You may take this test with you afterwards, but you must turn in your answer sheet.

This test has 25 multiple-choice questions, worth 4 points each, for a total of 100 points.

This test is worth 10% of your final grade. You must put your answers on the bubble form. This test is open book and open notes, but no computers. For the multiple choice problems, select the best answer for each one and select the appropriate letter on your answer sheet. Be careful - more than one answer may seem to be correct and some questions are tricky. When a section of code is described as a program segment you can assume it is placed in the context of a program that is otherwise correct and includes all declarations and system libraries needed to make it work.

1. Consider the exercise we did the first day of class, where we explored different methods of counting the number of students in the room. One approach we used was for a person to find another person standing, compare how many “stored” values each had, have one person sum them and represent that total, and then the other person sat down. This was then repeated.

What was the advantage of this approach that we discussed?
A) It is more accurate than the other approaches we explored
B) Everyone gets to participate and actively learn
C) The approach is similar to what computers do
D) This method of counting illustrates a process being scalable

2. In a C program the return type of main() should be:
A) void
B) int
C) float
D) It doesn’t matter, as long as the declared type is consistent with the actual value being returned.

3. Consider the following declaration:

```
char word1[] = "ABC";
```

Assuming a single byte is allocated for each character, how many characters total will array word1 take up in memory?
A) Only one, for the address of the array
B) 3
C) 4
D) It doesn’t compile because the size is not given inside the square brackets [ ]

4. A *switch-case* statement can always be rewritten with multiple *if-else* statements, however multiple *if-else* statements cannot always be equivalently rewritten using a *switch-case* statement. Why is this?
   
   A) *Switch-case* statements use the *break* statement, which is not used in *if* statements.
   
   B) *Switch-case* statements can be used to equivalently rewrite the *if* part, but not the *else* part of multiple *if-else* statements
   
   C) *Switch-case* statements can’t handle multiple conditions matching to a single section of code.
   
   D) *Switch-case* statements can be used with variables of type *char* and *int*, but can’t be used with variables of type *double*.

5. What is the relationship between programming in C and programming in C++?
   
   A) They are different names for the same thing
   
   B) A C compiler can run any C++ program
   
   C) A C++ compiler can run any C program
   
   D) They require different compilers

6. What is the point of the video shown in class of a man eating a bicycle?
   
   A) Parts of some problems can’t be solved without a computer
   
   B) It is helpful to view problems from different perspectives
   
   C) To solve a problem try and find solutions to a similar problem
   
   D) Bigger problems should be broken down into smaller problems

7. Consider the program or program segment shown at left below. Which of the options (at right below) is the best answer regarding this program or program segment?

   ```c
   #include <stdio.h>
   int main()
   {
       printf("Second One")
       return 0
   }
   ```
   
   A) It will not compile
   
   B) It will compile but will crash when it runs
   
   C) It will compile and run, but will give unexpected results
   
   D) It will compile and run as expected

8. Consider the program segment shown at left below. Which of the options (at right below) is the best answer regarding this program segment?

   ```c
   int value;
   printf("Enter value: \n");
   scanf("%d", value);
   printf("+1 is:%d",value+1);
   ```
   
   A) It will not compile
   
   B) It will compile but will crash when it runs
   
   C) It will compile and run, but will give unexpected results
   
   D) It will compile and run as expected
9. Consider the program or program segment shown at left below. Which of the options (at right below) is the best answer regarding this program or program segment?

```c
int value;
printf("Enter a number: ");
scanf("%d", &value);
if( value = 2) {
    printf("%d is even", value);
} else {
    printf("%d is odd", value);
}
```

A) It will not compile  
B) It will compile but will crash when it runs  
C) It will compile and run, but will give unexpected results  
D) It will compile and run as expected

10. Consider the program or program segment shown at left below. Which of the options (at right below) is the best answer regarding this program or program segment?

```c
char letter = 'B';
char values[3] = {'C','D','E'};
char aWord[] = "alibi";
```

A) It will not compile  
B) It will compile but will crash when it runs  
C) It will compile and run, but will give unexpected results  
D) It will compile and run as expected

11. Consider the program or program segment shown at left below. What is the output of this program segment?

```c
int i=0;
while( aWord[i]!="\0") {
    i++;
}
printf("%d", i);
```

A) The address of the first character of aWord  
B) a  
C) 5  
D) There is no output since it does not compile

12. Consider the following statements about using functions in a program:
   I. Functions allows the same code to be used in multiple circumstances  
   II. Functions help you subdivide a problem  
   III. Functions allow you to make parts of your program run faster

Which of the above are true statements about functions?
A) II only  
B) I and II  
C) I and III  
D) I, II and III
13. Consider using binary search to lookup a word in a dictionary of 44,000 words stored in alphabetical order. In the worst case, how many comparisons would need to be made to lookup a word or verify that it is not in the dictionary?

   A) 10
   B) 16
   C) 200
   D) 22,000

14. Consider the code segment shown below:

   ```c
   int i=0;
   do {
       printf("%d ", i); ++i;
   } while( i<5);
   ```

Which of the following two code segments will give the same output as the above code?

