x) {
   if (x <= 0) return "";
   if (x >= 9 && x % 2 == 1) return "x+1";
   if (x >= 9 || x % 3 == 0) return "x+2";
   return "5";
}

int rec(int x) {
  if (x < 100) return x/5;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-3) << endl;   // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << rec(36) << endl;   // line (c)
    cout << rec(-555) << endl; // line (d)
    cout << rec(987) << endl;  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

```
x+1
```

(c) What is the output at line (c)?

Answer:

```
7
```

(d) What is the output at line (d)?

Answer:

```
-111
```

(e) What is the output at line (e)?

Answer:

```
36
```

Problem 169 Write a function called *dropEvens* that forms a new number from a positive integer parameter by dropping all even digits. In case all digits are even or a negative parameter is given an answer of 0 is to be returned.

For example, a program that uses the function *dropEvens* follows.

```
int main() {
   cout << dropEvens(1245);  // prints 15
   cout << dropEvens(19683); // prints 193
   cout << dropEvens(0);     // prints 0
   cout << dropEvens(-10);   // prints 0
   return 0;
}
```

Answer:

```
int dropEvens(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return dropEvens(x/10);
   return 10 * dropEvens(x/10) + x % 10;
}
```

Problem 170 Write a function called *randChange* that selects one entry at random in an array of integers and changes it to a random negative integer that lies between -99 and -1 inclusive. (You must use an appropriate standard C++ function to generate all random numbers.)

For example, a program that uses the function *randChange* follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   randChange(x, 6);
   for (int i = 0; i <= 5; i++)
      cout << x[i] << " ";     // might print  3 1 -17 1 5 9
   cout << endl;
   return 0;
}
```

Answer:

```
void randChange(int x[], int cap) {
   int r = rand() % cap;
   x[r] = -(1 + rand() % 99);
}
```

Problem 171 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out file1 file2 file3
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
char a = 'b';
cout << a << endl;
```

Answer:

```
b
```

(ii)

```
char a = 'b';
while (a <= 'f') {
   cout << a - 'a';
   a = a + 1;
}
```

Answer:

```
12345
```

(iii)

```
int main(int argc, char *argv[]) {
   cout << argv[1];
```

Answer:

```
file1
```

(iv)

```
string x = "Easy Question";
cout << x.substr(1,2);
```

Answer:

```
as
```

(v)

```
string x = "Easy Question";
cout << x.rfind("E");
```

Answer:

```
0
```

Problem 172 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 20.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a square picture (as shown in the diagram, but with n rows) that uses the uppercase letters A, B, C, ... in sequence, to form an outer stepeter of As that contains a stepeter of Bs, that contains a permimeter of Cs, and so on.

Here is an example of how the program should work:

```
Give me an integer between 1 and 20:   7
AAAAAAA
ABBBBBA
ABCCCBA
ABCDCBA
ABCCCBA
ABBBBBA
AAAAAAA
```

Answer:

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

int main() {
   char picture[20][20];

   int n = 0;
   while (n < 1 || n > 20) {
      cout << "Give me an integer between 1 and 20:   ";
      cin >> n;
   }
   int mid = (n + 1) / 2;

   for (int step = 0; step <  mid; step++) {
      char x = 'A' + step;
      for (int r = step; r < n -step; r++)
       for (int c = step; c < n -step; c++)
          picture[r][c] = x;
   }

   for (int r = 0; r < n; r++) {
     for (int c = 0; c < n; c++)
        cout << picture[r][c];
     cout << endl;
   }
   return 0;
}
```

Problem 173 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   bool b[5] = {true, true, false, true, true};
   int x = 2;
   cout << isTrue(b[1 + 2]) << endl;   // (a) prints true
   allTrue(b, 5);                      // (b) prints False
   swap1(b, 3, x);                     // (d) swaps entry b[3] with b[x]
   swap2(b[x], b[x+1]);                // (d) swaps entries
   cout << sqrt(x) << endl;            // (e) prints the square root of x
   return 0;
}
```

(a) Title line for **isTrue** as called at the line marked (a).

Answer:

```
bool isTrue(bool x)
```

(b) Title line for **allTrue** as called at the line marked (b).
Answer:

```
void allTrue(bool x[], int cap)
```

(c) Title line for **swap1** as called at the line marked (c).
Answer:

```
void swap1(bool x[], int y, int z)
```

(d) Title line for **swap2** as called at the line marked (d).
Answer:

```
void swap2(bool &x, bool &y)
```

(e) Title line for **sqrt** as called at the line marked (e).
Answer:

```
double sqrt(int x)
```

Problem 174 Consider the following C++ program.

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

double fun(int x) {
   if (x <= 0) return sqrt((double) (-x));
   if (x >= 9 && x % 2 == 1) return x+1.0;
   if (x >= 9 || x % 3 == 0) return x+2.0;
   return 3.0;
}

int rec(int x) {
  if (x < 100) return x/3;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-3) << endl;   // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << rec(36) << endl;   // line (c)
    cout << rec(-555) << endl; // line (d)
    cout << rec(987) << endl;  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
1.73205
```

(b) What is the output at line (b)?
Answer:

```
34
```

(c) What is the output at line (c)?
Answer:

```
12
```

(d) What is the output at line (d)?

Answer:

```
-185
```

(e) What is the output at line (e)?

Answer:

```
61
```

Problem 175 Write a function called *onlyEvens* that forms a new number from a positive integer parameter by dropping all odd digits. In case all digits are odd or a negative parameter is given an answer of 0 is to be returned.

For example, a program that uses the function *onlyEvens* follows.

```
int main() {
   cout << onlyEvens(1245);  // prints 24
   cout << onlyEvens(19683); // prints 68
   cout << onlyEvens(0);     // prints 0
   cout << onlyEvens(-10);   // prints 0
   return 0;
}
```

Answer:

```
int onlyEvens(int x) {
   if (x <= 0) return 0;
   if (x % 2 != 0) return onlyEvens(x/10);
   return 10*onlyEvens(x/10) + x% 10;
}
```

Problem 176 Write a function called *randChange* that selects one entry at random in a 2-dimensional array of integers and changes it to -17. (You must use an appropriate standard C++ function to generate all random numbers.)

For example, a program that uses the function *randChange* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   randChange(x, 2, 3);
   for (int i = 0; i <= 1; i++)
    for (int j = 0; j <= 2; j++)
      cout << x[i][j] << " ";     // might print  3 1 -17 1 5 9
   cout << endl;
   return 0;
}
```

Answer:

```
void randChange(int x[][3], int rows, int cols) {
   int r = rand()%rows;
   int c = rand()%cols;
    x[r][c] = -17;
}
```

Problem 177 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out file1 file2 file3
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
char a = 'a';
cout << a << endl;
```

Answer:

```
a
```

(ii)

```
char a = 'a';
while (a <= 'f') {
   cout << 'a' - a;
   a = a + 1;
}
```

Answer:

```
0-1-2-3-4-5
```

(iii)

```
int main(int argc, char *argv[]) {
   cout << argc;
```

Answer:

```
4
```

(iv)

```
string x = "Easy Question";
cout << x.substr(6, 0);
```

Answer:

(v)

```
string x = "Easy Question";
cout << x.rfind("s");
```

Answer:

```
8
```

Problem 178 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 20.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a square picture (as shown in the diagram, but with n rows) that uses the uppercase letters O and X in sequence, to form an outer perimeter of Os that contains a perimeter of Xs, that contains a permimeter of Os, and so on.

Here is an example of how the program should work:

```
Give me an integer between 1 and 20:   7
OOOOOOO
OXXXXXO
OXOOOXO
OXOXOXO
OXOOOXO
OXXXXXO
OOOOOOO
```

Answer:

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

int main() {
   char picture[20][20];

   int n = 0;
   while (n < 1 || n > 20) {
      cout << "Give me an integer between 1 and 20:   ";
      cin >> n;
   }
   int mid = (n + 1) / 2;

   for (int perim = 0; perim <  mid; perim++) {
      char x = 'O';
      if (perim % 2 == 1) x = 'X';
      for (int r = perim; r < n -perim; r++)
       for (int c = perim; c < n -perim; c++)
          picture[r][c] = x;
   }

   for (int r = 0; r < n; r++) {
     for (int c = 0; c < n; c++)
        cout << picture[r][c];
     cout << endl;
   }
   return 0;
}
```

Problem 179 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string b[5] = {"1.9", "2.3", "3.0", "4.4", "5.7"};
   double d = 3.1415926;
   int x = 2;
   cout << decimalPart(b[1]) << endl;   // (a) prints 0.3
   medianPosition(b, 5);                // (b) prints 2, the index of the median
   swap1(d, b[1]);                      // (c) changes b[1] and d
   swap2(b, 3, x);                      // (d) swaps entry b[3] with b[x]
   cout << sqrt(d) << endl;             // (e) prints the square root of d
   return 0;
}
```

(a) Title line for **decimalPart** as called at the line marked (a).

Answer:

```
double decimalPart(string s)
```

(b) Title line for **medianPosition** as called at the line marked (b).
Answer:

```
void medianPosition(string x[], int cap)
```

(c) Title line for **swap1** as called at the line marked (c).
Answer:

```
void swap1(double &d, string &s)
```

(d) Title line for **swap2** as called at the line marked (d).
Answer:

```
void swap2(string x[], int i, int j)
```

(e) Title line for **sqrt** as called at the line marked (e).
Answer:

```
double sqrt(double d)
```

Problem 180 Consider the following C++ program.

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

string fun(char x) {
   if (x <= 'k') return "";
   if (x >= 'l' && x <= 't') return "x++";
   if (x >= 'p') return "x-1";
   return "20";
}

int rec(int x) {
  if (x < 1000) return x/5;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun('m') << endl;   // line (a)
    cout << fun('p') << endl;   // line (b)
    cout << rec(666) << endl;   // line (c)
    cout << rec(-555) << endl; // line (d)
    cout << rec(2013) << endl;  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
x++
```

(b) What is the output at line (b)?
Answer:

```
x++
```

(c) What is the output at line (c)?

Answer:

```
133
```

(d) What is the output at line (d)?

Answer:

```
-111
```

(e) What is the output at line (e)?

Answer:

```
42
```

Problem 181 Write a function called *upEvens* that forms a new number from a non-negative integer parameter by increasing all even digits. In case a negative parameter is given an answer of 0 is to be returned.

For example, a program that uses the function *upEvens* follows.

```
int main() {
   cout << upEvens(1245);  // prints 1355
   cout << upEvens(19683); // prints 19793
   cout << upEvens(0);     // prints 1
   cout << upEvens(-10);   // prints 0
   return 0;
}
```

Answer:

```
int upEvens(int x) {
   if (x < 0) return 0;
   if (x < 10)
      if (x % 2 == 0) return x + 1;
      else return x;
   return 10*upEvens(x/10) + upEvens(x%10);
}
```

Problem 182 Write a function called *randSelect* that selects one row at random in a 2-dimensional array of integers and returns the sum of the entries in that row. (You must use an appropriate standard C++ function to generate all random numbers.)

For example, a program that uses the function *randSelect* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << randSelect(x, 2, 3);  // might print 8 if the first row is selected
   cout << endl;
   return 0;
}
```

Answer:

```
int randSelect(int x[][3], int rows, int cols) {
   int r = rand() % rows;
   int ans = 0;
   for (int c = 0; c < cols; c++)
      ans += x[r][c];
   return ans;
}
```

Problem 183 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out file1 file2 file3
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
char a = 'a';
cout << (char) (a + 2) << endl;
```

Answer:

```
c
```

(ii)

```
char a = 'b';
while ((a - 'a') <= 5) {
   cout << a;
   a = a + 1;
}
```

Answer:

```
bcdef
```

(iii)

```
int main(int argc, char *argv[]) {
   cout << argv[2];
```

Answer:

```
file2
```

(iv)

```
string x = "Easy Question";
cout << x.substr(3,2);
```

Answer:

```
y
```

(v)

```
string x = "Easy Question";
cout << x.rfind("e");
```

Answer:

```
7
```

Problem 184 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 20.

2. It exits if the user enters an illegal value for n.

3. It prints out a triangular picture (as shown in the diagram, but with n rows) that uses the uppercase letters A, B, C, ... in sequence, to form the diagonal sides of the triangle. The vertical straight side should be at the right.

Here is an example of how the program should work:

```
Give me an integer between 1 and 20:   7
      A
     AB
    ABC
   ABCD
  ABCDE
 ABCDEF
ABCDEFG
```

Answer:

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

int main() {
   int n = 0;
   cout << "Give me an integer between 1 and 20:  ";
   cin >> n;
   if (n < 1 || n > 20) return 0;

   for (int r = n; r >= 1; r--) {
      char x = 'A';
      for (int c = 1; c <= n; c++) {
         if (c < r) cout << " ";
         else {
            cout << x;
            x++;
         }
      }
     cout << endl;
   }
   return 0;
}
```

Problem 185 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   char b[5] = {'t', 't', 'f', 't', 't'};
   int x = 2;
   cout << isT(b[1 + 2]) << endl;      // (a) prints true
   allTrue(b, 5);                      // (b) prints false
   swap1(b, 3, x);                     // (d) swaps entry b[3] with b[x]
   swap2(b[x], b[x+1]);                // (d) swaps entries
   cout << sqrt(x) << endl;            // (e) prints the square root of x
   return 0;
}
```

(a) Title line for **isT** as called at the line marked (a).

Answer:

```
bool isT(char x)
```

(b) Title line for **allTrue** as called at the line marked (b).

Answer:

```
void allTrue(char b[], int cap)
```

(c) Title line for **swap1** as called at the line marked (c).
Answer:

```
void swap1(char b[], int i, int j)
```

(d) Title line for **swap2** as called at the line marked (d).
Answer:

```
void swap2(char &x, char &y)
```

(e) Title line for **sqrt** as called at the line marked (e).
Answer:

```
double sqrt(int x)
```

Problem 186 Consider the following C++ program.

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

double fun(double x) {
   if (x <= 0.0) return sqrt(-x);
   if (x >= 9.0 && x <= 100.0) return x+1.0;
   if (x >= 90.0 || x >= 5.0) return x+2.0;
   return 3.0;
}

int rec(int x) {
  if (x < 100) return x/6;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-4.0) << endl;   // line (a)
    cout << fun(99.0) << endl;   // line (b)
    cout << fun(2.0)  << endl;   // line (c)
    cout << rec(-666) << endl;   // line (d)
    cout << rec(987) << endl;    // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
2
```

(b) What is the output at line (b)?
Answer:

```
100
```

(c) What is the output at line (c)?
Answer:

```
3
```

(d) What is the output at line (d)?
Answer:

```
-111
```

(e) What is the output at line (e)?

Answer:

```
30
```

Problem 187 Write a function called *downOdds* that forms a new number from a non-negative integer parameter by decreasing all odd digits. In case a negative parameter is given an answer of 0 is to be returned.

For example, a program that uses the function *downOdds* follows.

```
int main() {
   cout << downOdds(3245);  // prints 2244
   cout << downOdds(19683); // prints 8682
   cout << downOdds(1);     // prints 0
   cout << downOdds(-10);   // prints 0
   return 0;
}
```

Answer:

```
int downOdds(int x) {
   if (x <= 0) return 0;
   if (x < 10)
      if (x % 2 == 1) return x - 1;
      else return x;
   return 10*downOdds(x/10) + downOdds(x % 10);
}
```

Problem 188 Write a function called *randSelect* that selects one column at random in a 2-dimensional array of integers and returns the product of the entries in that row. (You must use an appropriate standard C++ function to generate all random numbers.)

For example, a program that uses the function *randSelect* follows.

```
int main() {
   int x[2][3] = {{3, 1, 4}, {1, 5, 9}};
   cout << randSelect(x, 2, 3);  // might print 36 if the last col is selected
   cout << endl;
   return 0;
}
```

Answer:

```
int randSelect(int x[][3], int rows, int cols) {
   int c = rand() % cols;
   int ans = 1;
   for (int r = 0; r < rows; r++)
      ans *= x[r][c];
   return ans;
}
```

Problem 189 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out file1 file2 file3
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
char c = 'a';
cout << (char) (c + 3) << endl;
```

Answer:

```
d
```

(ii)

```
char a = 'a';
while (('a' - a) <= 3) {
   cout << 'a';
   a = a - 1;
}
```

Answer:

```
aaaa
```

(iii)

```
int main(int argc, char *argv[]) {
   cout << argv[argc - 1];
```

Answer:

```
file3
```

(iv)

```
string x = "Easy Question";
cout << x.length();
```

Answer:

```
13
```

(v)

```
string x = "Easy Question";
cout << x.find("e");
```

Answer:

```
7
```

Problem 190 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 25.

2. It exits if the user enters an illegal value for n.

3. It prints out a downward pointing triangular picture (as shown in the diagram, but with n rows) that uses the lowercase letters a, b, c, ... in sequence, to form the diagonal sides of the triangle.

Here is an example of how the program should work:

```
Give me an integer between 1 and 25:   7
abcdefg
 abcdef
  abcde
   abcd
    abc
     ab
      a
```

Answer:

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

int main() {
   int n = 0;
   cout << "Give me an integer between 1 and 25:  ";
   cin >> n;
   if (n < 1 || n > 20) return 0;

   for (int r = n; r >= 1; r--) {
      for (int c = 1; c <= n; c++) {
         if (c < r) cout << " ";
         else cout << (char) ('a' + c - r);
      }
     cout << endl;
   }
   return 0;
}
```

Problem 191 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << min(i, j) << endl;              // prints minimum
   printArray(x, 3, 2);                    // prints array
   cout << average(a, 9) << endl;          // prints average
   swap(a, 3, 5);                          // swap elements 3 and 5
   reverse(a[1]);                          // reverse the digits in a[1]
   return 0;
}
```

(a) Title line for **min**

Answer:

```
int min (int i, int j) {
```

(b) Title line for **printArray**

Answer:

```
void printArray(int a[][2], int rows, int cols)
```

(c) Title line for **average**

Answer:

```
double average(int a[], int cap)
```

(d) Title line for **swap**

Answer:

```
void swap(int a[], int i, int j )
```

(e) Title line for **reverse**

Answer:

```
void reverse(int &a)
```

Problem 192 Consider the following C++ program.

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

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 10 * (n % 10);
}

int mystery (int x) {
   cout << x << "54321";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

```
7
```

(b) What is the output at line B?

Answer:

```
150
```

(c) What is the output at line C?

Answer:

```
16030
```

(d) What is the output at line D?

Answer:

```
2543212
```

(e) What is the output at line E?

Answer:

254321

Problem 193 Write a function called *smallestDigit* that finds the smallest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *smallestDigit* follows.

```
int main() {
   cout << smallestDigit(29) << endl;    // prints    2
   cout << smallestDigit(31415) << endl; // prints    1
   cout << smallestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

```
int smallestDigit(int x) {
   if (x < 10) return x;
   int ans = smallestDigit(x/10);
   if (ans < x % 10) return ans;
   return x % 10;
}
```

Problem 194 Write a function called *lastIndex* that finds the largest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << lastIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << lastIndex(x, capacity, 1) << endl;
     // prints  3  because the target 1 is last found as element number 3
   cout << lastIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

```
int lastIndex(int a[], int capacity, int target) {
   for (int i = capacity - 1; i >= 0; i--)
      if (a[i] == target) return i;
   return -1;
}
```

Problem 195 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << average(i, j) << endl;          // prints average
   printArray(a, 9);                       // prints array
   cout << min(x, 3, 2) << endl;           // prints minimal element
   reverse(a, 9);                          // reverse the order of elements
   swap(a[1], a[2]);                       // swap two values
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int i, int j)
```

(b) Title line for **printArray**

Answer:

```
void printArray(int a[], int cap)
```

(c) Title line for **min**

Answer: int min(int a[][2], int rows, int cols)

(d) Title line for **reverse**

Answer:

```
void reverse(int a[], int cap)
```

(e) Title line for **swap**

Answer:

```
void swap(int &i, int &j)
```

Problem 196 Consider the following C++ program.

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

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 11 * (n % 10);
}

int mystery (int x) {
   cout << x << "12345";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
```

```
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

```
7
```

(b) What is the output at line B?

Answer:

```
155
```

(c) What is the output at line C?

Answer:

```
16633
```

(d) What is the output at line D?

Answer:

```
2123452
```

(e) What is the output at line E?

Answer:

```
212345
```

Problem 197 Write a function called *biggestDigit* that finds the biggest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *biggestDigit* follows.

```
int main() {
   cout << biggestDigit(29) << endl;    // prints    9
   cout << biggestDigit(31415) << endl; // prints    5
   cout << biggestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

```
int biggestDigit(int x) {
   if (x < 10) return x;
   int ans = biggestDigit(x/10);
   if (ans > x % 10) return ans;
   return x % 10;
}
```

Problem 198 Write a function called $firstIndex$ that finds the smallest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << firstIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << firstIndex(x, capacity, 1) << endl;
     // prints  1  because the target 1 is first found as element number 1
   cout << firstIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

```
int firstIndex(int a[], int capacity, int target) {
   for (int i = 0; i < capacity; i++)
      if (a[i] == target) return i;
   return -1;
}
```

Problem 199 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int b[5] = {9, 3, 0, 4, 7};
   int x = 17;
   cout << decimalPart(3.14159) << endl;   // (a) prints 0.14159
   median(b, 5);                    // (b) prints 4, the median entry
   swap1(x, b[1]);                  // (c) swaps b[1] with x
   swap2(b, 3, 4);                  // (d) swaps entry b[3] with b[4]
   cout << sqrt(5, 10, 12) << endl; // (e) prints "Hello" for any input values
   return 0;
}
```

(a) Title line for **decimalPart** as called at the line marked (a).

Answer:

```
double decimalPart(double x)
```

(b) Title line for **median** as called at the line marked (b).

Answer:

```
void median(int a[], int cap)
```

(c) Title line for **swap1** as called at the line marked (c).

Answer:

```
void swap1(int &x, int &y)
```

(d) Title line for **swap2** as called at the line marked (d).

Answer:

```
void swap2(int x[], int a, int b)
```

(e) Title line for **sqrt** as called at the line marked (e).

Answer:

```
string sqrt(int a, int b, int c)
```

Problem 200 Consider the following C++ program.

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

int fun(int x) {
   if (x <= 0) return 10;
   if (x >= 9 && x % 2 == 1) return x + 1;
   if (x >= 9 || x % 3 == 0) return x + 2;
   return 5;
}

int rec(int x) {
  if (x < 100) return x/10;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-3) << endl;   // line (a)
    cout << fun(33) << endl;   // line (b)
    cout << rec(36) << endl;   // line (c)
    cout << rec(-666) << endl; // line (d)
    cout << rec(987) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
10
```

(b) What is the output at line (b)?

Answer:

```
34
```

(c) What is the output at line (c)?

Answer:

```
3
```

(d) What is the output at line (d)?

Answer:

```
-66
```

(e) What is the output at line (e)?

Answer:

```
17
```

Problem 201 Write a function called *multiDigit* that prints a new number formed from a positive integer parameter by printing each odd digit once and each even digit twice. If a negative parameter is given, it should print the word *Idiot* and if 0 is entered it should do nothing.

For example, a program that uses the function *multiDigit* follows.

```
int main() {
   multiDigit(1245); cout << endl;  // prints 122445
   multiDigit(19683); cout << endl; // prints 1966883
   multiDigit(0); cout << endl;     // prints
   multiDigit(-10); cout << endl;   // prints Idiot
   return 0;
}
```

Answer:

```
void multiDigit(int n) {
   if (n < 0) cout << "Idiot";
   else if (n == 0) return;
   else {
      multiDigit(n / 10);
      if (n % 2 == 0) cout << n % 10;
      cout << n % 10;
   }
}
```

Problem 202 Write a function called *randFill* that fills the entries of an array with random negative integers that lie between -99 and -1 inclusive. (Use an appropriate C++ function to generate the random numbers.)

For example, a program that uses the function follows.

