READ THIS NOW!

Failure to read and follow the instructions below may result in severe penalties.

- Print your name in the space provided below.
- Print your name and ID number on the Opscan form; be sure to code your ID number correctly on the Opscan form. Code Form B on the Opscan.
- Choose the single best answer for each question — some answers may be partially correct. If you mark more than one answer to a question, you will receive no credit for any of them.
- Unless a question involves determining whether given C++ code is syntactically correct, assume that it is. Unless a question specifically deals with compiler #include directives, you should assume the necessary header files have been included.
- Be careful to distinguish integer values from floating point values (containing a decimal point). In questions/answers that require a distinction between integer and real values, integers will be represented without a decimal point, whereas real values will have a decimal point, [1044 (integer), 1044.0 (floating point)].
- This is a closed-book, closed-notes examination. No calculators or other electronic devices may be used during this examination. You may not discuss (in any form: written, verbal or electronic) the content of this examination with any student who has not taken it. You must return this test form when you complete the examination. Failure to adhere to any of these restrictions is an Honor Code violation.
- There are 25 equal-valued multiple-choice questions.
- The answers you mark on the Opscan form will be considered your official answers.
- When you have finished, sign the pledge at the bottom of this page and turn in the test and your Opscan.

Do not start the test until instructed to do so!

Name (Last, First) _____________________________________________________________________

Pledge: On my honor, I have neither given nor received unauthorized aid on this examination.

___________________________________________________________________________________

Signature
Consider executing the following code fragment (assume any additional declarations, etc, needed to make the code syntactically correct):

```cpp
int j = 3;
while (j < 81) {
    j = j * 3;
    cout << j << endl;
}
```

1. How many times will the body of the loop be executed?
   1) 3  3) 5  5) None of these
   2) 4  4) infinite loop

2. What is the last value printed?
   1) 243  3) 80  5) 9  7) 3
   2) 81  4) 27  6) 6  8) None of these

For the next three questions, consider execution of the following program:

```cpp
void Foo(int Harry, int& Potter);
int Tmp = 3;

int main() {
    int Alpha = 12, Beta = 2;
    Foo(Alpha, Beta);
    cout << "Alpha = " << Alpha << endl;
    cout << "Beta = " << Beta << endl;
    cout << "Tmp = " << Tmp << endl;
    return 0;
}

void Foo(int Harry, int& Potter) {
    int Tmp;
    Tmp = Harry;
    Harry = Harry - 2 * Potter;
    Potter = Tmp;
}
```

3. What value is printed for the variable Tmp?
   1) -2  3) 3  5) 12  7) 21
   2) 2  4) 7  6) 19  8) None of these

4. What value is printed for the variable Alpha?
   1) -2  3) 3  5) 12  7) 21
   2) 2  4) 7  6) 19  8) None of these

5. What value is printed for the variable Beta?
   1) -2  3) 3  5) 12  7) 21
   2) 2  4) 7  6) 19  8) None of these
6. Consider executing the following code fragment (assume any additional declarations, etc, needed to make the code syntactically correct):

```c++
int j = 4;
double varQ = 1.5, newQ = 0.0;
while (j != 47) {
    newQ = newQ + varQ;
    j = j + 4;
}
```

How many times will the body of the loop be executed?

1) 11  
2) 12  
3) 13  
4) 46  
5) 47  
6) infinite loop  
7) None of these

7. A function, `someFunc`, has two formal parameters, `A1` of type `int` and `B2` of type `string`. The data flow (communication) for variable `A1` is two-way, into and out of the function. The data flow for variable `B2` is one-way, into the function. Which of the following is the most appropriate prototype for `someFunc`?

