CSCI 1201

Name ____

CSCI 1201.501

Final Exam (Practice)

Saturday, December 6, 2003

Part I. MULTIPLE CHOICE. Indicate the best answer from among the five choices available for each of the following questions. Record the answers on the supplied blue scantron sheet. *(1 pt. each)*

---

1. PCs are technically known as this type of machines:

    a. spinning tops
    b. flat-tops
    c. blacktops
    [✓ d. microcomputers]
    e. supercomputers

2. Of the computer's main components, these devices handle the conversion from human natural language to machine language.

    a. secondary memory devices
    b. main memory units
    [✓ c. input devices]
    d. CPUs
    e. output devices

3. C++ was created in the 1980s and is thus named because:

    [✓ a. it is the natural successor to C.]
    b. it is a post-fix language.
    c. it has two addition operators.
    d. All of these are true.
    e. None of these is true.

4. ISO standard C++ was approved as a national standard by the National Committee for Information Technology Standards in 1998. Standardization is important because:

    a. it allows C++ programs to be portable.
    b. it allows C++ compilers to be sold.
    c. it allows C++ programs to be readable.
    [✓ d. All of these are valid reasons.]
    e. None of these is a valid reason.

5. Compilation is the step in the software development cycle where this type of errors are detected.

    a. logic errors
    [✓ b. syntax errors]
    c. semantics errors
    d. specification errors
    e. all of these answers

6. Comments/documentation are meant for ____________ rather than the computer and provide explanation of the actual code.

    a. barcode readers
    b. optical readers
    c. mechanical readers
    d. alien readers
    [✓ e. human readers]

7. All of these are C++ data types ***except***:

    a. `int`
    [✓ b. `main`]
    c. `bool`
    d. `double`
    e. `char`

8. All of these are C++ keywords ***except***:

    a. `else`
    b. `switch`
    c. `using`
    d. `return`
    [✓ e. `constant`]

9. The selection logical structure is implemented in C++ using this statement:

a. `while` statement
[✓ b. `if` statement]
c. assignment statement
d. `for` statement
e. `return` statement

10. A structured data type of similarly typed elements which are accessible using indices:

[✓ a. array]
b. bag
c. set
d. union
e. none of these answers

11. The actual arguments in a function call have to match the _______ of parameters that appear in the function definition.

a. number
[✓ b. order]
c. data type
d. (a) and (b)
e. (b) and (c)

12. In the programming technique known as "top-down design," the main modules are written _______ the sub-modules.

a. after
[✓ b. before]
c. at the same time as
d. to replace
e. all of these answers

13. Searching is a fundamental problem associated with arrays that seeks to:

[✓ a. locate in the array where target element is]
b. insert into the array the target element
c. reorder the elements in the array
d. delete the elements in the array
e. all of these are correct answers

14. Sorting is a fundamental problem associated with arrays that seeks to:

a. locate in the array where target element is
b. insert into the array the target element
[✓ c. reorder the elements in the array]
d. delete the elements in the array
e. all of these are correct answers

15. This is a reason why C++ is still a viable programming language some twenty years after its creation:

a. It is a concise language.
b. It has advanced features for all-purpose computing.
c. There are lots of smart compilers for it that produce efficient code.
d. There is a substantial legacy library of useful C++ code
[✓ e. All of these are good reasons.]

16. This is one of the object-oriented languages that are competing with C++ in the marketplace:

[✓ a. Java]
b. Pascal
c. Ada
d. All of these are correct answers.
e. None of these is a correct answer.