```
int main() {
   int x[4];
   randFill(x, 4);
   for (int i = 0; i <= 3; i++)
      cout << x[i] << endl;     // prints 4 random negative numbers
   return 0;
}
```

Answer:

```
void randFill(int a[], int c) {
   for (int i = 0; i < c; i++)
      a[i] = rand() % 99 - 99;
}
```

Problem 203 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   int x = 4, y = 10;
   cout << (x/y + 1.0) << endl;
```

Answer:

```
1.0
```

(ii)

```
   char x = 'a';
   while (x <= 'f') {
      cout << (char) (x + 1);
      x = x + 1;
   }
```

Answer:

```
bcdefg
```

(iii)

```
   cout << 'a' - 'd';
```

Answer:

```
-3
```

(iv)

```
   string x = "Easy Question";
   cout << x.substr(1,2);
```

Answer:

```
as
```

(v)

```
int main(int argc, char *argv[]) {
   cout << argc;
```

Answer:

```
4
```

Problem 204 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 20.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n rows) that uses the uppercase letters A, B, C, ... in sequence, and if necessary returns to the letter A after any Z.

Here is an example of how the program should work:

```
Give me an integer between 1 and 20:   6
     A
    BC
   DEF
  GHIJ
 KLMNO
PQRSTU
```

Answer:

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

int main() {
   int n;
```

```
   cout << "Give me an integer between 1 and 20:  ";
   cin >> n;

   while (n < 1  || n > 20) {
      cout << "That is out of range. Give me an integer between 1 and 20:  ";
      cin >> n;
   }

   char x = 'A';
   for (int i = 1; i <= n; i++) {
       for (int j = 1; j <= n; j++)
          if ((i + j) <= n) cout << " ";
          else {
              cout << x;
              x = x + 1;
              if (x > 'Z') x = 'A';
          }
       cout << endl;
   }
}
```

Problem 205 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {9, 3, 0, 4, 7};
   int x = 17;
   cout << reducedFraction(2, 6) << endl;   // (a) prints 1/3
   swap1(a[1], a[2]);                   // (b) swaps a[1] with a[2]
   swap2(x, a, 3);                      // (c) swaps entry a[3] with x
   median(5, 4, 6);                     // (d) prints 5, the median entry
   cout << sqrt(5, 10, 12, 14) << endl; // (e) prints 25 for any input values
   return 0;
}
```

(a) Title line for **reducedFraction** as called at the line marked (a).

Answer:

```
string reducedFraction(int a, int b)
```

(b) Title line for **swap1** as called at the line marked (b).

Answer:

```
void swap1(int &x, int &y)
```

(c) Title line for **swap2** as called at the line marked (c).

Answer:

```
void swap2(int &x, int a[], int i)
```

(d) Title line for **median** as called at the line marked (d).

Answer:

```
void median(int a, int b, int c)
```

(e) Title line for **sqrt** as called at the line marked (e).

Answer:

```
int sqrt(int a, int b, int c, int d)
```

Problem 206 Consider the following C++ program.

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

int fun(int x) {
   if (x <= 0) return 10;
   if (x >= 9 && x % 2 == 1) return x + 1;
   if (x >= 9 || x % 3 == 0) return x + 2;
   return 5;
}

int rec(int x) {
  if (x < 100) return x/10;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-6) << endl;   // line (a)
    cout << fun(63) << endl;   // line (b)
    cout << rec(66) << endl;   // line (c)
    cout << rec(-747) << endl; // line (d)
    cout << rec(876) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
10
```

(b) What is the output at line (b)?

Answer:

```
64
```

(c) What is the output at line (c)?

Answer:

```
6
```

(d) What is the output at line (d)?

Answer:

```
-74
```

(e) What is the output at line (e)?

Answer:

```
15
```

Problem 207 Write a function called *multiDigit* that prints a new number formed from a positive integer parameter by printing each odd digit twice and each even digit once. If a negative parameter is given, it should print the word *Negative* and if 0 is entered it should do nothing.

For example, a program that uses the function *multiDigit* follows.

```
int main() {
   multiDigit(1245); cout << endl;  // prints 112455
   multiDigit(19683); cout << endl; // prints 11996833
   multiDigit(0); cout << endl;     // prints
   multiDigit(-10); cout << endl;   // prints Negative
   return 0;
}
```

Answer:

```
void multiDigit(int n) {
   if (n < 0) cout << "Negative";
   else if (n == 0) return;
   else {
      multiDigit(n / 10);
      if (n % 2 != 0) cout << n % 10;
      cout << n % 10;
   }
}
```

Problem 208 Write a function called *randFill* that fills the entries of an array with random integers between 1 and a specified maximum value. (Use an appropriate C++ function to generate the random numbers.)

For example, a program that uses the function follows.

```
int main() {
   int x[4];
   int max = 999;
   randFill(x, 4, max);
   for (int i = 0; i <= 3; i++)
      cout << x[i] << endl;     // prints 4 random numbers between 1 and 999
   return 0;
}
```

Answer:

```
void randFill(int a[], int c, int max) {
   for (int i = 0; i < c; i++)
      a[i] = rand() % max + 1;
}
```

Problem 209 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   int x = 8, y = 10;
   cout << ((x + 1.0)/y) << endl;
```

Answer:

```
0.9
```

(ii)

```
char x = 'f';
while (x <= 'a') {
   cout << (char) (x + 1);
   x = x + 1;
}
```

Answer:

(iii)

```
cout << 'e' - 'd';
```

Answer:

```
1
```

(iv)

```
string x = "Easy Question";
cout << x.substr(2,1);
```

Answer:

```
s
```

(v)

```
int main(int argc, char *argv[]) {
   cout << argv[2];
```

Answer:

```
input2
```

Problem 210 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 9.

2. It repeatedly reads n from the user until the supplied value of n is legal.

3. It prints out a triangular picture (as shown in the diagram, but with n rows) that uses the lowercase letters a, b, c, ... in sequence, and if necessary continues with uppercase letter starting at A after any z.

Here is an example of how the program should work:

```
Give me an integer between 1 and 9:    7
a
bc
def
ghij
klmno
pqrstu
vwxyzAB
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 9:  ";
   cin >> n;

   while (n < 1  || n > 9) {
      cout << "That is out of range. Give me an integer between 1 and 9:  ";
      cin >> n;
   }

   char x = 'a';
   for (int i = 1; i <= n; i++) {
       for (int j = 1; j <= i; j++) {
          cout << x;
          x = x + 1;
          if (x > 'z') x = 'A';
       }
       cout << endl;
   }
}
```

Problem 211 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int b[5] = {9, 3, 0, 4, 7};
   int x = 17;
   cout << integerPart(3.14159) << endl;   // (a) prints 3
   swap1(x, b[1]);                   // (b) swaps b[1] with x
   swap2(b, 1, x);                   // (c) swaps b[1] with x
   median(x +1, x, x+2);             // (d) prints 18 the median value
   cout << sqrt(5, 10, 12) << endl;  // (e) prints "Error" for any input values
   return 0;
}
```

(a) Title line for **integerPart** as called at the line marked (a).

Answer:

```
int integerPart(double x)
```

(b) Title line for **swap1** as called at the line marked (b).

Answer:

```
void swap1(int &a, int &b)
```

(c) Title line for **swap2** as called at the line marked (c).

Answer:

```
void swap2(int a[], int i, int &b)
```

(d) Title line for **median** as called at the line marked (d).

Answer:

```
void median(int a, int b, int c)
```

(e) Title line for **sqrt** as called at the line marked (e).

Answer:

```
string sqrt(int a, int b, int c)
```

Problem 212 Consider the following C++ program.

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

int fun(int x) {
   if (x <= 0) return 100;
   if (x >= 9 && x % 2 == 1) return x + 1;
   if (x >= 9 || x % 3 == 0) return x + 2;
   return 5;
}

int rec(int x) {
  if (x < 100) return x/10;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-144) << endl;   // line (a)
    cout << fun(92) << endl;   // line (b)
    cout << rec(92) << endl;   // line (c)
    cout << rec(-144) << endl; // line (d)
    cout << rec(678) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
100
```

(b) What is the output at line (b)?

Answer:

```
94
```

(c) What is the output at line (c)?

Answer:

```
9
```

(d) What is the output at line (d)?

Answer:

```
-14
```

(e) What is the output at line (e)?

Answer:

```
13
```

Problem 213 Write a function called *multiDigit* that prints a new number formed from a positive integer parameter by printing each odd digit twice and omitting all even digits. If a negative parameter is given, it should print the word *Done* and if 0 is entered it should do nothing.

For example, a program that uses the function *multiDigit* follows.

```
int main() {
   multiDigit(1245); cout << endl;  // prints 1155
   multiDigit(19683); cout << endl; // prints 119933
   multiDigit(220); cout << endl;   // prints
   multiDigit(-10); cout << endl;   // prints Done
   return 0;
}
```

Answer:

```
void multiDigit(int n) {
   if (n < 0) cout << "Done";
   else if (n == 0) return;
   else {
      multiDigit(n / 10);
      if (n % 2 != 0) {
         cout << n % 10;
         cout << n % 10;
      }
   }
}
```

Problem 214 Write a function called *randFill* that fills the entries of an array with random two digit integers. (Use an appropriate C++ function to generate the random numbers.)

For example, a program that uses the function follows.

```
int main() {
   int x[4];
   randFill(x, 4);
   for (int i = 0; i <= 3; i++)
      cout << x[i] << endl;     // prints 4 random two digit numbers
   return 0;
}
```

Answer:

```
void randFill(int a[], int c) {
   for (int i = 0; i < c; i++)
      a[i] = rand() % 90 + 10;
}
```

Problem 215 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   int x = 8, y = 10;
   cout << (x + 1.0/y) << endl;
```

Answer:

```
8.1
```

(ii)

```
char x = 'f';
while (x <= 'i') {
   cout << (char) (x - 1);
   x = x + 1;
}
```

Answer:

```
efgh
```

(iii)

```
cout << 'f' - 'c';
```

Answer:

```
3
```

(iv)

```
string x = "Easy Question";
cout << x.substr(4,1);
```

Answer:

(v)

```
int main(int argc, char *argv[]) {
   cout << argv[0];
```

Answer:

```
a.out
```

Problem 216 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 25.

2. It immediately stops if the supplied value of n is not legal.

3. Otherwise it prints out a triangular picture (as shown in the diagram, but with n rows) that uses the lowercase letters a, b, c, ... in sequence, and if necessary returns to the letter a after any z.

Here is an example of how the program should work:

```
Give me an integer between 1 and 25:   6
abcdef
 ghijk
  lmno
   pqr
    st
     u
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 25:  ";
   cin >> n;

   while (n < 1  || n > 25) {
      cout << "That is out of range. Give me an integer between 1 and 25:  ";
      cin >> n;
   }

   char x = 'a';
   for (int i = n; i >= 1; i--) {
       for (int j = 1; j <= n; j++)
          if ((i + j) <= n) cout << " ";
          else {
              cout << x;
              x = x + 1;
              if (x > 'z') x = 'a';
          }
       cout << endl;
   }
}
```

Problem 217 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {9, 3, 0, 4, 7};
   int x = 17;
   cout << asFraction(2, 6) << endl;    // (a) prints 2/6
   swap1(x, a[2]);                      // (b) swaps x with a[2]
   swap2(a[1], a[3]);                   // (c) swaps entry a[1] with a[3]
   median(1, 5, 4, 6, 7);               // (d) prints 5, the median entry
   cout << sqrt(5, 10, 12, 14) << endl; // (e) prints 0.5 for any input values
   return 0;
}
```

(a) Title line for **asFraction** as called at the line marked (a).

Answer:

```
string asFraction(int a, int b)
```

(b) Title line for **swap1** as called at the line marked (b).

Answer:

```
void swap1(int &a, int &b)
```

(c) Title line for **swap2** as called at the line marked (c).

Answer:

```
void swap2(int &a, int &b)
```

(d) Title line for **median** as called at the line marked (d).

Answer:

```
void median(int a, int b, int c, int d, int e)
```

(e) Title line for **sqrt** as called at the line marked (e).

Answer:

```
double sqrt(int a, int b, int c, int d)
```

Problem 218 Consider the following C++ program.

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

int fun(int x) {
   if (x <= 0) return 100;
   if (x >= 9 && x % 2 == 1) return x + 1;
   if (x >= 9 || x % 3 == 0) return x + 2;
   return 5;
}

int rec(int x) {
  if (x < 100) return x/10;
  return rec(x / 10) + rec(x % 100);
}

int main() {
    cout << fun(-144) << endl;   // line (a)
    cout << fun(71) << endl;   // line (b)
    cout << rec(71) << endl;   // line (c)
    cout << rec(-256) << endl; // line (d)
    cout << rec(729) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

100

(b) What is the output at line (b)?

Answer:

72

(c) What is the output at line (c)?

Answer:

7

(d) What is the output at line (d)?

Answer:

-25

(e) What is the output at line (e)?

Answer:

9

Problem 219 Write a function called *multiDigit* that prints a new number formed from an integer parameter by printing each odd digit and omitting all even digits. If a negative parameter is given, it should ignore the − sign and treat the parameter as if it was positive.

For example, a program that uses the function *multiDigit* follows.

```
int main() {
   multiDigit(1245); cout << endl;  // prints 15
   multiDigit(19683); cout << endl; // prints 193
   multiDigit(220); cout << endl;   // prints
   multiDigit(-132); cout << endl;  // prints 13
   return 0;
}
```

Answer:

```
void multiDigit(int n) {
   if (n < 0) multiDigit(-n);
   else if (n == 0) return;
   else {
      multiDigit(n / 10);
      if (n % 2 != 0) cout << n % 10;
   }
}
```

Problem 220 Write a function called *randFill* that fills the entries of an array with random integers between a specified pair of limits. (Use an appropriate C++ function to generate the random numbers.)

For example, a program that uses the function follows.

```
int main() {
   int x[4];
   int min = 20, max = 29;
   randFill(x, 4, min, max);
   for (int i = 0; i <= 3; i++)
      cout << x[i] << endl;     // prints 4 random numbers between 20 and 29
   return 0;
}
```

Answer:

```
void randFill(int a[], int c, int min, int max) {
   for (int i = 0; i < c; i++)
      a[i] = rand() % (max - min + 1) + min;
}
```

Problem 221 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   int x = 7, y = 10;
   cout << (x/y + 2.0/y) << endl;
```

Answer:

```
0.2
```

(ii)

```
char x = 'f';
while (x >= 'a') {
   cout << x;
   x = x - 1;
}
```

Answer:

```
fedcba
```

(iii)

```
cout << 'Z' - 'A';
```

Answer:

```
25
```

(iv)

```
string x = "Easy Question";
cout << x.substr(4,2);
```

Answer:

```
Q
```

(v)

```
int main(int argc, char *argv[]) {
   cout << argv[2];
```

Answer:

```
input2
```

Problem 222 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter an integer n that is between 1 and 9.

2. It immediately stops if the supplied value of n is not legal.

3. Otherwise it prints out a triangular picture (as shown in the diagram, but with n rows) that uses the lowercase letters a, b, c, ... in sequence, and if necessary continues with uppercase letter starting at A after any z.

Here is an example of how the program should work:

```
Give me an integer between 1 and 9:    7
abcdefg
hijklm
nopqr
stuv
wxy
zA
B
```

Answer:

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

int main() {
   int n;
   cout << "Give me an integer between 1 and 9:  ";
   cin >> n;
   if (n < 1  || n > 9) return 0;

   char x = 'a';
   for (int i = n; i >= 1; i--) {
       for (int j = 1; j <= i; j++) {
           cout << x;
           x = x + 1;
           if (x > 'z') x = 'A';
       }
       cout << endl;
   }
}
```

Problem 223 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int b[4] = {2,7,1,8};
   int x[2][2] = {{0,1},{3,2}};

   cout << max(i, j, k) << endl;          // prints maximum
   printMax(a, 4);                        // prints maximum
   cout << max2d(x, 2, 2) << endl;        // prints maximum
   swap(i, j);                            // swap
   swapArrays(a, b, 4);                   // swap first 4 elements in arrays
   return 0;
}
```

(a) Title line for **max**

Answer:

```
int max(int a, int b, int c)
```

(b) Title line for **printMax**

Answer:

```
void printMax(int a[], int cap)
```

(c) Title line for **max2d**

Answer:

```
int max2d(int a[][2], int r, int c)
```

(d) Title line for **swap**

Answer:

```
void swap(int &a, int &b)
```

(e) Title line for **swapArrays**

Answer:

```
void swapArrays(int a[], int b[], int n)
```

Problem 224 Consider the following C++ program.

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

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x > 0) cout << "Goodbye" << endl;
  if (x < -10) {
     cout << x + 2 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i < x; i++) cout << i;
  cout << endl;
  for (int i = 1; i <= -x; i++) {
       for (int j = 1; j <= 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *-729*?

Answer:

```
-727
```

(b) What is the output if the user enters *4*?

Answer:

```
Goodbye
123
```

(c) What is the output if the user enters *-5*?

Answer:

```
odd

***
***
***
***
***
```

(d) What is the output if the user enters *-4*?

Answer:

```
***
***
***
***
```

(e) What is the output if the user enters *3*?

Answer:

```
Goodbye
odd
12
```

Problem 225 Write a function called *doubleFirst* that places an extra copy of the first digit at the start of a number.

For example, a program that uses the function *doubleFirst* follows.

```
int main() {
   cout << doubleFirst(29) << endl;    // prints    229
   cout << doubleFirst(19683) << endl; // prints 119683
   cout << doubleFirst(9) << endl;     // prints     99
   return 0;
}
```

Answer:

```
int doubleFirst(int x) {
   if (x < 10) return 11*x;
   return doubleFirst(x / 10) * 10 + x % 10;
}
```

Problem 226 Write a function called *findLargest* that finds the largest possibility for the sum of the entries in a row of a 2-dimensional array of integers. The array and the capacities are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3) << endl;
     // prints    12, because the sum 12 = 2+4+6 is larger than 1+3+5
   return 0;
}
```

Answer:

```
int findLargest(int d[][3], int r, int c) {
   int value = 0;
   for (int col = 0; col < c; col++)
      value = value + d[0][col];
   for (int row = 0; row < r; row++) {
      int rowValue = 0;
      for (int col = 0; col < c; col++)
         rowValue = rowValue + d[row][col];
      if (rowValue > value) value = rowValue;
   }
   return value;
}
```

Problem 227 Write **title lines** for the functions most of which are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   cout << numSixes("19683") << endl;              // (a) prints 1
   printNumSixes(19683);                           // (b) prints 1
   cout << longest(961, 1961, 5) << endl;          // (c) prints 1961
   average(2.5, 3.4, 4.0);                         // (d) prints 3.3
   return 0;
}
```

(a) Title line for **numSixes**

Answer:

```
int numSixes(string a)
```

(b) Title line for **printNumSixes**

Answer:

```
void printNumSixes(int x)
```

(c) Title line for **longest**

Answer:

```
int longest(int a, int b, int c)
```

(d) Title line for **average**

Answer:

```
void average(double a, double b, double c)
```

(e) The required title line for a main program that uses arguments.

Answer:

```
int main(int argc, char *argv[])
```

Problem 228 Consider the following C++ program.

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
  ifstream infile("file.txt");
  for (int line = 1; line <= 5; line++) {
     cout << "Line " << line << " ";
     int x;
     if (infile.eof()) cout << "Done";
     infile >> x;
     if (x > 10) cout << ++x;
     if (x >  5) cout << 2 * x;
     if (x >  0) cout << x;
     if (x < 0) {
        infile >> x;
        cout << x;
     }
     cout << endl;
  }
  return 0;
}
```

The file called *file.txt* exists in the directory in which the above program is run. The file consists of the following data:

```
0   2  22   -2   2  -2 -22 22 222 2222
```

(a) What is the output line that begins: Line 1?

Answer:

```
Line 1
```

(b) What is the output line that begins: Line 2?

Answer:

```
Line 2 2
```

(c) What is the output line that begins: Line 3?

Answer:

```
Line 3 234623
```

(d) What is the output line that begins: Line 4?

Answer:

```
Line 4 2
```

(e) What is the output line that begins: Line 5?

Answer:

```
Line 5 -22
```

Problem 229 Write a function called *sum3* that determines the sum of the first 3 digits in a parameter. If the parameter has fewer than 3 digits, the sum of whatever digits are present is reported. (Assume that the parameter always has a positive value.)

For example, a program that uses the function *sumSq* follows.

```
int main() {
   cout << sum3(3456) << endl;  // prints 12 as the sum 3 + 4 + 5
   cout << sum3(11113) << endl; // prints 3 as the sum 1 + 1 + 1
   cout << sum3(9) << endl;     // prints 9
   return 0;
}
```

Answer:

```
int sum3(int x) {
  if (x < 1000) return x / 100 + (x / 10) % 10 + x % 10;
  return sum3(x / 10);
}
```

Problem 230 Write a function called *numPositive* that finds the number of rows with positive sum in a 2-dimensional array of decimals that has 4 columns. The array and the capacities are parameters. (Note that 0 is not positive.)

For example, a program that uses the function follows.

```
int main() {
   double d[2][4] = {{2, 4, -6, -8}, {-1, -3, 5, 1.5}};
   cout << numPositive(d, 2, 4) << endl;
     // prints    1  because only one row, the 2nd has a positive sum
   return 0;
}
```

Answer:

```
int numPositive(double d[][4], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++) {
      double rowSum = 0;
      for (int j = 0; j < c; j++)
          rowSum = rowSum + d[i][j];
      if (rowSum > 0) count++;
   }
   return count;
}
```

Problem 231 Write a function called *numX* that reports the number of elements in a array of strings that contain an uppercase letter *X*.

For example, a program that uses the function follows.

```
int main() {
   string data[4] = {"abcdXYZ", "Hello", "1234", "XXX"};
   cout << numX(data, 4);   // prints:  2 because 2 strings include an X
   return 0;
}
```

Answer:

```
int numX(string a[], int cap) {
   int ans = 0;
   for (int i = 0; i < cap; i++)
      if (((int) a[i].find("X")) >= 0) ans++;
   return ans;
}
```

Problem 232 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer n.

2. It repeatedly reads n from the user until the supplied value of n is positive.

3. It prints out a large letter N that has height n and width n. The locations of the printed characters should lie in the $n \times n$ square region that the letter occupies.

Here is an example of how the program should work:

```
Give me a positive integer:   5
N   N
NN  N
N N N
N  NN
N   N
```

Answer:

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

int main() {
   int n;
   cout << "Enter a value for n:";
```

```
   cin >> n;
   while (n <= 0) {
      cout << "No good. Give a positive value: ";
      cin >> n;
   }
   for (int i = 1; i <= n; i++) {
      for (int j = 1; j <= n; j++)
         if (j == 1 || j == n || i == j)
             cout << "N";
         else cout << " ";
      cout << endl;
   }
   return 0;
}
```

Problem 233 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};

   cout << average(i, j, k) << endl;          // prints average
   printAverage(a, 4);                        // prints average
   cout << average2d(x, 2, 2) << endl;        // prints average
   sort(i, j ,k );                            // sort into order
   sort3(a, 4);                               // sort into order
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int a, int b, int c)
```

(b) Title line for **printAverage**

Answer:

```
void printAverage(int a[], int cap)
```

(c) Title line for **average2d**

Answer:

```
double average2d(int x[][2], int r, int c)
```

(d) Title line for **sort**

Answer:

```
void sort(int &a, int &b, int &c)
```

(e) Title line for **sort3**

Answer:

```
void sort3(int a[], int cap)
```

Problem 234 Consider the following C++ program.