1) `void someFunc(int A1, string B2);`
2) `void someFunc(int A1, const string& B2);`
3) `void someFunc(int& A1, string B2);`
4) `void someFunc(int& A1, const string& B2);`
5) All of them are equally appropriate.
6) None of these

8. In C++, struct variables differ from arrays in what way(s)?

1) whether the elements must be of the same type
2) whether the elements are accessed by location
3) Both of these
4) None of these

9. Consider executing the following program:

```c++
int main() {
    int i, j, sum;
    sum = 0;
    for (i = 5; i > 0; i=i-2) {
        for (j = 0; j < 4; j=j+2)
            sum = sum + i + j;
        cout << sum << endl;
    }
    return 0;
}
```

What is the value printed on the last line of output?

1) 0  
2) 12  
3) 14  
4) 24  
5) 30  
6) 32  
7) 42  
8) None of these
10. Given the following code fragment:

```cpp
struct Name {
    string First;
    string Last;
};

struct Student {
    Name StudentName;
    string ID;
    double FinalAverage;
};

Student MyStudent;
```

which of the following statements are legal?

1) `MyStudent.First = "John";
2) `MyStudent.Name = "Bill";
3) `Name.Last = "Smith";
4) `MyStudent.StudentName.Last = "Jones";
5) All are legal
6) None are legal

11. Assume the following declarations:

```cpp
void Fix(double realParameter, int& intParameter);
int someInt = 12;
double someFloat = 6.28;
```

Which of the following would represent an appropriate call(s) of the function `Fix`?

1) `Fix(6.85, 24);
2) `Fix(6.85, someInt);
3) `Fix(someFloat, 24);
4) `Fix(someFloat, someInt+5);
5) `Fix(someFloat+2.0, someInt);
6) All of them are appropriate.
7) 1 and 2 only
8) 2 and 5 only
9) 2, 4 and 5 only
10) None of these

12. Suppose the first few lines of a function are as follows:

```cpp
void Calculate( float beta ) {
    alpha = 3.8 * beta;
}
```

Assuming the program compiles, the variable `beta` must be:

1) local to the function `Calculate`
2) local to the function that calls `Calculate`
3) a global variable
4) a formal parameter
5) an actual parameter
6) 1 and 4 only
7) 2 and 5 only
8) None of these
For the next two questions, assume the declarations below:

```cpp
MAXSIZE = 20;
char myArray[MAXSIZE + 1];
```

13. How should the blank preceding MAXSIZE be filled?

1) int  
2) char  
3) const int  
4) const double  
5) 1 or 3 only  
6) 3 or 4 only  
7) 1, 3 or 4 only  
8) None of these

14. What is the range of valid index values for myArray[]?

1) 0 through 21  
2) 1 through 20  
3) 0 through 19  
4) 1 through 21  
5) 0 through 19  
6) 0 through 20  
7) None of these

15. Given the following definition:

```cpp
struct Part {
    string Name;
    string Make;
    string Model;
    double Price;
};
```

which of the following are legal statements?

1) Part.Name = "CPU";
2) Part ComputerPart;
3) void foo(Part ComputerPart); // Function prototype
4) void foo(const Part ComputerPart); // Function prototype
5) void foo(Part ComputerPart.Name); // Function prototype
6) All are legal  
7) 1, 2 and 3 only  
8) 2, 3, and 4 only  
9) None are legal  
10) None of these
16. Consider the following code fragment:

```cpp
struct MyStruct {
    string FirstString;
    int AnInt;
    string SecondString;
};
const int Size = 50;
...
int main() {
    MyStruct Array[Size];
    MyStruct aStruct;
    ...
}
```

Which of the following are valid function invocations for a function with the prototype:

```cpp
void MyFunction(MyStruct Tmp)?
```

1) MyFunction(Array);
2) MyFunction(Array[3]);
3) MyFunction(Array[50]);
4) MyFunction(Array[]);
5) All of them are valid.
6) 1 and 2 only
7) 1, 2 and 3 only
8) 2, 3 and 4 only
9) None of these

For the next two questions questions 17 and 21, assume that the following array is declared as follows:

```cpp
int QuizScores[MAXSTUDENTS][MAXQUIZZES];
```