17. This condition states what is assumed to be true when a function is called.

a. air condition
[✓ b. precondition]
c. postcondition
d. conditional statement
e. none of these answers

18. This condition describes the effect of the function call; that is, it tells what will be true after the function is executed.

a. air condition
b. precondition
[✓ c. postcondition]
d. conditional statement
e. none of these answers

19. A(n) __________ is a kind of placeholder that is filled in with a function argument when the function is called.

a. literal
b. declaration
[✓ c. formal parameter]
d. operator
e. key word

20. A call-by-reference parameter is indicated by attaching the _________ to the end of the type name in the formal parameter.

a. hyphen, -
b. vertical bar, |
c. semicolon, ;
[✓ d. ampersand sign, &]
e. none of the above

21. One can specify a ________ for one or more call-by-value parameters in a function. If the corresponding argument is omitted, then it is replaced by this value.

a. default argument
b. final value
c. temporary value
d. resident address
e. none of the above

22. If an array variable is declared as `score[5]` then the valid range of array indexes for `score` is:

a. `1..5`
[✓ b. `0..4`]
c. `0..5`
d. `5..5`
e. none of these answers

23. In the definition of a class, it is preferred that data members of a class be declared as this:

a. `public`
[✓ b. `private`]
c. `int`
d. `double`
e. All of these answers are valid.

24. A well-designed class should provide a complete set of accessor functions. Assess the value of this "rule."

[✓ a. This rule is highly recommended.]
b. This rule is not at all recommended.

25. An object is an instance of a class. Assess the truth of this statement.

[✓ a. This statement is true.]
b. This statement is not true.

26. In a class definition, all data members should be of the same data type much like array definitions. Assess the truth of this statement.

a. This statement is true.
[✓ b. This statement is not true.]

27. In a class definition, member functions designed to modify the data members of the class are called accessor functions. Assess the truth of this statement.

[✓ a. This statement is true.]
b. This statement is not true.

28. In a class definition, member functions designed to initialize the data members of the class are called constructor functions Assess the truth of this statement.

[✓ a. This statement is true.]
b. This statement is not true.

Part II. PROGRAM SIMULATION. Armed with your knowledge of C++ syntax, supply the answers required below.

1. Suppose that the code segment in the box below were executed:

```
int a, b, z;
a = 3;
b = 4;
a = b;
b = a;
z = b;
```

What would be the value of z? *(5 pts.)* Record your answer here: 4

2. Suppose the following were executed:

```
int x, y, z;
x = 3;
y = 7;
z = 9;
if ( x > y )
  if ( x - y == 4 )
    z = 33;
  else
    z = 77;
```

What would be the value of z? *(5 pts.)* Record your answer here: 9

3. Suppose the following were executed:

```
int a, b, z;
a = 1;
b = 4;
while ( a < b ) {
  a = 2*a;
  b = b + 1;
}
z = b;
```

What would be the value of z? *(5 pts.)* Record your answer here: 7

Part III. SHORT CODING. Supply the C++ code that is required.

1. We want to write a boolean function called `isPrime` that determines whether a given integer `number` is prime or not (a number is prime if its only divisors are itself and 1). We give here the specification of the function. You must supply the missing code elements in the spaces provided. Use the comments and documentation to determine the most appropriate C++ code fragment for the blanks:

```
/* This function returns the value true if the supplied
 * parameter number is prime, i.e., not divisible by any
 * number other than 1 and itself, and false otherwise.
 * For example, isPrime( 13 ) should return true while
 * isPrime( 10 ) should return false.
 */

 bool isPrime( int number ) {

 //Declare local bool variable to store the result

 __bool result_________________________ ;
                3 pts.

 //Assign an initial value of true to the bool variable

 result = __true______________ ;
                2 pts.

 //Loop through possible divisors of number starting at 2 and
 //going all the way to half of the parameter number

 for ( int divisor = 2; _divisor < number/2_; divisor++ )
                          3 pts.
   //Test if divisor divides number exactly; update result if it does
   result = result && ( number / divisor * divisor != number );

 //return the correct value

 return __result__________ ;
               2 pts.

} //end isPrime
```

2. We want to write a function that will print the following output:

```
1
1   2
1   2   3
1   2   3   4
    :
1   2   3   4   5   ...   n
```

where *n* is a positive number less than or equal to 99. For example, the following call to the desired function:

```
          numberPrint( 7 );
```

should produce the following output:

```
1
1   2
1   2   3
1   2   3   4
1   2   3   4   5
1   2   3   4   5   6
1   2   3   4   5   6   7
```