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

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x < 0) cout << "Goodbye" << endl;
  if (x > 10) {
     cout << x % 10 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i <= x; i++) cout << i;
  cout << endl;
  for (int i = 1; i < -x; i++) {
       for (int j = 1; j < 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *729*?

Answer:

```
9
```

(b) What is the output if the user enters *9*?

Answer:

```
odd
123456789
```

(c) What is the output if the user enters *5*?

Answer:

```
odd
12345
```

(d) What is the output if the user enters *4*?

Answer:

```
1234
```

(e) What is the output if the user enters *-3*?

Answer:

```
Goodbye
odd

**
**
```

Problem 235 Write a function called *dropSecond* that removes the second digit of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one digit, return that digit.)

For example, a program that uses the function *dropSecond* follows.

```
int main() {
   cout << dropSecond(29) << endl;    // prints    2, the 9 dropped
   cout << dropSecond(19683) << endl; // prints 1683, the 9 dropped
   cout << dropSecond(9) << endl;     // prints    9
   return 0;
}
```

Answer:

```
int dropSecond(int x) {
   if (x < 10) return x;
   if (x < 100) return x / 10;
   return dropSecond(x / 10) * 10 + x % 10;
}
```

Problem 236 Write a function called *findLargest* that finds the largest entry in a specified column of a 2-dimensional array of integers. The array, the capacities, and the specified column are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3, 0) << endl;
     // prints    2, because this is the largest entry in column 0
   return 0;
}
```

Answer:

```
int findLargest(int d[][3], int r, int c, int x) {
   int ans = d[0][x];
   for (int row = 0; row < r; row++)
      if (d[row][x] > ans) ans = d[row][x];
   return ans;
}
```

Problem 237 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   cout << numDigits(19683) << endl;              // (a) prints 5
   printNumDigits("19683");                       // (b) prints 5
   cout << longer("Hello", "Goodbye") << endl;    // (c) prints "Goodbye"
   biggest(3.14, 2.718, 1.5);                     // (d) prints 3.14
   cout << sqrt(5, 10, 12) << endl;               // (e) prints the sum as 27
   return 0;
}
```

(a) Title line for **numDigits**

Answer:

```
int numDigits(int x)
```

(b) Title line for **printNumDigits**
Answer:

```
void printNumDigits(string x)
```

(c) Title line for **longer**
Answer:

```
string longer(string a, string b)
```

(d) Title line for **biggest**
Answer:

```
void biggest(double a, double b, double c)
```

(e) Title line for **sqrt** as called at the line marked (e).
Answer:

```
int sqrt(int a, int b, int c)
```

Problem 238 Consider the following C++ program.

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
  ifstream infile("file.txt");
  for (int line = 1; line <= 5; line++) {
     cout << "Line " << line << " ";
     int x;
     if (infile.eof()) cout << "Done";
     infile >> x;
     if (x > 10) cout << ++x;
     if (x >  5) cout << 2 * x;
     if (x >  0) cout << x;
     if (x < 0) {
        infile >> x;
        cout << x;
     }
     cout << endl;
  }
  return 0;
}
```

The file called *file.txt* exists in the directory in which the above program is run. The file consists of the following data:

```
0   4   6  14   -1   3  -2 -5 1 2 3
```

(a) What is the output line that begins: Line 1?
Answer:

```
Line 1
```

(b) What is the output line that begins: Line 2?
Answer:

```
Line 2 4
```

(c) What is the output line that begins: Line 3?
Answer:

```
Line 3 126
```

(d) What is the output line that begins: Line 4?
Answer:

```
Line 4 153015
```

(e) What is the output line that begins: Line 5?
Answer:

```
Line 5 3
```

Problem 239 Write a function called *sumSq* that determines the sum of the squares of the digits in a parameter.

For example, a program that uses the function *sumSq* follows.

```
int main() {
   cout << sumSq(34) << endl;    // prints 25 because this is 9 + 16
   cout << sumSq(11113) << endl; // prints 13 found as 1+1+1+1+9
   cout << sumSq(9) << endl;     // prints 81
   return 0;
}
```

Answer:

```
int sumSq(int n) {
   if (n < 10) return n * n;
   return sumSq(n/10) + sumSq(n%10);
}
```

Problem 240 Write a function called *smallestPositive* that finds the smallest positive entry in a 2-dimensional array of decimals that has 4 columns. The array and the capacities are parameters. If no entry in the array is positive, the function should return an answer of 0.0. (Note that 0 is not positive.)

For example, a program that uses the function follows.

```
int main() {
   double d[2][4] = {{2, 4, -6, 8}, {-1, -3, 5, 1.5}};
   cout << smallestPositive(d, 2, 4) << endl;
     // prints    1.5
   return 0;
}
```

Answer:

```
double smallestPositive(double d[][4], int r, int c) {
    double answer = d[0][0];
    for (int i = 0; i < r; i++)
       for (int j = 0; j < c; j++)
           if (d[i][j] > 0)
               if (answer <= 0 || d[i][j] < answer)
                   answer = d[i][j];
    if (answer > 0) return answer;
    return 0.0;
}
```

Problem 241 Write a function called *insertX* that inserts an X at the middle of each element of an array of strings. (If a string has even length, the X should be added exactly at its middle, otherwise the X should be added immediately before the middle.)

For example, a program that uses the function follows.

```
int main() {
   string data[4] = {"abcd", "Hello", "1234", ""};
   insertX(data, 4);
   for (int i = 0; i < 4; i++)
      cout << data[i] << " ";    // output: abXcd HeXllo 12X34 X
   return 0;
}
```

Answer:

```
void insertX(string d[], int cap) {
   for (int i = 0; i < cap; i++)
       d[i].insert(d[i].length()/2, "X");
}
```

Problem 242 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer n.

2. It repeatedly reads n from the user until the supplied value of n is positive.

3. It prints out a large letter Z that has height n and width n. The locations of the printed characters should lie in the $n \times n$ square region that the letter occupies.

Here is an example of how the program should work:

```
Give me a positive integer:   5
ZZZZZ
   Z
  Z
 Z
ZZZZZ
```

Answer:

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

int main() {
   int n;
   cout << "Enter a value for n:";
   cin >> n;
   while (n <= 0) {
      cout << "No good. Give a positive value: ";
      cin >> n;
   }
   for (int i = 1; i <= n; i++) {
      for (int j = 1; j <= n; j++)
         if (i == 1 || i == n || (i + j) == (n + 1))
             cout << "Z";
         else cout << " ";
      cout << endl;
   }
   return 0;
}
```

Problem 243 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **num7s** which returns the number of digits equal to 7 in an input integer.

Answer:

```
int num7s(int x)
```

(b) A function called **num7s** which returns the number of elements equal to 7 in an input array of integers.

Answer:

```
int num7s(int x[], int capacity)
```

(c) A function called **num7s** which returns the number of characters equal to 7 in an input string.

Answer:

```
int num7s(string x)
```

(d) A function called **num7s** which changes an integer parameter to be the number of 7's in its decimal expansion. (For example if the input is 777111 it would be changed to 3 because it has 3 digits equal to 7.)

Answer:

```
void num7s(int &x)
```

(e) A function called **num7s** which returns the number of elements equal to 7 in a 2-dimensional array of integers with size 7×7.

Answer:

```
int num7s(int x[][7], int rows, int cols)
```

Problem 244 Consider the following C++ program.

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

int fun(int x) {
   if (x <= 0) return 0;
   if (x >= 9 && x % 2 == 1) return x - 1;
   if (x >= 9 || x % 3 == 0) return x - 2;
   return 7;
}

int rec(int x) {
  if (x < 10) return fun(x);
  return rec(x / 10) + rec(x % 10);
}

int main() {
    cout << fun(3) << endl;    // line (a)
    cout << fun(30) << endl;   // line (b)
    cout << fun(33) << endl;   // line (c)
    cout << rec(33) << endl;   // line (d)
    cout << rec(999) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
1
```

(b) What is the output at line (b)?

Answer:

```
28
```

(c) What is the output at line (c)?

Answer:

```
32
```

(d) What is the output at line (d)?

Answer:

```
2
```

(e) What is the output at line (e)?

Answer:

```
24
```

Problem 245 Write a function called *startBinary* that returns a number giving the first 2 digits in the binary expansion of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one binary digit, return that digit.)

For example, a program that uses the function *startBinary* follows.

```
int main() {
   int x = startBinary(6);
   cout << x << endl;               // prints 11 because 6 in binary is 110
   cout << startBinary(23) << endl; // prints 10 because 23 is 10111 in binary
   cout << startBinary( 3) << endl; // prints 11 because  3 is    11 in binary
   cout << startBinary( 1) << endl; // prints  1 because  1 is     1 in binary
   return 0;
}
```

Answer:

```
int startBinary(int x) {
   if (x < 2) return x;
   if (x == 2) return 10;
   if (x == 3) return 11;
   return startBinary(x/2);
}
```

Problem 246 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter a positive integer n that is less than 100. If the user enters an incorrect value, the program terminates. The program next asks the user to enter n^2 strings to be stored in a 2-dimensional array with size $n \times n$. The program then reports the maximum number of times that it can find the string *Kamil* in any row or column of the array.

For example, if the user enters 4 for n and then enters the 16 strings:

```
Kamil Peter  Dustin Kamil
Kamil Andrew Carl   Phil
Rat   Rat    Rat    Rat
Kamil Peter  Dustin Kamil
```

The final output would be 3 because Kamil appears three times in the first column, and no more than three times in any row or column.

Answer:

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

int main() {
   int n;
   cout << "Enter a positive integer that is less than 100: ";
   cin >> n;
   if (n < 1 || n > 99) exit(1);

   string data[100][100];
   cout << "Enter " << n * n << " data items (each is a string)\n";
   for (int i = 0; i < n; i++)
     for (int j = 0; j < n; j++)
         cin >> data[i][j];

   int max = 0;
   for (int row = 0; row < n; row++) {
      int rowKamils = 0;
      for (int col = 0; col < n; col++)
          if (data[row][col] == "Kamil") rowKamils ++;
      if (rowKamils > max) max = rowKamils;
   }

   for (int col = 0; col < n; col++) {
      int colKamils = 0;
      for (int row = 0; row < n; row++)
          if (data[row][col] == "Kamil") colKamils ++;
      if (colKamils > max) max = colKamils;
   }

   cout << "The maximal number of Kamils in a row or column is " << max << endl;
   return 0;
}
```

Problem 247 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[10] = {3,1,4,1,5,9,2,6,5,3};
   int x[3][2] = {{0,1},{2,3},{4,5}};
   int n = 7, m = 2;
   int i = sum(n, m);                // sets i as the sum
   swap(n, m);                       // swaps n and m
   printArray(a, 10);                // prints content of a
   print2dArray(x, 3, 2);            // prints content of x
   cout << minElement(a, 10);        // minimum element of array
   cout << firstDigit(n*n + m*m);    // first digit
   return 0;
}
```

(a) Title line for **sum**

Answer:

```
int sum(int a, int b)
```

(b) Title line for **swap**
Answer:

```
void swap(int &a, int &b)
```

(c) Title line for **printArray**
Answer:

```
void printArray(int a[], int cap)
```

(d) Title line for **print2dArray**
Answer:

```
void print2dArray(int a[][2], int rows, int cols)
```

(e) Title line for **minElement**
Answer:

```
int minElement(int a[], int cap)
```

(f) Title line for **firstDigit**
Answer:

```
int firstDigit(int x)
```

Problem 248 Write a function called $array2F$ that returns the largest entry in a 2-dimensional array (of integer values). The parameters are the array, its number of rows and its number of columns. For example, a program that uses the function $array2F$ follows.

```
int main() {
   int a[3][4] = {{0, -2, 2, 4}, {10, -5, 1, 3}, {1, 4, 1, 0}};
   cout << array2F(a, 3, 4) << endl;    // output is 10
   return 0;
}
```

Answer:

```
int array2F(int a[][4], int rows, int cols) {
   int answer = a[0][0];
   for (int i = 0; i < rows; i++)
      for (int j = 0; j < cols; j++)
          if (a[i][j] > answer) answer = a[i][j];
   return answer;
}
```

Problem 249 Consider the following C++ program.

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

char recursive(char array[], int n) {
   char x = array[n];
   if ('a' <= x && x <= 'z') return x;
   cout << x;
   return recursive(array, n - 1);
}
```

```
int main() {
  char array[8] = {'a','b','c','d','0','1','2','3'};
  cout << array[1] << endl;                // line a
  cout << (char) (array[1] + 1) << endl;   // line b
  cout << recursive(array, 0) << endl;     // line c
  cout << recursive(array, 4) << endl;     // line d
  cout << recursive(array, 7) << endl;     // line e
  cout << array[6] - array[7] << endl;     // line f
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
b
```

(b) line b:

```
c
```

(c) line c:

```
a
```

(d) line d:

```
0d
```

(e) line e:

```
3210d
```

(f) line f:

```
-1
```

Problem 250 Write a function called *useRecursion* that returns the sum of the first two digits in a positive number. If there is only one digit, that digit is returned. For example, a program that uses the function *useRecursion* follows.

```
int main() {
   cout << useRecursion(567982) << endl;     // prints 11
   cout << useRecursion(107982) << endl;     // prints 1
   cout << useRecursion(7) << endl;          // prints 7
   return 0;
}
```

Answer:

```
int useRecursion(int x) {
   if (x < 100) return x % 10 + x / 10;
   return useRecursion(x / 10);
}
```

Problem 251 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Declare and initialize any variables that you use in each part.

(i) Print the number 7 to an output file whose system name is *out.txt*

```
ofstream f("out.txt");
f << 7 << endl;
```

(ii) Read the first line of text in an input file whose system name is *in.txt*. Store the line in an appropriate variable called *line*.

```
ifstream g("in.txt");
string line;
getline(g, line);
```

(iii) Write the title line for a main function that uses arguments.

```
int main(int argc, char *argv[])
```

(iv) Print the 5^{th} character of a string variable called *line* to the output screen.

```
cout << line[4] << endl;
```

(v) Print the character after the first character equal to K in a string variable called *line* to the output screen. If there is no character K, print the first character of the string.

```
int x = line.find('K');
if (x >= 0 && x < line.size() - 1)
   cout << line[x + 1];
else cout << line[0];
```

(vi) Print a random 2 digit integer to the output screen.

```
cout << rand() % 90 + 10 << endl;
```

Problem 252 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 20. It continues asking until the user enters a correct input.

2. The program generates two random upper case letters (using the standard C++ random number generation function).

3. The program prints an $n \times n$ square that uses the two characters to make a checkerboard pattern.

For example, if the user enters 5 and the random letters are K and W the following square picture is printed.

```
KWKWK
WKWKW
KWKWK
WKWKW
KWKWK
```

Answer:

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

int main() {

    int n;
```

```
    cout << "Enter a positive value for n that is at most 20: ";
    cin >> n;
    while ( n <= 0  || n > 20 ) {
       cout << "That is not legal.  Try again: ";
       cin >> n;
    }

    char x = 'A' + rand() % 26;
    char y = 'B' + rand() % 26;

    for (int i = 0; i < n; i++) {
       for (int j = 0; j < n; j++) {
           if ((i + j) % 2 == 0) cout << x;
           else cout << y;
       }
       cout << endl;
    }

    return 0;
}
```

Problem 253 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[10] = {3,1,4,1,5,9,2,6,5,3};
   int x[3][2] = {{0,1},{2,3},{4,5}};
   int n = 7, m = 2;
   int i = sum(n, m, n);              // sets i as the sum
   swap(n, m);                        // swaps n and m
   addToArray(a, 10, 5);              // adds 5 to every entry
   printArray(x, 3, 2);               // prints content of x
   cout << maxElement(a, 10);         // maximum element of array
   cout << firstDigit(n);             // first digit
   return 0;
}
```

(a) Title line for **sum**

Answer:

```
int sum(int a, int b, int c)
```

(b) Title line for **swap**

Answer:

```
void swap(int &a, int &b)
```

(c) Title line for **addToArray**

Answer:

```
void addToArray(int a[], int cap, int x)
```

(d) Title line for **printArray**

Answer:

```
void printArray(int a[][2], int rows, int cols)
```

(e) Title line for **maxElement**

Answer:

```
int maxElement(int a[], int cap)
```

(f) Title line for **firstDigit**

Answer:

```
int firstDigit(int x)
```

Problem 254 Write a function called *array2F* that returns the product of the negative entries in a 2-dimensional array (of integer values). The parameters are the array, its number of rows and its number of columns. For example, a program that uses the function *array2F* follows.

```
int main() {
   int a[3][4] = {{0, -2, 2, 4}, {10, -5, 1, 3}, {1, 4, 1, 0}};
   cout << array2F(a, 3, 4) << endl;    // output is 10
   return 0;
}
```

Answer:

```
int array2F(int a[][4], int rows, int cols) {
   int answer = 1;
   for (int i = 0; i < rows; i++)
      for (int j = 0; j < cols; j++)
          if (a[i][j] < 0) answer *= a[i][j];
   return answer;
}
```

Problem 255 Consider the following C++ program.

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

char recursive(char array[], int n) {
   char x = array[n];
   if ('a' == x || x == 'b') return x;
   cout << x;
   return recursive(array, n - 1);
}

int main() {
  char array[8] = {'a','b','c','d','0','1','2','3'};
  cout << array[0] << endl;                // line a
  cout << (char) (array[0] + 3) << endl;   // line b
  cout << recursive(array, 0) << endl;     // line c
  cout << recursive(array, 2) << endl;     // line d
  cout << recursive(array, 7) << endl;     // line e
  cout << array[7] - array[5] << endl;     // line f
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
a
```

(b) line b:

```
d
```

(c) line c:

```
a
```

(d) line d:

```
cb
```

(e) line e:

```
3210dcb
```

(f) line f:

```
2
```

Problem 256 Write a function called *useRecursion* that returns the larger of the first two digits in a positive number. If there is only one digit, that digit is returned. For example, a program that uses the function *useRecursion* follows.

```
int main() {
   cout << useRecursion(567982) << endl;     // prints 6
   cout << useRecursion(107982) << endl;     // prints 1
   cout << useRecursion(7) << endl;          // prints 7
   return 0;
}
```

Answer:

```
int useRecursion(int x) {
   if (x < 100) {
      if (x / 10 > x % 10) return x / 10;
      return x % 10;
   }
   return useRecursion(x / 10);
}
```

Problem 257 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Declare and initialize any variables that you use in each part.

(i) Read the first line of text in an input file whose system name is *input.txt*. Store the line in an appropriate variable called *line*.

```
   ifstream g("input.txt");
   string line;
   getline(g, line);
```

(ii) Print the number 2 to an output file whose system name is *output.txt*

```
   ofstream f("output.txt");
   f << 2 << endl;
```

(iii) Print the length of a string variable called *line* to the output screen.

```
   cout << line.length() << endl;
```

(iv) Write the title line for a main function that uses arguments.

```
int main(int argc, char *argv[])
```

(v) Print the character before the first character equal K in a string variable called *line* to the output screen. If there is no character K, or no character before it print the first character of the string.

```
   int x = line.find('K');
   if (x <= 0) cout << line[0];
   cout << line[x - 1];
```

(vi) Print a random 3 digit integer to the output screen.

```
   cout << rand() % 900 + 100 << endl;
```

Problem 258 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 20. It continues asking until the user enters a correct input.
2. The program generates n^2 random upper case letters (using the standard C++ random number generation function).
3. The program prints an $n \times n$ square that is filled with its chosen random letters.

For example, if the user enters 5 the following square picture might be printed.

```
KWXDG
YKWQT
AGDKE
IEXVL
UGBLQ
```

Answer:

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

int main() {

    int n;
    cout << "Enter a positive value for n that is at most 20: ";
    cin >> n;
    while ( n <= 0  || n > 20 ) {
       cout << "That is not legal.  Try again: ";
       cin >> n;
    }

    char x[20][20];
    for (int i = 0; i < 20; i++)
       for (int j = 0; j < 20; j++)
          x[i][j] = 'A' + rand() % 26;

    for (int i = 0; i < n; i++) {
       for (int j = 0; j < n; j++) {
           cout << x[i][j];
       }
       cout << endl;
    }

    return 0;
}
```

Problem 259 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[10] = {3,1,4,1,5,9,2,6,5,3};
   int x[3][2] = {{0,1},{2,3},{4,5}};
   int n = 7, m = 2;
   int i = diff(n, m);               // sets i as the difference
   swap(n, m);                       // swaps values of inputs
   printArray(a, 10);                // prints content of a
   addToArray(x, 3, 2, 5);           // adds 5 to every entry in array
   cout << average(a, 10);           // average of array
   cout << first2Digits(n + m);      // first two digits
   return 0;
}
```

(a) Title line for **diff**

Answer:

```
int diff(int a, int b)
```

(b) Title line for **swap**

Answer:

```
void swap(int &a, int &b)
```

(c) Title line for **printArray**

Answer:

```
void printArray(int a[], int cap)
```

(d) Title line for **addToArray**

Answer:

```
void addToArray(int a[][2], int rows, int cols, int x)
```

(e) Title line for **average**

Answer:

```
double average(int a[], int cap)
```

(f) Title line for **first2Digits**

Answer:

```
string first2Digits(int x)
```

Problem 260 Write a function called $array2F$ that returns the number of non-zero entries in a 2-dimensional array (of integer values). The parameters are the array, its number of rows and its number of columns. For example, a program that uses the function $array2F$ follows.

```
int main() {
   int a[3][4] = {{0, -2, 2, 4}, {10, -5, 1, 3}, {1, 4, 1, 0}};
   cout << array2F(a, 3, 4) << endl;    // output is 10
   return 0;
}
```

Answer:

```
int array2F(int a[][4], int rows, int cols) {
   int answer = 0;
   for (int i = 0; i < rows; i++)
      for (int j = 0; j < cols; j++)
          if (a[i][j] != 0) answer++;
   return answer;
}
```

Problem 261 Consider the following C++ program.

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

char recursive(char array[], int n) {
   char x = array[n];
   if ('0' <= x && x <= '9') return x;
   cout << x;
   return recursive(array, n - 1);
}

int main() {
  char array[8] = {'0','1','2','3','a','b','c','d'};
  cout << array[1] << endl;                // line a
  cout << (char) (array[1] + 1) << endl;   // line b
  cout << recursive(array, 0) << endl;     // line c
  cout << recursive(array, 4) << endl;     // line d
  cout << recursive(array, 7) << endl;     // line e
  cout << array[6] - array[7] << endl;     // line f
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
1
```

(b) line b:

```
2
```

(c) line c:

```
0
```

(d) line d:

```
a3
```

(e) line e:

```
dcba3
```

(f) line f:

```
-1
```

Problem 262 Write a function called *useRecursion* that returns the second digit in a positive number. If there is only one digit, that digit is returned. For example, a program that uses the function *useRecursion* follows.

```
int main() {
   cout << useRecursion(567982) << endl;     // prints 6
   cout << useRecursion(107982) << endl;     // prints 0
   cout << useRecursion(7) << endl;          // prints 7
   return 0;
}
```

Answer:

```
int useRecursion(int x) {
   if (x < 100) return x % 10;
   return useRecursion(x / 10);
}
```

Problem 263 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Declare and initialize any variables that you use in each part.