17. The maximum number of quizzes that can be stored in QuizScores is:

1) MAXSTUDENTS + 1) * (MAXQUIZZES + 1)
2) (MAXSTUDENTS - 1) * MAXQUIZZES
3) MAXSTUDENTS * (MAXQUIZZES + 1)
4) MAXSTUDENTS * MAXQUIZZES
5) (MAXSTUDENTS - 1) * (MAXQUIZZES - 1)
6) (MAXSTUDENTS + 1) * MAXQUIZZES
7) MAXSTUDENTS * (MAXQUIZZES - 1)
8) Not enough information is available.
9) None of these
For the next three questions, assume the following declarations for a program to keep track of a sporting event schedule:

```cpp
const int NUMGAMES = 5;
const int NUMTEAMS = 5;
string HomeTeam[5];
string AwayTeam[5];
string Winner[5];
```

18. Which of the following is the most appropriate choice for filling the blank in the declaration of `TEAMS`?

1) NUMGAMES
2) NUMTEAMS
3) 5
4) NUMGAMES + 1
5) NUMTEAMS + 1
6) None of these

19. Which of the following is the most appropriate choice for filling the blank in the declaration of `Winner`?

1) NUMGAMES
2) NUMTEAMS
3) 5
4) NUMGAMES + 1
5) NUMTEAMS + 1
6) None of these

20. The arrays `HomeTeam`, `AwayTeam`, and `Winner` are

1) parallel arrays
2) single-dimensional arrays
3) multi-dimensional arrays
4) All of the above
5) 1 and 2 only
6) 1 and 3 only
7) None of these

21. Suppose a function named `Average` exists and accepts `QuizScores` as a one-way parameter into the function and returns a double. The most appropriate function prototype is:

1) `void Average(int Scores[][MAXQUIZZES], int students, int quizzes, double& Result);`
2) `double Average(const int Scores[][MAXQUIZZES], int students, int quizzes);`
3) `void Average(const int Scores[][MAXQUIZZES], int students, int quizzes);`
4) `double Average(int Scores[MAXSTUDENTS][], int students, int quizzes);`
5) `double Average(const int Scores[MAXSTUDENTS][], int students, int quizzes);`
6) `double Average(int Scores[MAXSTUDENTS][MAXQUIZZES], int students, int quizzes, double& Result);`
7) `void Average(int Scores[MAXSTUDENTS][MAXQUIZZES], int students, int quizzes, double& Result);`
8) All of the above are equally appropriate.
9) None of these
For the next four questions, consider the incomplete function definition given below:

```c
// CalcAverage takes an array of grades stored as integers
// and the number of values it contains, prints the grades
// and determines the average grade.
//
// Parameters:
// Grades[] array containing grades.
// NumOfGrades number of values stored in Grades[]
//

double CalcAverage(Grades[], NumOfGrades) // Line A
{
    int Sum = 0;
    int LoopCounter;

    for(LoopCounter = 0; ; LoopCounter++) // Line B
    {
        Sum = Sum + Grades[LoopCounter]; // Line C
    }

    return Sum / NumOfGrades; // Line D
}
```

22. How should the blank preceding the first parameter in line A be filled?

1) const int  
2) int  
3) int&  
4) It should be left blank.  
5) None of these

23. How should the blank preceding the second parameter in line A be filled?

1) const int  
2) int  
3) int&  
4) It should be left blank.  
5) None of these

24. How should the blank in line B be filled?

1) Sum <= NumOfGrades  
2) LoopCounter <= NumOfGrades  
3) Sum < NumOfGrades  
4) LoopCounter < NumOfGrades  
5) None of these

25. How should the blank in line C be filled?

1) Sum / NumOfGrades  
2) Grades[NumOfGrades]  
3) Sum  
4) Grades[LoopCounter]  
5) Sum / LoopCounter  
6) double(Sum) / double(NumOfGrades)  
7) It should be left blank.  
8) None of these