The formal specification of the function is given below. As in No. 1, you will need to supply the missing code elements; use the documentation and comments to guide you to the most appropriate C++ code fragment.

```
/**
 *  Prints a "pyramidal" table of consecutive numbers
 *  with numRows rows; the i-th row consists of consecutive
 *  numbers from 1 to i.
 *    Precondition: numRows >=1 && numRows <= 99
 */
void numberPrint( int numRows )

//Declare an int array named numArray with 100 cells


  int numArray[100];
                   3 pts.

//Fill the i-th cell with the integer i for all 100 cells in numArray

  for ( int i=0; i < 100; i++ )

    numArray[ i ] = i         ;
                        2 pts.

//To produce the pyramid, we print increasing sequences of integers
//from numArray using nested for loops

//(turn to next page)
```

```
//Set up a loop to produce the rows of the pyramid
  for ( int i = 1; i <= numRows; _i++_________ ) {
                                      1 pt.

  //Print a row of the pyramid using another loop
    for ( int j = 1; j <= i; j++ )

       __cout___ << numArray[ j ];  //Print individual numbers
         2 pts.

    cout << __endl__________ ;  //Print end-of-line character
                 2 pts.
  }
} //end numberPrint
```

Part IV. Problem Solving and Coding. Read the problem analysis and supply the required code according to your understanding of the analysis.

Problem Statement

An input file consists of integers representing students' grades. We need a program to read the input file and then print the grades in descending order with a message at the end showing the average of the grades read. You may assume that the number of values in the input file is between 1 and 35, inclusive.

Sample Input:

```
99   67   85   49   98   91  99
```

Corresponding Output:

```
Grades in descending order:
99   99   98   91   85   67   49
Average grade:   84.0
```

Subprogram Availability:

You have at your disposal (*i.e.,* it is already written and available for use) a function called sort that takes a one-dimensional array of integers and sorts it in either increasing or decreasing order, depending on the value of one of its parameters). The formal specification of this function is given below:

```
/**
 * Sorts the integer array numArray containing size integers
 * in increasing order if mode is 1 and in decreasing order
 * if mode is -1
 */

 void sort( int numArray[] , int size , int mode );
```

(more)

You are required to design a program that solves this problem using functions. You already have a sorting function; build on this module. Include a function that reads in the integer values (call it `getGrades`) and a function that prints the results to standard output (call it `printGrades`).

Decide what parameters should be included for each of the functions mentioned above and invoke them at the appropriate time in your main program. Include as much documentation as is necessary to make your answer understandable. You do not have to write the bodies of the functions; all you have to do is provide their formal specifications, that is, the formal parameter list and the type declaration of these parameters (see function `sort` above).

Use the space below (and if needed, the back side of this page) to write your main program and the functions' formal specifications. *(20 pts.)*

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

/**
 * Program that reads in grades, sorts them in decreasing order, and
 * then computes their averages before the results are displayed.
 */

//specification of functions used in main

void getGrades( int gradeArray[], int& numValuesRead );
  //obtains grades from user and returns them in gradeArray  together with
  //the number of values read
double average( int gradeArray[], int numGrades );
  //computes the average of the numGrades grades in gradeArray
void printGrades( int gradeArray[], int numGrades, double average );
  //displays the grades and their average


int main() {

   int gradeArray[35];  //array to hold grade values
   int numGrades=0;     //holds the number of grades read in
   double avg=0.0;      //contains the average of the grades

   getGrades( gradeArray, numGrades ); //obtain grades from user
   sort( gradeArray, numGrades, -1 ); //sort grades in descending order
   avg = average( gradeArray, numGrades ); //compute average of grades
   printGrades( gradeArray, numgrades, avg ); //display results

   return 0;

} //end main
```