(i) Write the title line for a main function that uses arguments.

```
int main(int argc, char *argv[])
```

(ii) Print the number 13 to an output file whose system name is *out.txt*

```
   ofstream f("out.txt");
   f << 13 << endl;
```

(iii) Read the first string in an input file whose system name is *in.txt*. Store the string in an appropriate variable called *data*.

```
   ifstream g("in.txt");
   string data;
   getline(g, data);
```

(iv) Print the 8^{th} character of a string variable called *line* to the output screen.

```
   cout << line[7] << endl;
```

(v) Print the position of the first character equal to K in a string variable called *line* to the output screen. If there is no character K, print -1.

```
   int x = line.find('K');
   if (0 <= x && x < line.length())
      cout << x << endl;
   else cout << -1 << endl;
```

(vi) Print a random 5 digit integer to the output screen.

```
   cout << rand() % 90000 + 10000 << endl;
```

Problem 264 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 20. If an incorrect response is entered it exits.

2. The program generates a random upper case letter and a random lower case letter (using the standard C++ random number generation function).

3. The program prints an $n \times n$ square that uses the two characters to make a checkerboard pattern.

For example, if the user enters 5 and the random letters are K and w the following square picture is printed.

```
KwKwK
wKwKw
KwKwK
wKwKw
KwKwK
```

Answer:

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

int main() {

    int n;
    cout << "Enter a positive value for n that is at most 20: ";
    cin >> n;
    if ( n <= 0  || n > 20 ) {
       cout << "That is not legal.  " << endl;
       exit(0);
    }

    char x = 'A' + rand() % 26;
    char y = 'a' + rand() % 26;

    for (int i = 0; i < n; i++) {
       for (int j = 0; j < n; j++) {
           if ((i + j) % 2 == 0) cout << x;
           else cout << y;
       }
       cout << endl;
    }
    return 0;
}
```

Problem 265 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **isNegative** that tests whether a decimal number is negative.

Answer:

```
bool isNegative(double x)
```

(b) A function called **thirdChar** which uses a string as input and returns the third character in the string.

Answer:

```
char thirdChar(string s)
```

(c) A function called **swapLast2** which modifies an array of integers. The task of the function is to swap the last two elements of the array.

Answer:

```
void swapLast2(int a[], int cap)
```

(d) A function called **printPic** which uses as input an 6×6 array of characters that represents a picture. The task of the function is to print the picture.

Answer:

```
void printPic(char pic[][6], int rows, int cols)
```

(e) A function called **reverseArray** which is to reverse the order of elements in an array of integers.

Answer:

```
void reverseArray(int x[], int cap)
```

Problem 266 Consider the following C++ program.

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

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int &p, int q) {
  int temp = p;
  p = q;
  q = temp;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int x[8] = {0, 1, 2, 3, 4, 5, 6, 7};
  int y[7] = {0, 1, 2, 3, 4, 5, 6};
  int a = 3, b = 4;

  print(x, 3);                          // line (a)
  mystery(x, 1, 2);  print(x, 5);       // line (b)
  for (int i = 1; i <= 7; i++) mystery(x, 0 ,i);
  print(x, 8);                          // line (c)
  m2(a, b);    cout << a << b << endl; // line (d)
  m2(y[3], 7);       print(y, 6);       // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
0 1 2
```

(b) What is the output at line (b)?

Answer:

```
0 2 2 3 4
```

(c) What is the output at line (c)?

Answer:

```
7 2 2 3 4 5 6 7
```

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

```
0 1 2 7 4 5
```

Problem 267 Write a function called *doubleDigit* that makes each digit of an input parameter repeat twice. For example, a program that uses the function *doubleDigit* follows.

```
int main() {
   cout << doubleDigit(9)   << endl;          // prints 99
   cout << doubleDigit(81)  << endl;          // prints 8811
   cout << doubleDigit(243) << endl;          // prints 224433
   cout << doubleDigit(244) << endl;          // prints 224444
   return 0;
}
```

Answer:

```
int doubleDigit(int n) {
   if (n < 10) return n * 11;
   return 100 * doubleDigit(n / 10) + doubleDigit(n % 10);
}
```

Problem 268 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the digit or digits that appears least often.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, 2 appears 103 times, 3 appears 103 times, 4 appears 93 times, 5 appears 97 times, 6 appears 94 times, 7 appears 95 times, 8 appears 101 times, 9 appears 105 times the output would be:

```
The digits 0 and 4 are least frequent.
```

Answer:

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

int main() {
   int count[10];
   int x;
   for (int c = 0; c < 10; c++)
      count [c] = 0;

   cout << "Enter 1000 single digit integers: ";
   for (int c = 1; c <= 1000; c++) {
      cin >> x;
      count[x]++;
   }

   int min = count[0];
   for (int c = 1; c <= 9; c++) {
      if (count[c] < min) min = count[c];
   }
```

```
   bool found = false;
   cout << "The digits ";
   for (int c = 0; c <= 9; c++) {
      if (count[c] == min) {
         if (found) cout << "and ";
         cout << c << " ";
      }
      found = true;
   }
   cout << "are least frequent.\n";
   return 0;
}
```

Problem 269 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **detectAge** which returns a user's age (by asking for input and rejecting negative values).

Answer:

```
int detectAge()
```

(b) A function called **sortString** that sorts an array of strings into alphabetical order.

Answer:

```
void sortString(string a[], int cap)
```

(c) A function called **sort4** that sorts 4 integer parameters into increasing order.

Answer:

```
void sort4(int &a, int &b, int &c, int &d)
```

(d) A function called **printCode** that prints the ASCII code for a character.

Answer:

```
void printCode(char x)
```

(e) A function called **delete7** which alters an integer parameter by deleting every occurrence of the digit 7.

Answer:

```
void delete7(int &x)
```

Problem 270 Consider the following C++ program.

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

void mystery(int x[][4], int a, int b, int k) {
  for (int r = a; r <= b; r++) for (int c = a; c <= b; c++)
      x[r][c] = k;
}

void print(int x[][4], int s) {
  for (int r = 0; r < s; r++) {
     for (int c = 0; c < s; c++) cout << x[r][c];
     cout << endl;
  }
```

```
  cout << endl;
}

int main() {
  int x[4][4] = {{0,0,0,0}, {0,0,0,0}, {0,0,0,0}, {0,0,0,0}};
  mystery(x, 3, 2, 1); print(x, 4);  // line (a)
  mystery(x, 0, 1, 2); print(x, 4);  // line (b)
  mystery(x, 1, 2, 3); print(x, 4);  // line (c)
  mystery(x, 1, 3, 4); print(x, 4);  // line (d)
  mystery(x, 0, 3, 5); print(x, 2);  // line (e)
  return 0;
}
```

(a) What is the output at line (a)?

Answer:

```
0000
0000
0000
0000
```

(b) What is the output at line (b)?

Answer:

```
2200
2200
0000
0000
```

(c) What is the output at line (c)?

Answer:

```
2200
2330
0330
0000
```

(d) What is the output at line (d)?

Answer:

```
2200
2444
0444
0444
```

(e) What is the output at line (e)?

Answer:

```
55
55
```

Problem 271 Write a function called *cutNine* that prints the part of a number that follows its last 9 digit. (If there is no 9 digit, the whole number is printed. If the last digit is a 9, nothing is printed.)

For example, a program that uses the function *cutNine* follows.

```
int main() {
   cutNine(770);          cout << endl;          // prints 770
```

```
   cutNine(135792468);   cout << endl;          // prints 2468
   cutNine(1991991);     cout << endl;          // prints 1
   cutNine(1991999);     cout << endl;          // prints
   return 0;
}
```

Answer:

```
void cutNine(int n) {
  if (n == 0 || n % 10 == 9) return;
  cutNine(n/10);
  cout << n % 10;
}
```

Problem 272 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the number of times that each digit was seen.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, ..., 9 appears 105 times, the output would be:

```
0 count 93, 1 count 116, 2 count 103, 3 count 103, 4 count 93,
5 count 97, 6 count 94, 7 count 95, 8 count 101, 9 count 105.
```

Answer:

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

int main() {
   int count[10];
   int x;
   for (int c = 0; c < 10; c++)
      count [c] = 0;

   cout << "Enter 1000 single digit integers: ";
   for (int c = 1; c <= 1000; c++) {
      cin >> x;
      count[x]++;
   }

   for (int c = 0; c < 10; c++) {
      cout << c << " count " << count[c];
      if (c % 5 < 4) cout << ", ";
      else if (c == 4) cout << ",\n";
      else cout << ".\n";
   }
   return 0;
}
```

Problem 273 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};
```

```
   printArray(a, 3);                  // outputs:  3,1,4
   printVals(i + j, a[0]);            // outputs: 8 3
   reverse(a, 0, 3);                  // changes a to 1,4,1,3
   cout << sumElements(x, 2 , 2);     // outputs: 6
   sort(i, j, k);
   cout << i << j << k << endl;       // prints 345
   return 0;
}
```

(a) Title line for **printArray**

Answer:

```
void printArray(int a[], int cap)
```

(b) Title line for **printVals**

Answer:

```
void printVals(int x, int y)
```

(c) Title line for **reverse**

Answer:

```
void reverse(int a[], int i, int j)
```

(d) Title line for **sumElements**

Answer:

```
int sumElements(int x[][2], int r, int c)
```

(e) Title line for **sort**

Answer:

```
void sort(int &a, int &b, int &c)
```

Problem 274 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer that is between 1 and 26.

2. The program reads a value n entered by the user. If the value is not legal, the program exits.

3. The program prints an $n \times n$ pattern of characters, in which the bottom right character is an 'A'. The bottom right 2×2 block is completed by three 'B' characters. The bottom right 3×3 block is completed by five 'C' characters, and so on.

For example, if the user enters 5 for n the program should print the following picture.

```
EEEEE
EDDDD
EDCCC
EDCBB
EDCBA
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int n;
   cout << "Enter an integer between 1 and 26: ";
```

```
   cin >> n;
   if (n < 1 || n > 26) exit(1);

   for (int r = 1; r <= n; r++) {
      char k = (char) ('A' + n - r);
      for (int c = 1; c <= n; c++) {
         if (c < r) k = (char) ('A' + n - c);
         cout << k;
      }
      cout << endl;
   }
   return 0;
}
```

Problem 275 Write a function called *emergency* that detects whether a number contains the sequence of digits 911. For example, a program that uses the function *emergency* follows.

```
int main() {
   if (emergency(56791182)) cout << "Warning" << endl;  // prints warning
   if (emergency(56791212)) cout << "Warning" << endl;  // no print here
   if (emergency(91191191)) cout << "Warning" << endl;  // prints warning
   return 0;
}
```

Answer:

```
bool emergency(int x) {
   if (x <= 0) return false;
   if (x % 1000 = 911) return true;
   return emergency(x/10);
}
```

Problem 276 Consider the following C++ program.

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

string recursive(string x) {
   if (x.length() == 0) return ":";
   return x.substr(0,1) + "#" + recursive(x.substr(1));

}

int main(int argc, char *argv[]) {
  int i = 1, j = 2, k = 3;
  string array[2] = {"", "hello"};
  cout << ++k << endl;                      // line a
  k = ++i - j++;
  cout << i << j << k << endl;              // line b
  cout << recursive(array[0]) << endl;      // line c
  cout << recursive(array[1]) << endl;      // line d
  cout << argv[1]      << endl;             // line e
  return 0;
}
```

The program is compiled to produce a binary called a.out. The binary is run with the command:

```
venus>   ./a.out CS111 Final Exam
```

What is the output from the program at each of the following lines:
(a) line a:

```
4
```

(b) line b:

```
230
```

(c) line c:

```
:
```

(d) line d:

```
h#e#l#l#o#:
```

(e) line e:

```
CS111
```

Problem 277 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the quotient when x is divided into y.

```
cout << y/x << endl;
```

(ii) Print *table*[2][2] to the file *out.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream f("out.txt");
f << table[2][2];
```

(iii) Print HELLO if you can find the substring *Freddy* within *name*. Otherwise print HI.

```
if (name.find("Freddy") >= 0) cout << "HELLO";
else cout << "HI";
```

(iv) Print the sum of all the numbers in column number 17 of the 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns. As usual the array begins with row number 0.)

```
int ans = 0;
for (int r = 0; r <= 99; r++)
   ans += table[r][17];
cout << ans;
```

(v) Print a random integer value between 13 and 19 (inclusive) to the screen. (The random integer should be determined by using an appropriate C++ function.)

```
cout << rand() % 7 + 13;
```

Problem 278 Write a complete C++ program that does the following.

1. It asks the user to enter positive integers a and b that are each at most 100.
2. The program reads in a table of integers with a rows and b columns as entered by the user.
3. The program determines and prints the maximum entry in each column of the table.
4. The program then prints the smallest value among these maximum entries.

For example, the following represents one run of the program.

```
Enter integers for r and c (at most 100):     2 2
Enter 2 rows of 2 integers:
 1  4
 2  0
The maximum entries in the columns are:  2 4
The smallest of the printed maximum entries is :  2
```

Answer:

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

int main() {
   int a, b;
   int table[100][100];
   int max[100];
   int minMax;
   cout << "Enter integers for r and c (at most 100): ";
   cin >> a >> b;
   cout << "Enter " << a << " rows of " << b << " integers:\n";

   for (int r = 0; r < a; r++) {
      for (int c = 0; c < b; c++)
         cin >> table[r][c];
   }

   cout << "The maximum entries in the columns are:  ";
   for (int c = 0; c < b; c++) {
      max[c] = table[0][c];
      for (int r = 0; r < a; r++)
         if (table[r][c] > max[c]) max[c] = table[r][c];
      cout << max[c] << " ";
   }
   cout << "\nThe smallest of the printed maximum entries is : ";
   cout << endl;
   return 0;
}
```

Problem 279 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **middleDigit** which returns the middle digit of an integer.

Answer:

```
int middleDigit(int x)
```

(b) A function called **sqrt** that returns the square root of a double precision parameter.

Answer:

```
double sqrt(double x)
```

(c) A function called **duplicateString** which returns a new copy of string.

Answer:

```
string duplicate(string original)
```

(d) A function called **randomFile** which is to return a randomly created name to use for an output file.

Answer:

```
string randomFile()
```

(e) A function called **selectionSort** which is to sort an array of strings into alphabetical order.

Answer:

```
void selectionSort(string data[], int length)
```

Problem 280 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times (2n-1)$ pattern of $*$ symbols in the shape of a large solid triangle.

For example, if the user enters 4 for n the program should print the following picture.

```
   *
  ***
 *****
*******
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int n;
   cout << "Enter a positive integer: ";
   cin >> n;
   while (n <= 0) {
      cout << "That is not positive. Try again: ";
      cin >> n;
   }

   for (int row = 1; row <= n; r++) {
      int rowSpace = n - row;
      int rowStars = 2 * row - 1;
      for (int c = 1; c <= rowSpace; c++) cout << " ";
      for (int c = 1; c <= rowStars; c++) cout << "*";
      cout << endl;
   }
   return 0;
}
```

Problem 281 Write a function called *removeFirst* that removes the first digit from a number. The answer should be returned as an integer. (Drop any leading 0 digits in the answer. So that as in the example below, removing the first from 1024 leaves 24.)

A program that uses the function *removeFirst* follows.

```
int main() {
   int n = 19683;
   int m = removeFirst(n);
   cout << m << endl;                      // output 9683
   cout << removeFirst(1024);              // output   24
   return 0;
}
```

Answer:

```
int removeFirst(int n) {
  if (n < 10) return 0;
  return removeFirst(n / 10) * 10 + n % 10;
}
```

Problem 282 Consider the following C++ program.

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

string recursive(string x) {
   if (x.length() <= 1) return x;
   return x.substr(0,2) + recursive(x.substr(1));

}

int main(int argc, char *argv[]) {
  int i = 1, j = 2, k = 3;
  string array[2] = {"A", "hello"};
  cout << ++argc << endl;                  // line a
  k = ++i * j++;
  cout << i << j << k << endl;             // line b
  cout << recursive(array[0]) << endl;     // line c
  cout << recursive(array[1]) << endl;     // line d
  cout << recursive(argv[3]) << endl;     // line e
  return 0;
}
```

The program is compiled to produce a binary called a.out. The binary is run with the command:

```
venus>   ./a.out CS111 Final Exam
```

What is the output from the program at each of the following lines:
(a) line a:

```
5
```

(b) line b:

```
234
```

(c) line c:

```
A
```

(d) line d:

```
heellllloo
```

(e) line e:

```
Exxaamm
```

Problem 283 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Include declarations for any variable that you use.

(i) Print the word *HELLO* to the file *out.txt*.

```
ofstream f("out.txt");
f << "HELLO";
```

(ii) Print a random upper case letter to the screen. (The random letter should be determined by using an appropriate C++ function.)

```
cout << (char) ('A' + rand() % 26);
```

(iii) Read a line of text from the user and print the word *NO* if it contains the string *Fred*.

```
string name;
cin >> name;
if (name.find("Freddy") >= 0) cout << "NO";
```

(iv) Print the first 4 characters of the string s. Assume that the string has length at least 4.

```
cout << s.substr(0, 4) << endl;
```

(v) Swap the values of integer variables called p and q.

```
int temp = p;
p = q;
q = temp;
```

Problem 284 Write a complete C++ program that does the following.

1. It asks the user to enter positive integers a and b that are each at most 20.
2. The program generates random integer values between 1 and 6 as the entries in a table with a rows and b columns.
3. The program then prints the table.
4. The program then prints the diagonal entries from the table.

For example, the following represents one run of the program.

```
Enter integers for r and c (at most 20):     2 2
The table has been generated as:
6 3
1 2
The diagonal is:  6 2
```

Answer:

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

int main() {
   int a, b, row, col;
   int table[21][21];
   cout << "Enter integers for r and c (at most 20): ";
   cin >> a >> b;
```

```
   for (row = 1; row <= a; row++)
      for (col = 1; col <= b; col++)
         table[row][col] = rand() % 6 + 1;

   cout << " The table has been generated as: " << endl;
   for (row = 1; row <= a; row++) {
      for (col = 1; col <= b; col++)
         cout << table[row][col] << " ";
      cout << endl;
   }

   cout << " The diagonal is: ";
   for (row = 1; row <= a && row <= b; row++)
      cout << table[row][row] << " ";
   cout << endl;

   return 0;
}
```

Problem 285 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three double parameters.

Answer:

```
double add3 (double a, double b, double c)
```

(b) A function called **reverseIt** that returns the number obtained by reversing the digits of an integer parameter.

Answer:

```
int reverseIt (int x)
```

(c) A function called **randomArray** that sets the values in an array of doubles to have random values.

Answer:

```
void randomArray (double arr [], int capacity)
```

(d) A function called **add5** that adds 5 to every entry in a 2-dimensional array each of whose rows has 35 columns.

Answer:

```
void add5 (int arr [][35], int rows, int columns)
```

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.

Answer:

```
void deleteX (string &str)
```

Problem 286 Consider the following C++ program.

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

string fun(string x[], int y) {
   if (y <= 0) return x[1];
   if (y == 1) return x[0] + x[2];
   if (y == 2) return "illegal";
```

```
   if (y <= 4) return " 4";
   return "X" + fun(x, y - 6);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;            // line a
  cout << fun(array,1) << endl;            // line b
  cout << fun(array,2) << endl;            // line c
  cout << fun(array,4) << endl;            // line d
  cout << fun(array,12) << endl;           // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
2
```

(b) line b:

```
13
```

(c) line c:

```
illegal
```

(d) line d:

```
4
```

(e) line e:

```
XX2
```

Problem 287 Write a function called *makeOne* that returns the result of turning every odd valued digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

```
int makeOne (int x)
{
        if (x < 10 && x % 2 == 1) return 1;
        if (x < 10) return x;
        return 10 * makeOne (x / 10) + makeOne (x % 10);
}
```

Problem 288 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the least frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 123, 45678, and 200 the program should output 9 which occurs less often than any other digit in these numbers.

Answer:

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

void getDigits (int n, int count []) {
   while (n > 0) {
      int digit = n % 10;
      count [digit]++;
      n /= 10;
   } //while
}

int main () {
   int n1, n2, n3;
   cout << "Enter three positive integers: ";
   cin >> n1 >> n2 >> n3;

   int count [10];
   for (int i = 0; i < 10; i++)
      count [i] = 0;

   getDigits (n1, count);
   getDigits (n2, count);
   getDigits (n3, count);

   int min = count[0];
   for (int i = 1; i < 10; i++)
        if (count[i] < min) min = count[i];

   cout << "The following digits occur least often:" << endl;
   for (int i = 0; i < 10; i++)
        if (count[i] == min) cout << i << endl;

   return 0;
}
```

Problem 289 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three integer parameters.

Answer:

```
int add3(int a, int b, int c)
```

(b) A function called **reverseString** that returns the reverse of a string.

Answer:

```
string reverseString(string str)
```

(c) A function called **randomArray** that sets the values in an array of integers to have random values.

Answer:

```
void randomArray (int arr [], int capacity)
```

(d) A function called **add3** that adds 3 to every entry in a 2-dimensional array each of whose rows has 25 columns.
Answer:

```
void add3 (int arr [][25], int rows, int columns)
```

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.
Answer:

```
void deleteX (string &str)
```

Problem 290 Consider the following C++ program.

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

string fun(string x[], int y) {
   if (y <= 0) return x[0];
   if (y == 1) return x[1] + x[2];
   if (y == 2) return "illegal";
   if (y <= 4) return " <= 4";
   return "X" + fun(x, y - 5);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;            // line a
  cout << fun(array,1) << endl;            // line b
  cout << fun(array,2) << endl;            // line c
  cout << fun(array,4) << endl;            // line d
  cout << fun(array,12) << endl;           // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
1
```

(b) line b:

```
23
```

(c) line c:

```
Illegal
```

(d) line d:

```
<= 4
```

(e) line e:

```
XXillegal
```

Problem 291 Write a function called *makeOne* that returns the result of turning every non-zero digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

```
int makeOne(int x)
{
        if (x == 0) return 0;
        if (x < 10) return 1;
        return 10 * makeOne (x / 10) + makeOne (x % 10);
}
```

Problem 292 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the most frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 737, 13579, and 246 the program should output 7 which occurs more often than any other digit in these numbers.

Answer:

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

void getDigits (int n, int count []) {
   while (n > 0) {
      int digit = n % 10;
      count [digit]++;
      n /= 10;
   } //while
}

int main () {
   int n1, n2, n3;
   cout << "Enter three positive integers: ";
   cin >> n1 >> n2 >> n3;

   int count [10];
   for (int i = 0; i < 10; i++)
      count [i] = 0;

   getDigits (n1, count);
   getDigits (n2, count);
   getDigits (n3, count);

   int max = count[0];
   for (int i = 1; i < 10; i++)
        if (count[i] > max) max = count[i];

   cout << "The following digits occur most often:" << endl;
```

```
        for (int i = 0; i < 10; i++)
           if (count[i] == max) cout << i << endl;

   return 0;
}
```

Problem 293 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string name = "Freddy", secondName = "Fred";
   cout << thirdChar(name);             // print the 3rd character
   if ( !isLegal(name) )                // reject illegal names
      readName(name);                   // and reads a name entered by the user
   exchangeNames(name, secondName);      // Swap the two names
   cout << bothNames(name, secondName);  // print full name
   return 0;
}
```

(a) Title line for **thirdChar**

Answer:

```
char thirdChar(string name)
```

(b) Title line for **isLegal**

Answer:

```
bool isLegal(string name)
```

(c) Title line for **readName**

Answer:

```
void readName(string &name)
```

(d) Title line for **exchangeNames**

Answer:

```
void exchangeNames(string &name, string &otherName)
```

(e) Title line for **bothNames**

Answer:

```
string bothNames(string name, string otherName)
```

Problem 294 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the remainder when x is divided into y.

```
cout << y % x;
```

(ii) Print *name* to the file *out.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream fout("out.txt");
fout << name;
```

(iii) Read a line of text from the file *out.txt* into the variable *name*.

```
ifstream fin("out.txt");
getline(fin, name);
```

(iv) Print the average of all the numbers in row number 17 of the 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns. As usual the array begins with row number 0.)

```
int sum = 0;
for (int a = 0; a < 100; a++)
  sum += table[17][a];
cout << sum / 100.0;
```

(v) Print a sequence of 20 random integer values each between 1 and 20 (inclusive) to the screen. (The random integers should be determined by using an appropriate C++ function.)

```
for (int a = 0; a < 20; a++)
  cout << rand() % 20 + 1;
```

Problem 295 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times n$ pattern of $*$ symbols in the shape of an empty right triangle (with the point down).

For example, if the user enters 7 for n the program should print the following picture.

```
*******
 *    *
  *   *
   *  *
    * *
     **
      *
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int n = -1;
   while (n < 0) {
      cout << "Enter a positive integer: ";
      cin >> n;
   }

   for (int r = 1; r <= n; r++) {
      for (int c = 1; c <= n; c++) {
         if (r == c || r == 1 || c == n) cout << "*";
         else cout << " ";
      }
      cout << endl;
   } //for r
} //main
```

Problem 296 Write a function called *evenUp* that uses an integer parameter and returns a result that is found by increasing each even digit in the parameter by 1. For example, if the parameter has value 19683 the returned result would be 19793.

A program that uses the function *evenUp* follows.

```
int main() {
   cout << evenUp(10) << endl;       // prints 11
   cout << evenUp(2662) << endl;     // prints 3773
   cout << evenUp(19683) << endl;    // prints 19793
   return 0;
}
```

Answer:

```
int evenUp(int x) {
   if (x < 10 && x % 2 == 0) return x + 1;
   if (x < 10 && x % 2 == 1) return x;
   return 10 * evenUp(x / 10) + evenUp(x % 10);
}
```

Problem 297 For each of the following short segments of a program write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   double x = 4, y = 8;
   bool z = (x <= y || y <= x);
   if (z) cout << y / x;
   else cout << x / y;
   cout << endl;
```

Answer:

```
2.0
```

(ii)

```
   char Int = 'C';
   Int = Int + 1;
   cout << Int << endl;
```

Answer:

```
D
```

(iii)

```
   int i = 1;
   while (i++ < 10) {
      cout << ++i << endl;
   }
```

Answer:

```
3
5
7
9
11
```

(iv)

```
int x[3][3] = {{1,2,3}, {4,7,10}, {11,15,19}};
for (int i = 0; i <= 2; i++)
   cout << x[i][i];
cout << endl;
```

Answer:

```
1719
```

(v)

```
string x[3] = {"Hello", "CS111", "Exam"};
for (int j = 1; j <= 3; j++) for (int i = 2; i >= 0; i--)
   cout << x[i][j];
cout << endl;
```

Answer:

```
xSea1lm1l
```

Problem 298 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 20.

2. The program then reads n words from the user. (You should assume that each word contains between 1 and 10 characters.)

3. The program then prints a summary giving the number of words with each length.

For example, the following represents one run of the program.

```
Enter an integer n (at most 20):     3
Enter 3 words:              Hello CS111 Exam
Length 4:  count 1
Length 5:  count 2
```

In the exam the words *Hello* and *CS111* have length 5, and give the count of 2 words with length 5. No counts are printed for word lengths other than 4 and 5 because no other word lengths are encountered in this example.

Answer:

```
#include <iostream>
using namespace std;
int main ()
{
   int n;
   cout << "Enter positive integer that is at most 20: ";
   cin >> n;

   string words[20];
   cout << "Enter " << n << " words: ";
   for (int a = 0; a < n; a++) cin >> words[a];

   int count [11]; //for lengths 1 thru 10 inclusive, nothing of length 0
   for (int a = 0; a < 11; a++) count[a] = 0;
   for (int a = 0; a < n; a++) {
      int len = words[a].length();
      count[len]++;
   } //for

   for (int a = 0; a < 11; a++)
      if (count[a] != 0)
         cout << "Length " << a << ": count " << count[a] << endl;
} //main
```

Problem 299 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string name = "Freddy", secondName = "Fred";
   fixThirdChar(name);                  // change the 3rd character to X
   if ( !isLegal(secondName) )          // reject illegal names
      secondName = readName();          // and reads a name entered by the user
   exchangeNames(name, secondName);     // Swap the two names
   printBothNames(name, secondName);    // print full name
   return 0;
}
```

(a) Title line for **fixThirdChar**

Answer:

```
void fixThirdChar(string &name)
```

(b) Title line for **isLegal**

Answer:

```
bool isLegal(string name)
```

(c) Title line for **readName**

Answer:

```
string readName()
```

(d) Title line for **exchangeNames**

Answer:

```
void exchangeNames(string &name, string &otherName)
```

(e) Title line for **printBothNames**

Answer:

```
void printBothNames(string name, string otherName)
```

Problem 300 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the remainder when y is divided by x.

```
cout << y % x;
```

(ii) Print *table*[0][0] to the file *output.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream fout("output.txt");
fout << table[0][0];
```

(iii) Read a line of text from the file *output.txt* into the variable *name*.

```
ifstream fin("output.txt");
getline(fin, name);
```

(iv) Print the average of all the numbers in column number 37 of the 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns. As usual the array begins with column number 0.)

```
int sum = 0;
for (int a = 0; a < 100; a++)
  sum += table[a][37];
cout << sum / 100.0;
```

(v) Print a sequence of 10 random integer values each between 1 and 100 (inclusive) to the screen. (The random integers should be determined by using an appropriate C++ function.)

```
for (int a = 0; a < 10; a++)
  cout << rand() % 100 + 1;
```

Problem 301 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times n$ pattern of $*$ symbols in the shape of an empty right triangle (with the point up).

For example, if the user enters 7 for n the program should print the following picture.

```
      *
     **
    * *
   *  *
  *   *
 *    *
*******
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int n = -1;
   while (n < 0) {
      cout << "Enter positive integer: ";
      cin >> n;
   }

   for (int r = 1; r <= n; r++) {
      for (int c = 1; c <= n; c++) {
         if (r == n || c == n || r + c == n + 1) cout << "*";
         else cout << " ";
      }
      cout << endl;
   } //for r
} //main
```

Problem 302 Write a function called *bigDown* that uses an integer parameter. It returns a result that is found from the parameter by subtracting 1 from any digit that is 5 or larger. For example, if the parameter has value 19683 the returned result would be 18573.

A program that uses the function *bigDown* follows.

```
int main() {
   cout << bigDown(10) << endl;       // prints 10
   cout << bigDown(2654) << endl;     // prints 2544
   cout << bigDown(19683) << endl;    // prints 18573
   return 0;
}
```

Answer:

```
int bigDown(int x) {
   if (x < 5) return x;
   if (x < 10) return x - 1;
   return 10 * bigDown(x / 10) + bigDown(x % 10);
}
```

Problem 303 For each of the following short segments of a program write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   double x = 4, y = 8;
   bool z = (x <= y && y <= x);
   if (z) cout << y / x;
   else cout << x / y;
   cout << endl;
```

Answer:

```
0.5
```

(ii)

```
   char Int = 'D';
   Int = Int - 1;
   cout << Int << endl;
```

Answer:

```
C
```

(iii)

```
   int i = 1;
   while (++i < 10) {
      cout << i++ << endl;
   }
```

Answer:

```
2
4
6
8
```

(iv)

```
   int x[3][3] = {{4,7,10}, {11,15,19}, {1,2,3}};
   for (int i = 0; i <= 2; i++)
      cout << x[i][i];
   cout << endl;
```

Answer:

```
4153
```

(v)

```
string x[3] = {"CS111", "Exam", "Hello"};
for (int j = 1; j <= 3; j++) for (int i = 2; i >= 0; i--)
   cout << x[i][j];
cout << endl;
```

Answer:

```
exSla1lm1
```

Problem 304 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 25.

2. The program then reads n words from the user. (You should assume that each word contains between 3 and 12 characters.)

3. The program then prints a summary giving the number of words with each length.

For example, the following represents one run of the program.

```
Enter an integer n (at most 20):     3
Enter 3 words:              Hello CS111 Exam
Length 4:  count 1
Length 5:  count 2
```

In the exam the words *Hello* and *CS111* have length 5, and give the count of 2 words with length 5. No counts are printed for word lengths other than 4 and 5 because no other word lengths are encountered in this example.

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int n;
   cout << "Enter positive integer that is at most 25: ";
   cin >> n;

   string words[25];
   cout << "Enter " << n << " words: ";
   for (int a = 0; a < n; a++) cin >> words[a];

   int count[13]; // lengths upto 12
   for (int a = 0; a < 13; a++) count[a] = 0;
   for (int a = 0; a < n; a++) {
      int len = words[a].length();
      count [len]++;
   } //for

   for (int a = 0; a < 13; a++)
      if (count[a] != 0)
         cout << "Length " << a << ": count " << count[a] << endl;
} //main
```

Problem 305 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the remainder when x is divided by y.

```
cout << x % y;
```

(ii) Print *table*[1][1] to the file *outfile.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream fout ("outfile.txt");
fout << table[1][1];
```

(iii) Read a line of text from the file *infile.txt* into the variable *name*.

```
ifstream fin("outfile.txt");
getline(fin, name);
```

(iv) Print the average of all the numbers in row number 27 of the 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns. As usual the array begins with row number 0.)

```
int sum = 0;
for (int a = 0; a < 100; a++)
  sum += table[27][a];
cout << sum / 100.0;
```

(v) Print two random integer values each between 100 and 200 (inclusive) to the screen. (The random integers should be determined by using an appropriate C++ function.)

```
cout << rand() % 101 + 100;
cout << rand() % 101 + 100;
```

Problem 306 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times n$ pattern of $*$ symbols in the shape of an empty right triangle (with the point up).

For example, if the user enters 7 for n the program should print the following picture.

```
*
**
* *
*  *
*   *
*    *
*******
```

Answer:

```
#include <iostream>
using namespace std;
int main(){
   int n = -1;
   while (n < 0) {
      cout << "Enter positive integer: ";
      cin >> n;
   }
```

```
   for (int r = 1; r <= n; r++) {
      for (int c = 1; c <= n; c++) {
         if (r == n || c == 1 || r == c) cout << "*";
         else cout << " ";
      }
      cout << endl;
   } //for r
} //main
```

Problem 307 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the remainder when y is divided into x.

```
cout << x % y;
```

(ii) Print x and y to the file *out.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream fout("out.txt");
fout << x << y;
```

(iii) Read a word of text from the file *infile.txt* into the variable *name*.

```
ifstream fin("infile.txt");
fin >> name;
```

(iv) Print the average of all the numbers in column number 27 of the 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns. As usual the array begins with column number 0.)

```
(iv)
 int sum = 0;
 for (int a = 0; a < 100; a++)
   sum += table[a][27];
 cout << sum / 100.0;
```

(v) Print two random integer values each between 10 and 99 (inclusive) to the screen. (The random integers should be determined by using an appropriate C++ function.)

```
cout << rand() % 90 + 10;
cout << rand() % 90 + 10;
```

Problem 308 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times n$ pattern of $*$ symbols in the shape of an empty right triangle (with the point down).

For example, if the user enters 7 for n the program should print the following picture.

```
*******
*    *
*   *
*  *
* *
**
*
```

Answer:

```
#include <iostream>
using namespace std;
int main(){
   int n = -1;
   while (n < 0) {
      cout << "Enter positive integer: ";
      cin >> n;
   }

   for (int r = 1; r <= n; r++) {
      for (int c = 1; c <= n; c++) {
         if (r == 1 || c == 1 || r + c == n + 1) cout << "*";
         else cout << " ";
      }
      cout << endl;
   } //for r
} //main
```

Problem 309 For each of the following short segments of a program write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   double x = 4, y = 8;
   bool z = (x > y || y > x);
   if (z) cout << y / x;
   else cout << x / y;
   cout << endl;
```

Answer:

```
2.0
```

(ii)

```
   char Int = 'd';
   Int = Int + 1;
   cout << Int << endl;
```

Answer:

```
e
```

(iii)

```
   int i = 1;
   while (i++ < 10) {
      cout << i++ << endl;
   }
```

Answer:

```
2
4
6
8
10
```

(iv)

```
int x[3][3] = {{1,2,3}, {4,7,10}, {11,15,19}};
for (int i = 0; i <= 2; i++)
   cout << x[i][2 - i];
cout << endl;
```

Answer:

```
3711
```

(v)

```
string x[3] = {"Hello", "CS111", "Exam"};
for (int j = 1; j <= 3; j++) for (int i = 0; i <= 2; i++)
   cout << x[i][j];
cout << endl;
```

Answer:

```
eSxl1al1m
```

Problem 310 For each of the following short segments of a program write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
double x = 4, y = 8;
bool z = (x > y && y > x);
if (z) cout << y / x;
else cout << x / y;
cout << endl;
```

Answer:

```
0.5
```

(ii)

```
char Int = 'b';
Int = Int - 1;
cout << Int << endl;
```

Answer:

```
a
```

(iii)

```
int i = 1;
while (++i < 10) {
   cout << i++ << endl;
}
```

Answer:

```
2
4
6
8
```

(iv)

```
int x[3][3] = {{4,7,10}, {11,15,19}, {1,2,3}};
for (int i = 0; i <= 2; i++)
   cout << x[i][2 - i];
cout << endl;
```

Answer:

```
10151
```

(v)

```
string x[3] = {"CS111", "Exam", "Hello"};
for (int j = 1; j <= 3; j++) for (int i = 0; i <= 2; i++)
   cout << x[i][j];
cout << endl;
```

Answer:

```
Sxe1al1ml
```

Problem 311 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3,1,4,1}, b[5] = {2,7,1,8,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};
   cout << max(x, 2 , 2); // outputs:  3
   printArray(a, 4);  // outputs:  3,1,4,1
   reverse(a, 0, 3);  // changes a to 1,4,1,3
   sort1(b, 5);
   printArray(b, 5);  // outputs:  1,1,2,7,8
   sort2(i, j, k);
   cout << i << j << k << endl;  // outputs: 345
   return 0;
}
```

(a) Title line for **max**

Answer:

```
int max(int x[][2], int a, int b)
```

(b) Title line for **printArray**

Answer:

```
void printArray(int array[], int cap)
```

(c) Title line for **reverse**

Answer:

```
void reverse(int array[], int from, int to)
```

(d) Title line for **sort1**

Answer:

```
void sort1(int array[], int n)
```

(e) Title line for **sort2**

Answer:

```
void sort2(int &a, int &b, int &c)
```

Problem 312 Consider the following C++ program.

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

void rec(int a[], int start, int stop) {
   if (stop <= start) return;
   a[start] = a[stop];
   rec(a, start + 1, stop -1);
}

void printA(int a[], int cap) {
  for (int c = cap - 1; c >= 0; c--) cout << a[c] << " ";
  cout << endl;
}

int main() {
  int x[6] = {0, 1, 2, 3, 4, 5};

  printA(x, 6);                  // line (a)
  printA(x, 4);                  // line (b)
  rec(x, 3, 3); printA(x, 4);    // line (c)
  rec(x, 3, 4); printA(x, 6);    // line (d)
  rec(x, 0, 5); printA(x, 6);    // line (e)

  return 0;
}
```

What is the output at each of the following lines?

(a) line (a)

```
5 4 3 2 1 0
```

(b) line (b)

```
3 2 1 0
```

(c) line (c)

```
3 2 1 0
```

(d) line (d)

```
5 4 4 2 1 0
```

(e) line (e)

```
5 4 4 4 4 5
```

Problem 313 Write a function called *maxMid* that determines the maximum value in the middle column of a 2-dimensional array of numbers of type double. (You should assume that the 2-dimensional array has an odd number of columns.)

For example, a program that uses the function *maxMid* follows. Your function must complete this program.

```
int main() {
   double x[4][5] = {{0,1,2,3,4}, {1,2,3,4,5}, {2,3,4,5,6}, {5,6,7,8,9}};
   cout << maxMid(x, 4, 5) << endl;  // prints 7.0
   return 0;
}
```

Answer:

```
double maxMid(double x[][5], int rows, int cols) {
   double ans = x[0][cols / 2];
   for (int i = 0; i < rows; i++)
      if ((x[i][cols / 2] > ans) ans = x[i][cols / 2];
   return ans;
}
```

Problem 314 Write a complete C++ program that does the following. (In your program, you do not need to check whether the user enters legal input.)

1. It asks the user to enter a positive integer n that is at most 100.

2. The program reads n single digit integers entered by the user. (A single digit integer is an integer n with $0 \leq n \leq 9$.)

3. The program prints a list of all single digit integers that were not entered at all by the user.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):    11
Enter 11 single digit integers:
1 1 7 3 3 2 0 3 7 7 7
The following were not entered: 4 5 6 8 9
```

Answer:

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

int main() {
   int n, c, x, count[10];
   for (int c = 0; c <= 9; c++) count[c] = 0;

   cout << "Enter a positive integer (at most 100): ";
   cin >> n;
   cout << "Enter " << n << " single digit integers: ";

   for (int c = 0; c < n; c++) {
      cin >> x;
      count[x]++;
   }
   cout << "The following were not entered:";
   for (int c = 0; c <= 9; c++)
      if (count[c] == 0) cout << " " << c;
   cout << endl;

   return 0;
}
```

Problem 315 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **welcome** which prints the word "Hello" to the screen.

Answer:

```
void welcome()
```

(b) A function called **addTwo** that adds 2 to every entry in an array of integers.

Answer:

```
void addTwo(int array[], int cap)
```

(c) A function called **randomTruth** which determines and returns a random true/false result.

Answer:

```
bool randomTruth()
```

(d) A function called **numberPrimes** which returns the number of prime numbers that lie between a specified pair of input values.

Answer:

```
int numberPrimes(int a, int b)
```

(e) A function called **biggerAverage** which determines which of two arrays of integers has the bigger average. It should return the value of this bigger average.

Answer:

```
double biggerAverage(int array1[], int cap1, int array2[], int cap2)
```

Problem 316 Consider the following C++ program.

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

int fun(int &x, int y) {
   x = y + 1;
   y = x + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  fun(x, 8);                    // line a
  fun(x, y);                    // line b
  fun(y, x);                    // line c
  fun(y, x);                    // line d
  cout << fun(x, 3) << endl;    // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
910
```

(b) line b:

```
12
```

(c) line c:

```
23
```

(d) line d:

```
23
```

(e) line e:

```
45
20
```

Problem 317 Write a function called *alternates* that prints every second digit of an integer parameter, starting from the right.

For example, a program that uses the function *alternates* follows.

```
int main() {
   alternates(10);  cout << endl;     // prints 0
   alternates(123456); cout << endl;  // prints 642
   alternates(1000); cout << endl;    // prints 00
   return 0;
}
```

Answer:

```
void alternates(int n) {
   cout << n % 10;
   if (n >= 100) alternates(n/100);
}
```

An alternative solution that does not use recursion follows:

```
void alternates (int x) {
   while (x > 0) {
      cout << x % 10;
      x /= 100;
   }
}
```

Problem 318 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer that is between 1 and 26.

2. The program reads a value n entered by the user. If the value is not legal, the program exits.

3. The program prints an $n \times n$ pattern of characters, in which the top left character is an 'A'. The top left 2×2 block is completed by three 'B' characters. The top left 3×3 block is completed by five 'C' characters, and so on.

For example, if the user enters 5 for n the program should print the following picture.

```
ABCDE
BBCDE
CCCDE
DDDDE
EEEEE
```

Answer:

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

```
int main() {
   int r, c, x, n;
   char pic[26][26];

   cout << "Enter an integer between 1 and 26: ";
   cin >> n;
   if (n < 1 || n > 26) exit(1);

   for (x = n - 1; x >= 0; x--)
      for (r = 0; r <= x; r++)
         for (c = 0; c <= x; c++) pic[r][c] = 'A' + x;

   for (r = 0; r <= n - 1; r++) {
      for (c = 0; c <= n - 1; c++) cout << pic[r][c];
      cout << endl;
   }

   return 0;
}
```

Problem 319 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string name;  int x, y, array[20];
   name = enterName();        // Reads a name entered by the user
   cout << lastChar(name);    // Print the last character
   enterNumbers(x, y);        // Ask for and read in values for x and y
   cout << power(x, y);       // x raised to the power y
                              // answer is decimal to allow for negative powers
   cout << reverse(name);     // Prints the name backwards
                              // so Fred would be printed as derF
   randomize(array, 20);      // fill the array with random numbers
   return 0;
}
```

(a) Title line for **lastChar**

```
char lastChar (string name)
```

(b) Title line for **enterNumbers**

```
void enterNumbers (int &a, int &b)
```

(c) Title line for **power**

```
double power (int a, int b)
```

(d) Title line for **reverse**

```
string reverse (string name)
```

(e) Title line for **randomize**

```
void randomize (int arr[], int cap)
```

Problem 320 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
int x[10], z[10][10], r, c;
```

(i) Increase every entry of x by 1.

```
for (int i = 0; i < 10; i++) x[i]++;
```

(ii) Set r to be a random integer between c and c + 10. (The random integer should be determined by an appropriate C++ function.)

```
r = rand () % 11 + c;
```

(iii) Print the sum of all 100 entries of the 2-dimensional array z.

```
int sum = 0;
for (int i = 0; i < 10; i++)
  for (int j = 0; j < 10; j++)
    sum += z [i][j];
```

(iv) Print the last 5 entries of the array x.

```
for (int i = 5; i < 10; i++) cout << x [i];
```

(v) Swap column number 2 with column number 3 in the 2-dimensional array z.

```
for (int i = 0; i < 10; i++) {
  int temp = z [i][2];
  z [i][2] = z [i][3];
  z [i][3] = temp;
} //for
```

Problem 321 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints the first n squares and their sum.

For example, if the user enters 4 for n the program should produce the following output.

```
1 4 9 16
sum to 30
```

Answer:

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

int main ()
{
  int a = -1;
  while (a < 0) {
     cout << "Give me a positive integer: ";
     cin >> a;
```

```
  } //while

  int sum = 0;
  for (int i = 1; i <= a; i++) {
     int temp = i * i;
     cout << temp << " ";
     sum += temp;
  } //for
  cout << endl << "sum to " << sum << endl;

  return 0;
} //main
```

Problem 322 Write a function called *boeing* that prints a parameter with additional digits of 7 before each digit and at the end of the number. (So that a parameter 4 would be printed as 747 and a parameter 666 would be printed as 7676767.)

For example, a program that uses the function *boeing* follows.

```
int main() {
   boeing(4);     cout << endl;   // prints 747
   boeing(66);    cout << endl;   // prints 76767
   boeing(7);     cout << endl;   // prints 777
   boeing(1000);  cout << endl;   // prints 717070707
   return 0;
}
```

Answer:

```
void boeing(int n) {
   if (n < 10) cout << 7 << n << 7;
   else {
      boeing(n / 10);
      cout << n % 10 << 7;
   }
}
```

Problem 323 Consider the following C++ program.

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

int recursive(int x[], int n) {
   if (n <= 0 || n > 10) return 0;
   if (n == 1) return x[0];
   if (n <= 3) return x[n - 1] + recursive(x, n - 1);
   x[0]++;
   return recursive(x, n - 3);
}

int main() {
  int x, a[10] = {1,2,3,4,5,6,7,8,9,10};
  cout << "Enter a number: ";
  cin >> x;
  cout << recursive(a, x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.
(a) The user enters 0
Answer: 0
(b) The user enters 1
Answer: 1
(c) The user enters 3
Answer: 6
(d) The user enters 5
Answer: 4
(e) The user enters 10
Answer: 4

Problem 324 Write a complete C++ program that does the following.
1. It asks the user to enter positive integers a and b that are each at most 100.
2. The program reads in a table of integers with a rows and b columns as entered by the user.
3. The program determines and prints the minimum entry in each column of the table.
4. The program then prints the average value of these minimum entries.
For example, the following represents one run of the program.

```
Enter integers for r and c (at most 100):     2 2
Enter 2 rows of 2 integers:
 1  4
 2  0
The minimum entries in the columns are:  1 0
The average minimum entry is :  0.5
```

Answer:

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

int main () {
   int a, b, r, c, min, sumMin = 0;
   int table [100][100];

   cout << "Give me two integers, each at most 100: ";
   cin >> a >> b;

   cout << "Enter " << a << " rows of " <<  b << " integers: " << endl;
   for (r = 0; r < a; r++)
      for (c = 0; c < b; c++)
         cin >> table [r][c];

   cout << "The minimum entries in the columns are:  ";
   for (c = 0; c < b; c++) {
      min = table [0][c];
      for (r = 0; r < a; r++) if (table [r][c] < min) min = table[r][c];
      cout << min << " ";
      sumMin += min;
  } //for  c

  cout << "\nThe average minimum entry is :  ";
  cout << ((double) sumMin) / b << endl;

  return 0;
} //main
```

Problem 325 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string name;
   name = enterName();                // Reads a name entered by the user
   greet(name);                       // Says hello to the user
   cout << numberAs(name);            // Finds the number of As in the name
   string theClass[20];
   enterNames(theClass, 20);          // Enter the names of all students
   sort(theClass, 20, "decreasing");  // sort names into decreasing
                                      // alphabetical order
   printNames(theClass, 20);
   return 0;
}
```

(a) Title line for **enterName**

```
string enterName()
```

(b) Title line for **greet**

```
void greet(string name)
```

(c) Title line for **numberAs**

```
int numberAs(string name)
```

(d) Title line for **enterNames**

```
void enterNames(string names[], int cap)
```

(e) Title line for **sort**

```
void sort(string names[], int cap, string ordering)
```

Problem 326 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part. All other necessary variables should be declared and initialized.

```
int x, y, table[100][100];
string name;
```

(i) Print the larger of integer variables called x and y.

```
int larger = x;
if (y > x) larger = y;
cout << larger;
```

(ii) Print the numbers *10 9 8* to the file *out.txt*. (In this part you need to declare a variable to access the file.)

```
ofstream fout ("out.txt");
fout << 10 << 9 << 8;
```

(iii) Read a line of text from the user and print the word *Yes* if it contains the substring *Freddy*.

```
cin >> name;
if (name.find ("Freddy") != -1) cout << "Yes";
```

(iv) Print the sum of all the numbers in column number 0 of a 2-dimensional array called *table*. (The array *table* has 100 rows and 100 columns.)

```
int sum = 0;
for (int i = 0; i < 100; i++)
  sum += table [i][0];
cout << sum;
```

(v) Print 8 random **negative** integers to the screen. (The random integers should be determined by using an appropriate C++ function.)

```
for (int i = 0; i < 8; i++) {
   int num = rand ();
   if (num > 0) num *= -1;
   cout << num << endl;
} //for
```

Problem 327 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times (2n-1)$ pattern of $*$ symbols in the shape of a large triangle.

For example, if the user enters 4 for n the program should print the following picture.

```
   *
  * *
 *   *
*******
```

Answer:

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

int main() {
   int c, r, n;
   cout << "Enter a positive integer: ";
   cin >> n;
   while (n <= 0) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }
   for (r = n; r >= 1; r--) {
     for (c = 1; c <= 2 * n - 1; c++)
       if ( r == 1 || c == r || c + r == 2 * n ) cout << "*";
       else cout << " ";
     cout << endl;
   }
   return 0;
}
```

Problem 328 Write a function called *oddDigits* that determines the number of odd digits in an integer parameter. For example, a program that uses the function *oddDigits* follows. (In this example, the number 10 has one odd digit namely 1; the number 26 has no odd digits; the number 19683 has three odd digits namely 1, 9 and 3.)

```
int main() {
   cout << oddDigits(10) << endl;    // prints 1
   cout << oddDigits(26) << endl;    // prints 0
   cout << oddDigits(19683) << endl; // prints 3
   return 0;
}
```

Answer:

```
int oddDigits(int x) {
   if (x == 0) return 0;
   return oddDigits(x/10) + x % 2;
}
```

Problem 329 For each of the following short segments of a program write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
   int x = 4, y = 5;
   if (x <= y && y <= x) cout << "Yes";
   else cout << "No";
```

Answer:

```
No
```

(ii)

```
   int x = 4, y = 5;
   cout << (x / y  + 1.0) << endl;
```

Answer:

```
1
```

(iii)

```
   for (int i = 1; i <= 10; i++) {
      cout << i << endl;
      i++;
   }
```

Answer:

```
1
3
5
7
9
```

(iv)

```
   int x[3][3] = {{1,3,5}, {2,4,6}, {7,8,9}};
   for (int i = 0; i <= 2; i++) for (int j = 0; j <= 2; j++)
      if (i == j) cout << x[i][j];
```

Answer:

```
149
```

(v)

```
int x[3][3] = {{1,3,5}, {2,4,6}, {7,8,9}};
for (int j = 0; j <= 2; j++) for (int i = 0; i <= 2; i++)
   cout << x[i][j];
cout << endl;
```

Answer:

```
127348569
```

Problem 330 Write a complete C++ program that does the following.

1. It asks the user to enter positive integers a and b that are each at most 20.
2. The program generates random integer values between 1 and 6 as the entries in a table with a rows and b columns.
3. The program then prints the table.
4. The program prints a picture with a rows and b columns. The character printed in row i and column j is X or O according as the entry of the table in row i and column j is even or odd.

For example, the following represents one run of the program.

```
Enter integers for r and c (at most 20):     2 2
The table has been generated as:
6 3
1 3
The picture is:
XO
OO
```

Answer:

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

int main ()
{
        int a, b;
        int table [20][20];

        cout << "Give me two integers, each at most 20: ";
        cin >> a >> b;

        for (int r = 0; r < a; r++)
                for (int c = 0; c < b; c++)
                        table [r][c] = rand () % 6 + 1;

        cout << "The table has been generated as:" << endl;
        for (int r = 0; r < a; r++)
        {
                for (int c = 0; c < b; c++)
                        cout << table [r][c] << " ";
                cout << endl;
        } //for

        cout << "The picture is:" << endl;
        for (int r = 0; r < a; r++)
        {
                for (int c = 0; c < b; c++)
```

```
                        if (table [r][c] % 2 == 0) cout << "X";
                        else cout << "O";
                cout << endl;
        } //for

        return 0;
} //main
```

Problem 331 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **firstDigit** which returns the first digit of an integer.

Answer:

```
int firstDigit(int x)
```

(b) A function called **sqrt** that returns the square root of a double precision parameter.

Answer:

```
double sqrt(double x)
```

(c) A function called **oddString** which returns a string made up of the characters in odd position of an input string.

Answer:

```
string oddString(string s)
```

(d) A function called **randomWord** which is to create and return a random word.

Answer:

```
string randomWord()
```

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

```
void sort(string data[], int cap)
```

Problem 332 Consider the following C++ program.

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

int recursive(int n) {
   if (n < 10) return n;
   if (n < 100) return n/10;
   return 10 * recursive(n / 100) + n % 10;
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer:

```
5
```

(b) The user enters 16 for x.

Answer:

```
1
```

(c) The user enters 123 for x.

Answer:

```
13
```

(d) The user enters 1234 for x.

Answer:

```
14
```

(e) The user enters 19683 for x.

Answer:

```
163
```

Problem 333 Write a function called *evens* that deletes all odd digits from a positive integer parameter. For example, a program that uses the function *evens* follows.

```
int main() {
   cout << evens(16) << endl;       // prints 6
   cout << evens(666) << endl;      // prints 666
   cout << evens(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int evens(int n){
   if (n % 2 == 1) return evens(n / 10);
   if (n < 10) return n;
   return 10 * evens(n / 10) + n % 10;
}
```

Problem 334 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the average of the entries for each column of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
 3 -1    4
10 30 -100
 2 -2   99
The averages of the 3 columns are: 5.0 9.0 1.0
```

Answer:

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

int main ()
{
        int num;
        int arr [100] [100];
        cout << "Give a number that's at most 100: ";
        cin >> num;

        cout << "Give me " << num << " rows of " << num << " integers: ";
        for (int r = 0; r < num; r++)
                for (int c = 0; c < num; c++)
                        cin >> arr [r] [c];

        cout << "The averages of the " << num << " columns are:";

        for (int c = 0; c < num; c++)
        {
                int colSum = 0;
                for (int r = 0; r < num; r++)
                    colSum += arr[r][c];
                cout << ((double) colSum) / num << " ";
        } //for

        cout << endl << endl;

        return 0;
} //main
```

Problem 335 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Include declarations for any variable that you use.

(i) Print the remainder when 101 is divided by 17 to the file *out.txt*. **Answer:**

```
ofstream out("out.txt");
out << 101 % 17;
```

(ii) Print a random lower case letter to the screen. (The random letter should be determined by using an appropriate C++ function.) **Answer:**

```
cout << (char) ('a' + rand() % 26);
```

(iii) Read a line of text from the user and print the word *Yes* if it contains the character 7. **Answer:**

```
string input;
getline(cin, input);
if (input.find("7") >= 0) cout << "Yes";
```

(iv) Print the middle character of the string s. (Assume that the string has odd length.) **Answer:**

```
cout << s[s.length() / 2];
```

(v) Swap the values of integer variables called x and y. **Answer:**

```
int temp = y;
y = x;
x = temp;
```

Problem 336 Consider the following C++ program.

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

int recursive(int n) {
   if (n < 10) return n;
   return 100 * recursive(n / 100) + 11* (n % 10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer:

```
5
```

(b) The user enters -10 for x.

Answer:

```
-10
```

(c) The user enters 65 for x.

Answer:

```
55
```

(d) The user enters 123 for x.

Answer:

```
133
```

(e) The user enters 19683 for x.

Answer:

```
16633
```

Problem 337 Write a function called *twoPart* that returns the largest power of 2 that divides a positive integer parameter.

For example, a program that uses the function *twoPart* follows.

```
int main() {
   cout << twoPart(16) << endl;   // prints 16
   cout << twoPart(666) << endl;   // prints 2
   cout << twoPart(777) << endl;   // prints 1
   return 0;
}
```

Answer:

```
int twoPart(int x) {
   if (x % 2 == 1) return 1;
   return 2 * twoPart (x / 2);
}
```

Problem 338 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the maximum entry found for each row of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
3 -1 4
10 30 -100
0 0 0
The maximum entries in the 3 rows are: 4 30 0
```

Answer:

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

int main ()
{
        int num;
        int arr [100] [100];
        cout << "Give a number that's at most 100: ";
        cin >> num;

        cout << "Give me " << num << " rows of " << num << " integers: ";
        for (int r = 0; r < num; r++)
                for (int c = 0; c < num; c++)
                        cin >> arr [r] [c];

        cout << "The maximum entries in the " << num << " rows are:";

        int max;
        for (int i = 0; i < num; i++)
        {
                max = arr [i] [0];
                for (int j = 0; j < num; j++)
                        if (max < arr [i] [j])
                                max = arr [i] [j];
                cout << max << " ";
        } //for

        cout << endl << endl;

        return 0;
} //main
```

Problem 339 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Include declarations for any variable that you use.

(i) Print the word *output* to the file *out.txt*.

Answer:

```
ofstream out("out.txt");
out << "output";
```

(ii) Print a random negative integer to the screen. (The random integer should be determined by using an appropriate C++ function.)

Answer:

```
int r = rand();
while (r == 0) r = rand();
if (r > 0) r = -r;
cout << r;
```

(iii) Read a line of text from the user and print the word *Yes* if it contains at most 7 characters.

Answer:

```
string line;
getline(cin, line);
if (line.length() <= 7) cout << "Yes";
```

(iv) Print the last but one character of the string s.

Answer:

```
cout << s[s.length() - 2];
```

(v) Print the average of integer variables called x and y.

Answer:

```
cout << (x + y) / 2.0;
```

Problem 340 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times (2n-1)$ pattern of $*$ symbols in the shape of a large upside down triangle.

For example, if the user enters 4 for n the program should print the following picture.

```
*******
 *   *
  * *
   *
```

Answer:

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

int main() {
   int c, r, n;
   cout << "Enter a positive integer: ";
   cin >> n;
   while (n <= 0) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }
   for (r = 1; r <= n; r++) {
     for (c = 1; c <= 2 * n - 1; c++)
```

```
      if ( r == 1 || c == r || c + r == 2 * n ) cout << "*";
      else cout << " ";
    cout << endl;
  }
  return 0;
}
```

Problem 341 Write a function called *reverse* that reverses the entries in an array.

For example, a program that uses the function *reverse* follows.

```
int main() {
   int a[5] = {3, 1, 4, 1, 5};
   reverse(a, 5);
   cout << a[0] << a[1] << a[2] << a[3] << a[4];   // prints 51413
   return 0;
}
```

Answer:

```
void reverse(int a[], int cap) {
   for (int i = 0; i < cap / 2; i++) {
      int temp = a[i];
      a[i] = a[cap - 1 - i];
      a[cap - 1 - i] = temp;
   }
}
```

Problem 342 Write a complete C++ program that does the following.

1. It asks the user to enter positive integers r and c that are at most 100.
2. The program reads in a table of integers with r rows and c columns as entered by the user.
3. The program prints out all values of an integer x for which the entries in row x have a sum of 7.

For example, the following represents one run of the program.

```
Enter integers for r and c (at most 100):     3 2
Enter 3 rows of 2 integers:
 3  4
 1  0
 8 -1
The following rows add to 7:  0 2
```

Answer:

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

int main ()
{
        int table[100][100], r, c;
        cout << "Enter integers for r and c (at most 100):   ";
        cin >> r >> c;

        cout << "Enter " << r << " rows of " << c << " integers: ";
        for (int i = 0; i < r; i++)
                for (int j = 0; j < c; j++)
                        cin >> table[i][j];
```

```
        cout << "The following rows add to 7: ";
        for (int x = 0; x < r; x++)
        {
            int rowS = 0;
            for (int i = 0; i < c; i++) rowS += table[x][i];
            if (rowS == 7) cout << x << " ";
        } //for

        cout << endl << endl;

        return 0;
} //main
```

Problem 343 Consider the following C++ program.

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

string recursive(string s) {
   if (s.length() < 3) return s;
   if (s.length() < 5) return "a";
   return recursive(s.substr(3));
}

int main() {
  string x;
  cout << "Enter a string: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters *Hi*

Answer:

```
Hi
```

(b) The user enters *Hello*

Answer:

```
lo
```

(c) The user enters *Goodbye*

Answer:

```
a
```

(d) The user enters *12345678*

Answer:

```
78
```

(e) The user enters *1234 5678*

Answer:

```
a
```

Problem 344 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
int x = 4, y = 5;
cout << ++x + y--;
```

Answer:

```
10
```

(ii)

```
int main(int argc, char *argv[]) {
   cout << argv[1];
```

Answer:

```
input1.txt
```

(iii)

```
   for (int i = 2; i >= 0; i--) {
      for (int j = 0; j < i; j++) cout << "*";
      cout << endl;
   }
```

Answer:

```
**
*
```

(iv)

```
   int c = 4, d = 5;
   c = d;
   d = c;
   cout << c << " " << d;
```

Answer:

```
5 5
```

(v)

```
   for (int i = 2; i >= 0; i--)
      for (int j = 0; j < i; j++) cout << "*";
   cout << endl;
```

Answer:

***

Problem 345 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **firstChar** which returns the first character of a string.

Answer:

```
char firstChar(string s)
```

(b) A function called **power** that returns an integer power of a double precision decimal number.

Answer:

```
double power(double x, int n)
```

(c) A function called **As** which returns the number of times the letter A appears in a string.

Answer:

```
int As(string s)
```

(d) A function called **randomEven** which is to create and return a random even number.

Answer:

```
int randomEven()
```

(e) A function called **inOrder** which is to determine whether an array of strings is in alphabetical order.

Answer:

```
bool inOrder(string s[], int cap)
```

Problem 346 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer.

2. The program reads a value n entered by the user. If the value is not legal, the program repeatedly makes the user type in another value until a legal value of n has been entered.

3. The program prints an $n \times (2n-1)$ pattern of $*$ symbols in the shape of a large letter V.

For example, if the user enters 4 for n the program should print the following picture.

```
*     *
 *   *
  * *
   *
```

Answer:

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

int main() {
   int c, r, n;
   cout << "Enter a positive integer: ";
   cin >> n;
   while (n <= 0) {
      cout << "Illegal.  Try again: ";
      cin >> n;
   }
```

```
   for (r = 1; r <= n; r++) {
     for (c = 1; c <= 2 * n - 1; c++)
       if ( c == r || c + r == 2 * n ) cout << "*";
       else cout << " ";
     cout << endl;
   }
   return 0;
}
```

Problem 347 Write a function called *sort* that sorts three integer parameters into decreasing order.

For example, a program that uses the function *sort* follows.

```
int main() {
   int a = 2, b = 7, c = 1;
   sort(a, b, c);
   cout << a << b << c << endl;   // prints 721
   return 0;
}
```

Answer:

```
void swap(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}

void order(int &a, int &b) {
   if (a < b) swap(a, b);
}

void sort(int &a, int &b, int &c) {
   order(a, b);
   order(a, c);
   order(b, c);
}
```

Problem 348 Write a complete C++ program that does the following.

1. It asks the user to enter positive integers r and c that are at most 100.
2. The program reads in a table of integers with r rows and c columns as entered by the user.
3. The program prints out all values of an integer x for which row x and column x of the table have the same sum.

For example, the following represents one run of the program.

```
Enter integers for r and c (at most 100):     3 2
Enter 3 rows of 2 integers:
 3  2
 1  0
 1  1
The row and column sums are equal at 0.
```

(Note the program prints 0 because row 0 sums to $3 + 2 = 5$ and column 0 sums to $3 + 1 + 1 = 5$.)

Answer:

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

```
int main ()
{
        int table[100][100], r, c;
        cout << "Enter integers for r and c (at most 100):   ";
        cin >> r >> c;

        cout << "Enter " << r << " rows of " << c << " integers: ";
        for (int i = 0; i < r; i++)
                for (int j = 0; j < c; j++)
                        cin >> table[i][j];

        for (int x = 0; x < r && x < c; x++)
        {
            int rowS = 0, colS = 0;
            for (int i = 0; i < c; i++) rowS += table[x][i];
            for (int i = 0; i < r; i++) colS += table[i][x];
            if (rowS == colS)
               cout << "The row and column sums are equal at " << x <<".\n";
        } //for

        cout << endl << endl;

        return 0;
} //main
```

Problem 349 Consider the following C++ program.

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

string recursive(string s) {
   if (s.length() < 3) return s;
   if (s.length() < 6) return "a";
   return recursive(s.substr(4));
}

int main() {
  string x;
  cout << "Enter a string: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters *Hi*

Answer:

```
Hi
```

(b) The user enters *5*

Answer:

```
5
```

(c) The user enters *five*

Answer:

```
a
```

(d) The user enters *string*

Answer:

```
ng
```

(e) The user enters *recursive*

Answer:

```
a
```

Problem 350 Suppose that a C++ program called *prog.cpp* is compiled and correctly executed on venus with the instructions:

```
venus> g++ prog.cpp
venus> a.out input1.txt input2 out.txt
```

For each of the following short segments of the program *prog.cpp* write exactly what output is produced. Each answer should consist of those symbols printed by the given part of the program and nothing else.

(i)

```
int x = 4, y = 5;
if (x < y || y < x) cout << "Yes";
else cout << "No";
```

Answer:

```
Yes
```

(ii)

```
int main(int argc, char *argv[]) {
   cout << argc;
```

Answer:

```
4
```

(iii)

```
for (int i = 2; i < 0; i--) {
   for (int j = 0; j < i; j++) cout << "*";
   cout << endl;
}
```

Answer:

(iv)

```
int c = 4, d = 5;
if (++c < d) cout << "Yes";
else cout << "No";
```

Answer:

```
No
```

(v)

```
string s = "Hello";
for (int i = s.length(); i > 0; i--) {
   for (int j = 0; j < i; j++) cout << (char) s[j];
   cout << endl;
}
```

Answer:

```
Hello
Hell
Hel
He
H
```

Problem 351 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
int x[10], z[10][10], r, c;
string s;
```

(i) Print the remainder when r is divided by c.

```
cout << r % c;
```

(ii) Set r to be a random integer between 1 and 10. (The random integer should be determined by an appropriate C++ function.)

```
r = rand() % 10 + 1;
```

(iii) Print the sum of all 10 entries of the array x.

```
int sum = 0;
for (int i = 0; i < 10; i++) sum += x[i];
cout << sum;
```

(iv) Print the last character of the string s.

```
cout << s[s.size() - 1];
```

(v) Swap row number 0 with row number 4 in the 2-dimensional array z.

```
for (int i = 0; i < 10; i++) {
   int temp = z[0][i];
   z[0][i] = z[4][i];
   z[4][i] = temp;
}
```

Problem 352 Consider the following C++ program.

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

void x1(int a[][6], int n) {
   for (int i = 0; i < 5; i++) cout << a[n][i];
```

```
   cout << endl;
}

void x2(int b[][6], int n) {
   for (int i = 0; i < n; i++)
      cout << b[i][i] << " ";
   x1(b, n);
}

main() {
  int x[6][6], a[6][6], b[6][6];
  for (int i = 0; i < 6; i++) for (int j = 0; j < 6; j++) {
     x[i][j] = i + j;
     a[i][j] = i * j;
     b[i][j] = (i + 1) / (j + 1);
  }
  cout << "Part a: " << x[5][4] << endl;
  cout << "Part b: " << a[5][4] << endl;
  cout << "Part c: "; x1(x, 5);
  cout << "Part d: "; x2(x, 5);
  cout << "Part e: "; x2(b, 3);
  return 0;
}
```

Complete the line of output that begins:

Answer:

```
Part a: 9
Part b: 20
Part c: 56789
Part d: 0 2 4 6 8 56789
Part e: 1 1 1 42110
```

Problem 353 Write a function called *sixCount* that returns a count of the number of digits that are equal to 6 in its positive integer parameter.

For example, a program that uses the function *sixCount* follows.

```
int main() {
   cout << sixCount(16) << endl;   // prints 1
   cout << sixCount(666) << endl;   // prints 3
   cout << sixCount(777) << endl;   // prints 0
   return 0;
}
```

Answer:

```
int sixCount(int x) {
   if (x == 6) return 1;
   if (x < 10) return 0;
   return sixCount(x % 10) + sixCount(x / 10);
}
```

Problem 354 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in an array n integers entered by the user.
3. The program prints the negative entries from the array, in order.
4. The program prints the positive entries from the array in reverse order.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     8
Enter 8 integers:  3 -1 4 -10 17 18 19 -11
-1 -10 -11
19 18 17 4 3
```

Answer:

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

int main() {
   int x[100];
   int count;

   cout << "Enter a positive integer (at most 100):  ";
   cin >> count;
   cout << "Enter " << count << " integers:  ";
   for (int i = 0; i < count; i++) cin >> x[i];

   for (int i = 0; i < count; i++)
      if (x[i] < 0) cout << x[i] << " ";
   cout << endl;
   for (int i = count - 1; i >= 0; i--)
      if (x[i] > 0) cout << x[i] << " ";
   cout << endl;
   return 0;
}
```

Problem 355 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer n.

2. It repeatedly reads n from the user until the supplied value of n is positive.

3. It prints out a large letter X that has height n and width n. The locations of the printed characters should lie on the diagonals of the $n \times n$ square region that the letter occupies.

Here is an example of how the program should work:

```
Give me a positive integer:   7
X     X
 X   X
  X X
   X
  X X
 X   X
X     X
```

Answer:

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

int main() {
   int n;
   cout << "Give me a positive integer: ";
   cin >> n;
```

```
   while (n <= 0) {
      cout << "Enter a POSITIVE integer:  ";
      cin >> n;
   }

   for (int i = 1; i <= n; i++) {
      for (int j = 1; j <= n; j++)
         if ((i == j) || (i + j == n + 1)) cout << "X";
         else cout << " ";
      cout << endl;
   }
   return 0;
}
```

Problem 356 Write C++ statements to carry out the following tasks.
Do not write complete programs, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  string f, l;
```

Declare any other variables that you use.

(i) Write the strings f and l as the first two lines of the file *data.txt*.
Answer:

```
   ofstream out("data.txt");
   out << f << endl << l << endl;
```

(ii) Print the message *Hello Freddy* if the input file *input.txt* begins with the string $Freddy$. Otherwise do nothing.
Answer:

```
   ifstream file("input.txt");
   file >> f;
   if (f == "Freddy") cout << "Hello Freddy" << endl;
```

(iii) Convert the string f to upper case letters and then print it.
Answer:

```
   for (int i = 0; i < f.size(); i++)
      f[i] = toupper(f[i]);
   cout << f << endl;
```

(iv) Print the number of times that the uppercase letter F appears in the string f.
Answer:

```
   int count = 0;
   for (int i = 0; i < f.size(); i++)
      if (f[i] == 'F') count++;
   cout << count << endl;
```

(v) Swap the strings stored in the variables f and l.
Answer:

```
   string temp = f;
   f = l;
   l = temp;
```

Problem 357 Consider the following C++ program.

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

int main(){

    int i;
    string words[4] = {"zero", "one", "two", "three"};

    for (i = 1; i <= 4; i++) cout << words[4 - i] << " ";          // line A
    cout << endl;

    i = 0;
    while( i + 1 < 4){ cout << words[i+1] << " "; i++; }           // line B
    cout << endl;

    for(i = 0; i < words[1].length(); i++) cout << (words[i])[0];  // line C
    cout << endl;

    return 0;
}
```

(a) What is the output from the loop at line A?

Answer:

```
three two one zero
```

(b) What is the output from the loop at line B?

Answer:

```
one two three
```

(c) What is the output from the loop at line C?

Answer:

```
zot
```

Problem 358 Write a function called *thirdDigit*. The function has an integer parameter and returns the third digit in its parameter. If the parameter is less than 100 the function returns 0 because there is no third digit.
For example, a program that uses the function follows.

```
int main() {
   cout << thirdDigit(777) << " " << thirdDigit(2048) << " " << thirdDigit(500125) << endl;
   return 0;
}
```

It should print: 7 4 0

Answer:

```
int thirdDigit(int x) {
   if (x < 100) return 0;
   if (x < 1000) return x % 10;
   return thirdDigit(x/10);
}
```

Problem 359 Write a function called *sixCount* that returns a count of the number of entries that are equal to 6 in a 2-dimensional array with 6 columns. The function should use a parameter to specify the array and parameters for the row count and column count.

For example, a program that uses the function *sixCount* follows.

```
int main() {
   int arr[2][6] = {{6,4,3,1,2,2}, {6,6,5,2,3,6}};  // array has 4 entries of 6
   cout << sixCount(arr, 2, 6) << endl;              // prints 4
   return 0;
}
```

Answer:

```
int sixCount(int a[][6], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
     for (int j = 0; j < c; j++)
        if (a[i][j] == 6) count++;
   return count;
}
```

Problem 360 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer n.
2. If n is not positive, it prints an error message and exits.
3. Otherwise it calculates and prints the product of the digits of n.

Here is an example of how the program should work:

```
Enter a positive integer n:  373
The product of its digits is 63
```

In this example the product is $3 \times 7 \times 3$ which is 63.

Answer:

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

int product(int x) {
   if (x < 10) return x;
   return (x % 10) * product(x/10);
}

int main() {
   int n;
   cout << "Enter a positive integer: ";
   cin >> n;

   if (n <= 0) {
      cout << "That is not POSITIVE." << endl;
      exit(0);
   }

   cout << "The product of its digits is " << product(n) << endl;
   return 0;
}
```

Problem 361 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
int x[10], y[10], z[10][10], r, c;
```

(i) Read 10 integers into the array x.

Answer:

```
for (c = 0; c <= 9; c++) cin >> x[c];
```

(ii) Set all the entries of the array z so that the entry in row r and column c stores the product of r and c.

Answer:

```
for (r = 0; r <= 9; r ++) for (c = 0; c <= 9; c++)
   z[r][c] = r * c;
```

(iii) Print the smallest value in the array x.

Answer:

```
int min = x[0];
for (c = 1; c <= 9; c++) if (x[c] < min) min = x[c];
cout << min;
```

(iv) Print the word *Divides* if r divides exactly into c otherwise do nothing.

Answer:

```
if (c % r == 0) cout << "Divides";
```

(v) Swap each entry of the array x with the correpsonding entry of array y.

Answer:

```
for (c = 0; c <= 9; c++) {
   int temp = x[c];
   x[c] = y[c];
   y[c] = temp;
}
```

Problem 362 Consider the following C++ program.

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

int recursive(int n) {
   if (n < 100) return n%10;
   return 10 * recursive(n / 100) + n % 10;
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters -10 for x.

Answer: 0

(b) The user enters 5 for x.

Answer: 5

(c) The user enters 55 for x.

Answer: 5

(d) The user enters 123 for x.

Answer: 13

(e) The user enters 19683 for x.

Answer: 163

Problem 363 Write a function called *toTen* that calculates how many entries of an array need to be added to make a sum of 10 or more. (Start adding from index 0.)

For example, a program that uses the function *toTen* follows.

```
int main() {
   int x[8] = {5, 3, 1, 6, 10, 1, -30, -100};
   cout << toTen(x, 8) << endl;
   return 0;
}
```

The output from this program would be 4, because the sum of the first 4 entries $5 + 3 + 1 + 6$ is the first sum that exceeds 10.

Answer:

The following function returns an answer of -1 in case no sum of entries in the array reaches a value of 10. Exam solutions are not required to deal with this possibility.

```
int toTen(int x[], int c) {
  int sum = x[0];
  int col = 1;
  while (sum < 10 && col < c) {
    sum = sum + x[col];
    col ++;
  }
  if (sum < 10) return -1;
  return col;
}
```

Problem 364 Write a complete C++ program that does the following.

1. It asks the user to enter their name as a string *name*.
2. The program reads the name entered by the user.
3. The program converts all letters in the name to uppercase and prints the name.
4. The program prints the uppercase characters of the name in reverse.

For example, the following represents one run of the program.

```
What is your name:     Freddy
FREDDY
YDDERF
```

Answer:

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

int main() {
   string name;
   cout << "What is your name: ";
   getline(cin, name);
   for (int i = 0; i < name.size(); i++)
      name[i] = toupper(name[i]);
   cout << name << endl;
   for (int i = name.size() - 1; i >= 0; i--)
     cout << name[i];
   cout << endl;
   return 0;
}
```

Problem 365 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n.
2. It reads n from the user and exits if n is not positive.
3. It prints out an $n \times n$ checkerboard pattern made from the characters X and O.

Here is an example of how the program should work:

```
Give me a positive integer:   3
XOX
OXO
XOX
```

In a checkerboard pattern, the horizontal and vertical neighbors of each X are Os, and the horizontal and vertical neighbors of each O are Xs.

Answer:

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

int main() {
   int n;
   cout << "Give me a positive integer: ";
   cin >> n;

   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++)
         if (((i + j) % 2) == 0) cout << "X";
         else cout << "O";
      cout << endl;
   }
   return 0;
}
```

Problem 366 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  string f, l, name;
```

Declare any other variables that you use.

(i) From the input file *data.txt*, read a first name to f and a last name to l.

Answer:

```
ifstream file("data.txt");
file >> f >> l;
```

(ii) Print the second character in f to an output file *output.txt*.

Answer:

```
ofstream out("output.txt");
out << f[1] << endl;
```

(iii) Convert the string f to lower case letters and then print it.

Answer:

```
for (int i = 0; i < f.size(); i++)
   f[i] = tolower(f[i]);
cout << f << endl;
```

(iv) Check whether the string f contains the letters *Fred* as a substring. If it does, print the message *Hello Freddy* . Otherwise do nothing.

Answer:

```
if (f.find("Fred") >= 0)
   cout << "Hello Freddy" << endl;
```

(v) Concatenate the strings f and l separated by a space into the string *name*.

Answer:

```
name = f + " " + l;
```

Problem 367 Consider the following C++ program.

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

void mystery(int x[][4], int a, int b, int k) {
  for (int r = 0; r <= a; r++) for (int c = 0; c <= b; c++)
      x[r][c] = k;
}

void print(int x[][4], int s) {
  for (int r = 0; r < s; r++) {
     for (int c = 0; c < s; c++) cout << x[r][c];
     cout << endl;
  }
  cout << endl;
}

int main() {
  int x[4][4];
  mystery(x, 3, 3, 0); print(x, 4);
  mystery(x, 1, 2, 1); print(x, 4);
  mystery(x, 3, 1, 2); print(x, 3);
  mystery(x, 3, 2, 3); print(x, 1);
  return 0;
}
```

(a) What is the output from the first call to the function print?

Answer:

```
0000
0000
0000
0000
```

(b) What is the output from the second call to the function print?

Answer:

```
1110
1110
0000
0000
```

(c) What is the output from the third call to the function print?

Answer:

```
221
221
220
```

(d) What is the output from the fourth call to the function print?

Answer:

```
3
```

Problem 368 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **lastChar** which uses a string as input and returns the last character in the string.

Answer:

```
char lastChar(string x)
```

(b) A function called **isSquare** that tests whether an integer is a perfect square. (For example, 16 is a perfect square, but -5 is not.)

Answer:

```
bool isSquare(int x)
```

(c) A function called **addTwo** which uses as input an array of integers. The task of the function is to add 2 to every element in the array.

Answer:

```
void addTwo(int a[], int capacity)
```

(d) A function called **exchangeArrays** which uses two arrays of integers that have the same capacity and exchanges the entries between them.

Answer:

```
void exchangeArrays(int a[], int b[], int capacity)
```

(e) A function called **exchange** which exchanges the values of two integers.

Answer:

```
void exchange(int &x, int &y)
```

Problem 369 Write a function called *sevenUp*. The function has an integer parameter and calculates an answer by turning any digit equal to 7 in the input to an 8.

For example, a program that uses the function follows.

```
int main() {
   cout << sevenUp(777) << " " << sevenUp(471) << " " << sevenUp(50) << endl;
   return 0;
}
```

It should print: 888 481 50

Answer:

```
int sevenUp(int x) {
   if (x == 7) return 8;
   if (x < 10) return x;
   return 10*sevenUp(x / 10) + sevenUp(x % 10);
}
```

Problem 370 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter 9 integers as the entries of a 3×3 table.
2. The program reads the 9 entries, row by row and prints the table.
3. If every row and column of the table have the same sum then the program adds the message: $MAGIC$.

Here is an example of how the program should work:

```
Enter 9 entries of a 3 x 3 table:  10 14 18  15 16 11  17 12 13

10 14 18
15 16 11
17 12 13

MAGIC
```

This example is magic because each row and each column has a sum of 42.

Answer:

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

int main() {
   int table[3][3];
   cout << "Enter 9 entries of a 3 x 3 table: ";

   int r, c;
   for (r = 0; r < 3; r++) for (c = 0; c < 3; c++)
      cin >> table[r][c];

   for (r = 0; r < 3; r++) {
      cout << endl;
      for (c = 0; c < 3; c++) cout << table[r][c] << " ";
   }
   cout << endl;

   int sum = table[0][0] + table[0][1] + table[0][2];
```

```
   bool isMagic = true;
   for (int i = 0; i < 3; i++) {
      int rowSum = 0, colSum = 0;
      for (int j = 0; j < 3; j++) {
         rowSum += table[i][j];
         colSum += table[j][i];
      }
      if (sum != rowSum || sum != colSum) isMagic = false;
   }
   if (isMagic) cout << "MAGIC" << endl;
   return 0;
}
```

Problem 371 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sumDigits** which returns the sum of the digits of an integer.

Answer:

```
int sumDigits(int x)
```

(b) A function called **isSmall** that returns an answer of true if a double precision parameter has a value between 0 and 0.001. (It returns false otherwise.)

Answer:

```
bool isSmall(double x)
```

(c) A function called **randomLetter** which generates and returns a random letter of the alphabet. (The output is to be a single character between 'A' and 'Z'.)

Answer:

```
char randomLetter()
```

(d) A function called **sort3** which is to change a collection of three input values so that they appear in increasing order.

Answer:

```
void sort3(int &x, int &y, int &z)
```

(e) A function called **total** which is to determine the sum of all the entries in an array.

Answer:

```
int total(int x[], int capacity)
```

Problem 372 Consider the following C++ program.

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

int recursive(int n) {
   if (n < 10) return n;
   return n % 10 - recursive(n/10);
}

main() {
  int x;
  cout << "Enter a positive integer: ";
  cin >> x;
  if (x <= 0) cout << "Error" << endl;
  else cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 0 for x.

Answer:

```
Error
```

(b) The user enters 5 for x.

Answer:

```
5
```

(c) The user enters 55 for x.

Answer:

```
0
```

(d) The user enters 555 for x.

Answer:

```
5
```

(e) The user enters 19683 for x.

Answer:

```
-7
```

Problem 373 Write a function called *quadratic* that calculates the value of a quadratic function $ax^2 + bx + c$. For example, a program that uses the function *quadratic* follows.

```
int main() {
   double a = 1.0, b = 2.2, c = 1.21, x = 0.1;
   cout << quadratic(a, b, c, x) << endl;
   return 0;
}
```

Answer:

```
double quadratic(double a, double b, double c, double x) {
   return c + b * x + a * x * x;
}
```

Problem 374 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, n.

2. The program reads a value entered by the user. If the value is not positive, the program should terminate.

3. The program should consider every number x between 1 and n and print out any value of x that divides exactly into n.

The printed values should all appear on a single line, separated by spaces.

For example, the following represents one run of the program. (The user chooses the number 28.)

```
Enter a positive integer:     28
1 2 4 7 14 28
```

Answer:

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

int main() {
   int n, x;

   cout << "Enter a positive integer value for n: ";
   cin >> n;
   if (n <= 0) exit(1);

   for (x = 1; x <= n; x++)
     if (n % x == 0) cout << x << " ";

   cout << endl;
   return 0;
}
```

Problem 375 Write a complete C++ program that does the following.

1. It asks the user to enter some positive integers.
2. It reads positive integers from the user.
3. As soon as the user enters a non-positive integer, the program stops reading.
4. The program reports the sum of all the positive numbers that it read.

Here is an example of how the program should work:

```
Give me some positive integers:    1 12 1 100 -1000
sum: 114
```

Answer:

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

int main() {
   int sum = 0;
   int n = 1;
   cout << "Give me some positive integers: ";
   while (n > 0) {
     cin >> n;
     if (n > 0) sum += n;
   }
   cout << "sum: " << sum << endl;
   return 0;
}
```

Problem 376 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  string f, l;
```

(i) Read a first name to f and a last name to l. Then, print out the string f followed by the string l on another line.

Answer:

```
cin >> f >> l;
cout << f << endl << l << endl;
```

(ii) Print the second character in f.

Answer:

```
cout << f[1];
```

(iii) Convert the string f to upper case letters and then print it.

Answer:

```
for (int i = 0; i < f.size(); i++) f[i] = toupper(f[i]);
cout << f;
```

(iv) Read a word into f from a user. If the program can find the smaller string "reddy" within the string f, print the word "Hello", otherwise do nothing.

Answer:

```
cin >> f;
if (f.find("reddy") >= 0) cout << "Hello";
```

(v) Print the last character of l.

Answer:

```
cout << l[l.size() - 1];
```

Problem 377 Consider the following C++ program.

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

void mystery(char x[][4], int a, int b, char k) {
  for (int r = a; r <= b; r++) for (int c = a; c <= b; c++)
      x[r][c] = k;
}

void print(char x[][4], int s) {
  for (int r = 0; r < s; r++) {
     for (int c = 0; c < s; c++) cout << x[r][c];
     cout << endl;
  }
  cout << endl;
}

int main() {
  char x[4][4];
  mystery(x, 0, 3, 'X'); print(x, 4);
  mystery(x, 1, 2, 'Y'); print(x, 4);
  mystery(x, 2, 3, 'Z'); print(x, 4);
  mystery(x, 3, 2, '0'); print(x, 4);
  return 0;
}
```

(a) What is the output from the first call to the function print?

Answer:

```
XXXX
XXXX
XXXX
XXXX
```

(b) What is the output from the second call to the function print?

Answer:

```
XXXX
XYYX
XYYX
XXXX
```

(c) What is the output from the third call to the function print?

Answer:

```
XXXX
XYYX
XYZZ
XXZZ
```

(d) What is the output from the fourth call to the function print?

Answer:

```
XXXX
XYYX
XYZZ
XXZZ
```

Problem 378 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **isPrime** that tests whether an integer is prime. (For example, 7 is prime, but 9 is not.)

Answer:

```
bool isPrime(int x)
```

(b) A function called **firstChar** which uses a string as input and returns the first character in the string.

Answer:

```
char firstChar(string x)
```

(c) A function called **printThree** which uses as input an array of integers. The task of the function is to print the first three elements of the array.

Answer:

```
void printThree(int x[])
```

(d) A function called **printChess** which uses as input an 8×8 array of characters that represents a chess board. The task of the fuction is to print the board to output.

Answer:

```
void printChess(char x[][8], int r, int c)
```

(e) A function called **reverseWord** which is to use a string parameter and change it to become the string obtained by reversing its letters. (For example, an input string *was* would be changed to *saw*.)

Answer:

```
void reverseWord(string &x)
```

Problem 379 Write a function called *biggestEntry* that uses a two dimensional array (with 3 columns) with integer entries as its first parameter. It also uses parameters representing the row and column capacities. The function should return the value of the biggest entry in the array.

For example, a program that uses the function follows.

```
int main() {
   int x[2][3] = {{1,2,3},{4,7,3}};
   cout << biggestEntry(x, 2, 3) << endl;
   return 0;
}
```

It should print 7 (since 7 is the biggest entry in the array).

Answer:

```
int biggestEntry(int a[][3], int r, int c) {
   int answer = a[0][0];
   for (int i = 0; i < r; i++) for (int j = 0; j < c; j++)
      if (a[i][j] > answer) answer = a[i][j];
   return answer;
}
```

Problem 380 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer value, n.
2. The program reads a value entered by the user. If n is not positive, the program should exit.
3. It prints out the number of digits in n.
4. It prints the number digits in the binary representation of n.

Here is an example of how the program should work:

```
Enter a positive integer n: 17
Digits in n: 2
Binary digits in n: 5
```

The number of binary digits is 5 because the binary representation of 17 is 10001. However, it is not necessary for your program to determine this binary representation.

Answer:

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

int length(int x, int base) {
   if (x < base) return 1;
   return 1 + length(x / base, base);
}

int main() {
   int n;
   cout << "Enter a positive integer n: ";
   cin >> n;
   if (n <= 0) exit(1);
   cout << "Digits in n: " << length(n, 10) << endl;
   cout << "Binary digits in n: " << length(n, 2) << endl;
   return 0;
}
```

Problem 381 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sum** which returns the sum of 4 double precision values.

Answer:

```
double sum(double a, double b, double c, double d)
```

(b) A function called **midDigit** that is used to return the middle digit of an integer.

Answer:

```
int midDigit(int x)
```

(c) A function called **isPositive** which is to return an answer of true if the sum of the entries of an array of double precision data is positive (and return false otherwise).

Answer:

```
bool isPositive(double x[], int capacity)
```

(d) A function called **average2DArray** which is to print (to cout) the average of the entries in a 2-dimensional array (the array stores integers and has 10 rows and 15 columns).

Answer:

```
void average2DArray(int array[][15], int rows, int cols)
```

(e) A function called **makeZero** which is to use two integer input variables and change their values to zero. (After the function ends, the input variables must be zero.)

Answer:

```
void makeZero(int &x, int &y)
```

Problem 382 Consider the following C++ program.

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

void mystery(int n) {
   cout << n % 100;
   if (n < 1000) return;
   mystery(n/10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  mystery(x);
  cout << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer: 5

(b) The user enters 512 for x.

Answer: 12

(c) The user enters 4370 for x.

Answer: 7037

(d) The user enters 175560 for x.

Answer: 60565575

Problem 383 Write a function called *sum2D* that returns the sum of all elements in a 2-dimensional array that has 4 columns of integer entries.

For example, a program that uses the function *sum2D* follows.

```
int main() {
   int array[3][4] = {{1,2,3,4},{1,2,3,4},{1,2,3,4}};
   cout << sum2D(array, 3, 4) << endl;
   return 0;
}
```

The input values 3 and 4 specify the number of rows and columns in the array. The program should print an answer of 30 (since this is the sum of $1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3,$ and 4).

Answer:

```
int sum2D(int a[][4], int r, int c) {
   int ans = 0;
   for (int row = 0; row < r; row++)
     for (int col = 0; col < c; col++)
       ans += a[row][col];
   return ans;
}
```

Problem 384 Write a complete C++ program that does the following.

1. It asks the user to enter a 5-digit integer value, n.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program calulates and stores the 5 individual digits of n.
4. The program outputs a "bar code" made of 5 lines of stars that represent the digits of the number n.

For example, the following represents one run of the program. (The user chooses the number 16384.)

```
Enter a 5 digit integer:     16384
*
******
***
********
****
```

Answer:

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

int bar(int l) {
   for (int c = 0; c < l; c++) cout << "*";
   cout << endl;
}

int main() {
   int i, n;
   int digit[5];

   cout << "Enter a 5 digit integer: ";
   cin >> n;
   if (n < 10000 || n > 99999) exit(0);

   for (i = 0; i < 5; i++) {
      digit[i] = n % 10;
```

```
      n = n / 10;
   }

   for (i = 4; i >= 0; i--) bar(digit[i]);
   return 0;
}
```

Here is an alternative solution that is shorter, but makes use of recursion:

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

void bars(int n) {
   if (n == 0) return;
   bars(n/10);
   for (int c = 0; c < n % 10; c++) cout << "*";
   cout << endl;
}

int main() {
   int n;
   cout << "Enter a 5 digit integer: ";
   cin >> n;
   if (n < 10000 || n > 99999) exit(0);
   bars(n);
   return 0;
}
```

Problem 385 Write a complete C++ program that does the following.

1. It asks the user to enter 5 single digit positive integers.
2. If any number is out of range, it says: "That is too hard."
3. Otherwise it adds the numbers and prints their sum.

Here is an example of how the program should work:

```
Give me 5 single digit positive integers:    9 9 9 6 9
42
```

Answer:

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

int main() {
   int answer = 0, x;
   cout << "Give me 5 single digit positive integers: ";
   for (int i = 1; i <= 5; i++) {
      cin >> x;
      if (x <= 0 || x >= 10) {
         cout << "That is too hard." << endl;
         exit(0);
      }
      answer += x;
   }
   cout << answer << endl;
   return 0;
}
```

Problem 386 Write C++ statements to carry out the following tasks. Do not write complete programs, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
int x;
string f, l;
```

(i) Read a user's first name to f and their last name to l.

Answer:

```
cin >> f >> l;
```

(ii) Print out the string f followed by the string l with a space between them.

Answer:

```
cout << f << " " << l;
```

(iii) Set x to be $1 - 2 + 3 - 4 + 5 - \ldots + 999$. The formula involves all integers from 1 to 999. Odd numbers are added, even numbers subtracted.

Answer:

```
x = 0;
for (int i = 1; i < 1000; i++)
   if (i % 2 == 0) x -= i;
   else x += i;
```

(iv) Repeatedly double x, until the value of x exceeds 1024.

Answer:

```
while (x <= 1024) x *= 2;
```

(v) Read a word into f from a user. If the word is "Freddy", print output saying "Hello", otherwise do nothing.

Answer:

```
cin >> f;
if (f == "Freddy") cout << "Hello" << endl;
```

Problem 387 Consider the following C++ program.

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

void mystery(string array[], int p[], int q) {
   if (q < 0) cout << "Help!" << endl;
   else if (q <= 2) cout << p[q] << endl;
   if (q > 2) {
     for (int i = 0; i <= q; i++) cout << array[p[i]] << " ";
     cout << endl;
   }
}

int main() {
  string x[5] = {"This", "is", "a", "dumb", "question"};
  int a[10] = {0, 4, 1, 3, 3, 3, 2, 2, 2, 2};
  mystery(x, a, -10);
  mystery(x, a, 0);
  mystery(x, a, 1);
  mystery(x, a, 3);
  mystery(x, a, 5);
  return 0;
}
```

(a) What is the output from the first call to the function mystery?

Answer:

```
Help!
```

(b) What is the output from the second call to the function mystery?

Answer:

```
0
```

(c) What is the output from the third call to the function mystery?

Answer:

```
4
```

(d) What is the output from the fourth call to the function mystery?

Answer:

```
This question is dumb
```

(e) What is the output from the fifth call to the function mystery?

Answer:

```
This question is dumb dumb dumb
```

Problem 388 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **isLeapYear** that tests whether an integer represents a leap year. (For example, 2008 is a leap year, but 2007 is not.)

Answer:

```
bool isLeapYear(int y)
```

(b) A function called **temperatureDifference** which uses as input two double precision values that represent the temperature in New York measured in degrees Fahrenheit and the temperature in Paris measured in degrees Celsius. The function is to calculate and return the difference between the temperatures in degrees Fahrenheit.

Answer:

```
double temperatureDifference(double n, double p)
```

(c) A function called **addCurve** which uses as input an array of integer test scores. The task of the function is to add 10 to every score in the array.

Answer:

```
void addCurve(int s[], int capacity)
```

(d) A function called **printTicTacToe** which uses as input a 3×3 array of characters that represents a Tic-Tac-Toe game. The task of the fuction is to print the board to output.

Answer:

```
void printTicTacToe(char [][3])
```

(e) A function called **reverseDigits** which is to use an integer parameter and return the integer obtained by reversing the digits in the parameter.

Answer:

```
int reverseDigits(int x)
```

Problem 389 Write a function called *biggestDigit* that uses an integer input parameter and returns the largest digit in the input. The input should be assumed to be positive.

For example, a program that uses the function follows.

```
int main() {
   cout << biggestDigit(1760) << endl;
   return 0;
}
```

It should print 7 (since 7 is the biggest digit in 1760).

A little extra credit will be given for good recursive solutions.

Answer:

```
int biggestDigit(int x) {
   if (x < 10) return x;
   int b = biggestDigit(x/10);
   if (x % 10 > b) return x % 10;
   return b;
}
```

Problem 390 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer value, n that is at most 100.

2. The program reads a value entered by the user. If n is not positive, or n is greater than 100, the program should exit.

3. It prints out all numbers between 1 and 1000 for which the sum of the digits is exactly n.

For example, if the user chooses 13 for n, the program should print out 49 because $4 + 9 = 13$. It would also print 58, 67, and other numbers with the same digit sum. It would not print 48 or 50.

(Suggestion: It might be convenient to write a function called *digitSum*.)

Answer:

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

int digitSum(int x) {
   if (x < 10) return x;
   return x % 10 + digitSum(x/10);
}

int main() {
   int n;
   cout << "Enter a value of n that is at most 100:";
   cin >> n;
   if (n <= 0 || n > 100) exit(0);

   for (int x = 1; x <= 1000; x++)
     if (digitSum(x) == n) cout << x << " ";
   cout << endl;
   return 0;
}
```

Problem 391 Write a complete C++ program that does the following.

1. It asks the user to enter a (single) first name.

2. The program stores the name, but if it is "Freddy", the program changes it to "you".

3. The program says hello to the user, using their name (or changed version).

Here is an example of how the program should work:

```
Who are you?    Max
Hello Max.
```

Answer:

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

int main() {
   string name;
   cout << "Who are you?   ";
   cin >> name;
   if (name == "Freddy") name = "you";
   cout << "Hello " << name << "." << endl;
   return 0;
}
```

Problem 392 Write C++ statements to carry out the following tasks. Do not write complete programs, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  int x;
  string s;
```

(i) Read a user's first name to s and their age to x.

Answer:

```
cout << "Enter your name and age: ";
cin >> s >> x;
```

(ii) Print out the number of characters in the string s.

Answer:

```
cout << s.size();
```

(iii) Set x to be $1^3 + 2^3 + \ldots + 71^3$, the sum of the cubes of the numbers from 1 to 71.

Answer:

```
x = 0;
for (int i = 1; i <= 71; i++)
   x += i * i * i;
```

(iv) Repeatedly generate and add a random value between 1 and 6 to x, until the value of x exceeds 100.

Answer:

```
x = 0;
while (x <= 100)
   x += (rand() % 6 + 1);
```

(v) Read a complete line of text into s from a user. If their text includes a substring "Queens", print output saying "College", otherwise do nothing.

Answer:

```
getline(cin, s);
if (s.find("Queens", 0) >= 0) cout << "College";
```

Problem 393 Consider the following C++ program.

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

void mystery(int &p, int q) {
  int temp = p;
  p = q;
  q = temp;
}

int main() {
  int p, q;
  for (p = 0; p < 5; p++) cout << p; cout << endl;
  for (q = 0; q < 5; ++q) cout << q;
  cout << endl;
  for (p = 3; p < 6; p++)
    for (q = 1; q <= 3; q++)
       cout << p - q; cout << endl;
  p = 4; q = 14;
  mystery(q, p);
  cout << p << " " << q << endl;
  p = 4; q = 14;
  cout << ++p - q-- << endl;
  return 0;
}
```

What is the output from the program?

Answer:

```
01234
01234
210321432
4 4
-9
```

Problem 394 Write header lines (prototypes) for the following functions. Do not attempt to supply the blocks for the functions.

(a) A function called **numberDigits** that is to return the number of digits of an integer.

Answer:

```
int numberDigits(int x)
```

(b) A function called **differenceMax** which is to return the difference between the maximum entries in two arrays of integers. (Do not assume that the arrays have the same capacities.)

Answer:

```
int differenceMax(int a[], int capA, int b[], int capB)
```

(c) A function called **swap** which is used to swap two values of type double.

Answer:

```
void swap(double &x, double &y)
```

(d) A function called **firstCharacter** which is to return the first character in a string.

Answer:

```
char firstCharacter(string s)
```

(e) A function called **median** which is to return the median (middle valued) entry in an array that holds an odd number of integer entries.

Answer:

```
int median(int a[], int cap)
```

Problem 395 Write a function called *plusTax* that uses parameters that specify a price (in cents) and a tax rate (as a percentage). The function calculates the amount of tax, rounded to the nearest cent. (Half cents must round up.) It adds the tax to the price and returns the result.

For example, a program that uses the function follows.

```
int main() {
   int cost = 100;        // cost is 100 cents
   double taxRate = 4.8;  // tax is at 4.8 percent
   cout << "With tax that is " << plusTax(cost, taxRate) << " cents." << endl;
   return 0;
}
```

It should find a tax of 4.8 cents, round up to 5 cents and print:

```
With tax that is 105 cents.
```

Answer:

```
int plusTax(int price, double rate) {
   double tax = price * rate / 100;
   int rounded = (int) (tax + 0.5);
   return price + rounded;
}
```

Problem 396 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer value, n that is at most 100.

2. The program reads a value entered by the user. If n is not positive, or n is greater than 100, the program should exit.

3. The program reads n integers from the user and then prints their last digits in reverse order of input.

For example, a run of the program might be as follows:

```
What is n?   7
Enter 7 numbers:    143 259 63 17 12 8 9
9 8 2 7 3 9 3
```

Answer:

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

int main() {
   int i, n;
   int numbers[100];
   cout << "What is n?  ";
   cin >> n;
   if (n <= 0 || n > 100) exit(1);

   cout << "Enter " << n << " numbers: ";
   for (i = 0; i < n; i++) cin >> numbers[i];
   for (i = n - 1; i >= 0; i--) cout << numbers[i] % 10 << " ";
   cout << endl;
   return 0;
}
```

Problem 397 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **lastDigit** that is used to find the last digit of an integer.

Answer:

```
int lastDigit(int x)
```

(b) A function called **average** which determines the average of 3 integer values.

Answer:

```
double average(int x, int y, int z)
```

(c) A function called **largest** which is used to find the largest value in an array of double precision data.

Answer:

```
double largest(double array[], int cap)
```

(d) A function called **print2DArray** which is to print out all of the data in a 2-dimensional array of integers(the array has 100 columns).

Answer:

```
void print2DArray(int array[][100], int rows, int cols)
```

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

```
void sort(string array[], int cap)
```

Problem 398 Consider the following C++ program.

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

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int p, int q) {
  int temp = p;
  q = p;
  p = temp;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(scores, 3);
  print(quiz, 4);
  mystery(scores, 1, 2);
  print(scores, 5);
  for (int i = 0; i < 3; i++)
     m2(quiz[i], quiz[i+ 1]);
  print(quiz, 6);
}
```

What is the output from the program?

Answer:

```
3 1 4
0 1 2 3
3 4 4 1 5
0 1 2 3 4 5
```

Problem 399 Write a function called *countChange* that uses four parameters q, d, n, and p and converts the value of q auarters, d dimes, n nickels, and p cents into dollars.

For example, a program that uses the function *countChange* follows.

```
int main() {
   int q = 10, d = 5, n = 1, p = 2;
   double x = countChange(q, d, n, p);
   cout << "You have $" << x << endl;
}
```

It should print:

```
You have $3.07
```

Answer:

```
double countChange(int quarters, int dimes, int nickels, int pennies) {
   return quarters * 0.25 + dimes * 0.1 + nickels * 0.05 + pennies * 0.01;
}
```

Problem 400 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, r that is at most 100.

2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.

3. The program reads and stores r integers from the user and then prints a pattern of r rows of stars, the lengths of which are the other integers entered by the user.

For example, the following represents one run of the program.

```
How many rows? 4
Enter 4 row lengths:  2 7 1 5
**
*******
*
*****
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int arr[100];
   int r, i, j;

   cout << "How many rows? ";
   cin >> r;
   if (r < 1 || r > 100) exit(1);

   cout << "Enter " << r << " row lengths: ";
```

```
   for (i = 0; i < r; i++) cin >> arr[i];

   for (i = 0; i < r; i++) {
      for (j = 0; j < arr[i]; j++) cout << "*";
      cout << endl;
   }

   return 0;
}
```

Problem 401 Write a complete C++ program that first asks a user to do a simple math problem of your choosing. The user enters an answer and the program grades it as right or wrong.

For example the program might ask about 6×9 and respond to an incorrect answer of 42 as follows:

```
What is 6 x 9?
   42
Wrong!
```

Your program can always ask the same question. **Answer:**

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

int main() {
   int x;
   cout << "What is 6 x 9? ";
   cin >>  x;
   if (x != 54) cout << "Wrong!" << endl;
   else cout << "Right!" << endl;
}
```

Problem 402 Write a complete C++ program that asks a user to enter the prices of 100 different grocery items (each price as a decimal showing dollars and cents). The program calulates and prints the total cost of the items.

Answer:

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

int main() {
   double prices[100];
   cout << "Enter 100 item costs: " << endl;
   for (int i= 0; i < 5; i++) {
      cin >> prices[i];
   }
   double total = 0.0;
   for (int i = 0; i < 5; i++)
      total += prices[i];
   cout << "The total cost is: $" << total << endl;
}
```

Problem 403

Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer value, x.

2. The program reads a value entered by the user. If the value is not positive, the program repeatedly makes the user type in another value until a positive value of x has been entered. (Note positive means greater than 0.)

3. The program prints out x squares on top of each other, the first with size 1, the second with size 2, and so on.

For example, if the user enters 3 for x the program should print:

```
*

**
**

***
***
***
```

Answer:

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

void square(int s) {
  for (int row = 1; row <= s; row++) {
     for (int col = 1; col <= s; col++)
        cout << "*";
     cout << endl;
  }
  cout << endl;
}

int main() {
   int x;
   cout << "What is x? ";
   cin >> x;

   while (x <= 0) {
      cout << "Please give a positive value for x: ";
      cin >> x;
   }

   for (int i = 1; i <= x; i++) square(i);
}
```

Problem 404 Write a function called *percent* that uses two parameters x and y and returns the ratio x/y as a percentage.

For example, a program that uses the function percent follows.

```
int main() {
   double z;
   z = percent(1.5, 3.0);
   cout << z << endl;
}
```

It should print:

```
50.0
```

because $1.5/3 = 1/2 = 50\%$.

Answer:

```
double percent(double a, double b) {
     return 100 * a / b;
}
```

Problem 405 Write a C++ function called *range* that returns the difference between the largest and smallest elements in an array.

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[6] = {11, 12, 11, 3, 12, 13};
   int x;
   x = range (data, 6);
     //   data is the array to search, 6 is the number of elements of the array
   cout << "The range is: " << x  << endl;
}
```

Answer:

```
int range(int d[], int c) {
  int min = d[0];
  int max = d[0];
  for (int i = 1; i < c; i++) {
     if (d[i] < min) min = d[i];
     if (d[i] > max) max = d[i];
  }
  return max - min;
}
```

Problem 406 Consider the following C++ program.

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

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = 0;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(quiz, 4);
  print(scores, 4);
  mystery(scores, 3, 4);
  print(scores, 8);
  for (int i = 0; i < 3; i++)
     mystery(quiz, i, i+ 1);
  print(quiz, 7);
}
```

What is the output from the program?

Answer:

```
0 1 2 3
3 1 4 1
3 1 4 5 0 9 2 6
1 2 3 0 4 5 6
```

Problem 407 Write C++ functions called *elementSwap* and *swap* that swap either the values of two elements of an array or the values of two variables.

It should be possible to use your function in the following program. (The output from this program is: 4 3 because the values of x and y are exchanged.)

```
main() {
   int a[6] = {11, 12, 11, 3, 12, 13};
   int x = 3, y = 4;
   elementSwap(a, 0, 5);
   swap(x, y);
   cout << x  << " " << y << endl;
}
```

Answer:

```
void elementSwap(int a[], int x, int y) {
  int temp = a[x];
  a[x] = a[y];
  a[y] = temp;
}

void swap(int &x, int &y) {
   int temp = x;
   x = y;
   y = temp;
}
```

Problem 408 Write a complete C++ program that asks a user to enter the 10 quiz scores for each student in a class of 30 students. For each of the 10 quizzes, the program decides which student(s) have got the highest scores and prints their numbers. (Hint: Store quiz data in a table.)

Sample output might look like:

```
Top Scores:

Quiz 0:  Students: 5 17 23
Quiz 1:  Students: 2 11 17 26
Quiz 2:  Students: 2 17 23 26 27

and so on....
```

Answer:

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

void topScores(int quiz[][10], int n, int q) {
   int max = quiz[0][q];
   int s;
   for (s = 1; s < n; s++)
     if (quiz[s][q] > max) max = quiz[s][q];
   cout << "Quiz " << q << ":  Students: ";
   for (s = 0; s < n; s++)
     if (quiz[s][q] == max) cout << s << " ";
   cout << endl;
}

int main() {
```

```
   int quiz[30][10];
   int s, q;

   for (s = 0; s < 30; s++) {
      cout << "Enter 10 quiz scores for student " << s << " : ";
      for (q = 0; q < 10; q++)
         cin >> quiz[s][q];
   }

   for (q = 0; q < 10; q++) topScores(quiz, 30, q);
   return 0;
}
```

Problem 409 Consider the following C++ program. What is the output?

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

main() {
  int i = 1, j = 1, k = 1;
  while (i < 10)
     cout << i++;
  cout << endl;
  while (j < 10)
     cout << ++j;
  cout << endl;
  while (++k < 10)
     cout << k++;
  cout << endl;

  return 0;
}
```

Answer:

```
123456789
2345678910
2468
```

Problem 410 Write a C++ program that asks a user how many times it should say hello and then says hello the required number of times. For example, a run of the program might produce the following output:

```
How many hellos do you want: 6
Hello Hello Hello Hello Hello Hello
```

Answer:

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

int main() {
   int n;
   cout << " How many hellos do you want: ";
   cin >> n;
   for (int c = 1; c <= n; c++) cout << "Hello ";
   cout << endl;
   return 0;
}
```

Problem 411 Two numbers are considered as very different if they differ by more than 10. Write a C++ function called areVeryDifferent that determines whether two integers are very different.

For example, your function could be used in the following program.

```
int main() {
   int x = 4, y = 10, z = -4;
   if (areVeryDifferent(x, y)) cout << "x and y are very different" << endl;
   if (areVeryDifferent(x, z)) cout << "x and z are very different" << endl;
   if (areVeryDifferent(y, z)) cout << "y and z are very different" << endl;
   return 0;
}
```

The output from this program would be:

```
y and z are very different
```

Answer:

```
bool areVeryDifferent(int x, int y) {
   if ((x - y) > 10 || (y - x) > 10) return true;
   return false;
}
```

Problem 412 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, x that is at most 100.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program reads and stores x words from the user and then prints them in reverse order.

For example, the following represents one run of the program.

```
How many words? 5
Freddy and Max were absent
absent were Max and Freddy
```

Answer:

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

int main() {
   string data[100];
   int n;
   cout << "How many words (between 1 and 100): ";
   cin >> n;
   if (n <= 0 || n > 100) exit(0);

   for (int c = 0; c < n; c++) cin >>data[c];
   for (int c = (n - 1); c >= 0; c--) cout << data[c] << " ";
   cout << endl;

   return 0;
}
```

Problem 413 Consider the following C++ program.

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

void mystery(int data[], int p, int q) {
  data[p] = data[q] + data[p];
  data[q] = 0;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(quiz, 7);
  print(scores, 8);
  mystery(scores, 3, 4);
  print(scores, 8);
  for (int i = 1; i < 7; i++)
     mystery(quiz, 0, i);
  print(quiz, 7);
}
```

What is the output from the program?

Answer:

```
0 1 2 3 4 5 6
3 1 4 1 5 9 2 6
3 1 4 6 0 9 2 6
21 0 0 0 0 0 0
```

Problem 414 Write a complete C++ program that does the following:

1. It generates two random numbers x and y each between 1 and 100. (You should use the functions *rand* and *srand*.)
2. It adds x and y to make a secret code.
3. It prints the secret code.

For example, if the program generated the numbers $x = 11$ and $y = 13$ which add to 24, the output would be:

```
The secret code is 24.
```

Answer:

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
using namespace std;

int main() {
  srand(time(NULL));
  int x, y;
  x = rand() % 100 + 1;
  y = rand() % 100 + 1;
  int code = x + y;
  cout << " The secret code is " << code << endl;
  return 0;
}
```

Problem 415 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer value, x.

2. The program reads the value entered by the user.

3. If the value is not positive, the program terminates. Otherwise, the program prints a checkerboard pattern that forms a square of side x.

For example, if the user enters 5 for x the program should print the following diagram with 5 lines.

```
* * *
 * *
* * *
 * *
* * *
```

(Hint: How is an even numbered row printed? How about an odd numbered row?)

Answer:

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

int main() {
   int x;
   cout << "Enter a positive integer value, x:";
   cin >> x;
   if (x <= 0) exit(1);

   for (int r = 1; r <= x; r++) {
     for (int c = 1; c <= x; c++) {
        if ((r + c) % 2 == 0) cout << "*";
        else cout << " ";
     }
     cout << endl;
   }

   return 0;
}
```

Problem 416 Write a C++ function called *negSum* that returns the sum of all negative elements in an array of integers.

It should be possible to use your function in the following program. (The output from this program is -12 because the negative elements -5, -4, and -3 have a sum of $-12 = -5 + (-4) + (-3)$.)

```
main() {
   int data[6] = {-5, -4, 1, 3, 2, -3};
   int x;
   x = negSum (data, 6);
     //   data is the array to search, 6 is the number of elements of the array
   cout << "The negative sum is: " << x  << endl;
}
```

Answer:

```
int negSum(int array[], int cap) {
   int answer = 0;
   for (int i = 0; i < cap; i++)
      if (array[i] < 0) answer += array[i];
   return answer;
}
```

Problem 417 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **isOdd** that is used to decide whether an integer is odd.

Answer:

```
bool isOdd(int x)
```

(b) A function called **max** which determines the largest of 3 double precision values.

Answer:

```
double max(double x, double y, double z)
```

(c) A function called **swap** which is used to swap two integer values.

Answer:

```
void swap(int &x, int &y)
```

(d) A function called **total** which is to find the sum of all entries in an array of integers.

Answer:

```
int total(int array[], int cap)
```

(e) A function called **maxIndex** which is to find the index of the largest element in an array of double precision values.

Answer:

```
int maxIndex(double array[], int cap)
```

(f) A function called **sort** which is to sort an array of integers into order.

Answer:

```
void sort(int array[], int cap)
```

Problem 418 Write a complete C++ program that:

1. Asks a user to enter the number of students in a class and the number of quizzes taken by the class.

2. If either of these numbers is less than 1 or more than 99 the program should exit.

3. The program should then prompt the user to enter all of the scores for each of the quizzes, starting with all scores for Quiz 1, followed by all scores for Quiz 2 and so on.

4. The program should print the number of the student with the highest total.

Number students and quizzes starting at 1.

A sample run of the program might look like:

```
How many students:  3
How many quizzes:   4

Enter scores for Quiz 1:  10  7 0
Enter scores for Quiz 2:  10 10 0
Enter scores for Quiz 3:  10  6 0
Enter scores for Quiz 4:  10  9 0

Student 1 got the highest total.
```

Answer:

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

int main() {
   int score[100][100];
   int r, c;
   int totals[100];
   int numStudents, numScores;

   cout << "Enter the number of students and the number of quizzes: ";
   cin >> numStudents >> numScores;
   if (numStudents <= 0 || numStudents >= 100
        || numScores <= 0 || numScores >= 100) exit(1);

   for (r = 1; r <= numScores; r++) {
      cout << "Enter the scores for Quiz " << r << ": ";
      for (c = 1; c <= numStudents; c++) cin >> score[r][c];
   }

   for (c = 1; c <= numStudents; c++) totals[c] = 0;
   for (r = 1; r <= numScores; r++) {
      for (c = 1; c <= numStudents; c++)
         totals[c] += score[r][c];
   }

   int topStudent = 1;
   for (c = 1; c <= numStudents; c++)
      if (totals[c] > totals[topStudent])
         topStudent = c;
   cout << "Student " << topStudent << " got the highest total." << endl;
   return 0;
}
```