   **Option I:**
   ```c
   int j=1;
   while( j<=5) {
       printf("%d ", j);
       j++;
   }
   ```

   **Option II:**
   ```c
   int k;
   for( k=0; k<5; k++) {
       printf("%d ", k);
   }
   ```

   A) Neither I nor II will give the same output.
   B) I will give the same output, but II will not
   C) II will give the same output, but I will not
   D) Both I and II will give the same output.

15. For which of the following cases is a **for** loop the most appropriate?

   A) Displaying a menu and getting the input, verifying it is a valid menu option
   B) Checking a logical condition to decide whether some code should be executed at all
   C) Repeating code a known number of times

16. For which of the following cases is a **do-while** loop the most appropriate?

   A) Displaying a menu and getting the input, verifying it is a valid menu option
   B) Checking a logical condition to decide whether some code should be executed at all
   C) Repeating code a known number of times
17. Consider the following statement:
\[
\text{grade} = (\text{score} > 90) \ ? \ 'A' : 'B';
\]
This is equivalent to which of the following sections of code?

A) \[
\text{if (score > 90)} \\
\quad \text{grade} = 'A' - 'B'; \\
\text{else} \\
\quad \text{grade} = 'B' - 'A';
\]

B) \[
\text{if (score < 90)} \\
\quad \text{grade} = 'A'; \\
\text{else} \\
\quad \text{grade} = 'B';
\]

C) \[
\text{if (score >= 90)} \\
\quad \text{grade} = 'A'; \\
\text{else} \\
\quad \text{grade} = 'B';
\]

D) \[
\text{if (score <= 90)} \\
\quad \text{grade} = 'B'; \\
\text{else} \\
\quad \text{grade} = 'A';
\]

For the following three problems, consider the following four alternatives of types of code, where the layout helps convey what the code is doing:

A) \[
\text{if (expression1)} \\
\quad \text{action1;} \\
\text{else if (expression2)} \\
\quad \text{action2;} \\
\text{else if (expression3)} \\
\quad \text{action3;}
\]

B) \[
\text{if (expression1)} \\
\quad \text{action1;} \\
\quad \text{if (expression2)} \\
\quad \quad \text{action2;} \\
\quad \text{if (expression3)} \\
\quad \quad \text{action3;}
\]

C) \[
\text{switch (variable)} \\
\quad \text{case 1: action1; break;} \\
\quad \text{case 2: action2; break;} \\
\quad \text{case 3: action3; break;}
\]

D) \[
\text{if (expression1)} \\
\quad \text{action1;} \\
\text{else} \\
\quad \text{if (expression2)} \\
\quad \quad \text{action2;} \\
\text{else} \\
\quad \quad \text{if (expression3)} \\
\quad \quad \quad \text{action3;}
\]

18. Which of the above types of code would be best to use for a program that checks a test score and assigns a letter grade? Select A, B, C or D.

19. Which of the above types of code would be best to use for a program that checks conditions for a tax program, where multiple conditions may all be true? Select A, B, C or D.

20. Which of the above types of code would be best to use for a program that checks whether input is upper-case, lower-case, or numeric? Select A, B, C or D.
21. Consider what function should be called at the underlined section in the code shown below:

```c
void f1( int *p, int *q)
{
    *p = *p + 1;
    *q = *q - 1;
}

void f2( int a, int b)
{
    a++; 
    b--; 
}

void f3( int *x, int *y)
{
    *x = *x - 1;
    *y = *y + 1;
}
```

Which of the following is the best answer regarding which function call should go in the underlined section, so that when we call function `parameters()` we print out the value 10?

A) Any of the three of them could be used
B) We should use: `f1(&y,&x);`
C) We should use: `f2(x,y);`
D) We should use: `f3(&x,&y);`

22. What is the output from the code segment shown at right below, called with `scope();`

```c
int x=0;      // global variable

void s2( int y)
{
    printf("%d", x+y);
}

void s1( int y)
{
    y++; 
    int x=y;
    s2( y);
}

void scope()
{
    x=2;
    s1( x);
}
```
23. What is the output from the following section of code?

```c
char words[] = "Able Baker Charlie ";
char * pSpace;
pSpace = strchr( words,' ');
*pSpace='\0';
int n = pSpace - words;
char aWord[ 81 ];
strncpy( aWord, words, n+1);
printf("%s\n", aWord);
```

A) A
B) Able
C) Able Baker Charlie
D) Baker

24. What is the output from the following section of code?

```c
char words[] = "the force use you must"; char * pOne = strchr( words, ' ');
char * pTwo = strchr( pOne+1, ' ');
*pOne = '\0';
*pTwo = '\0';
printf("%s %s\n", pOne+1, pTwo+1);
```

A) the force
B) force use
C) force use you must
D) force use must use you must

25. Consider the function shown below:

```c
char * stringFunctions2( char * *pCurrent, char theArray[])
{
char * pSpace = strchr(*pCurrent,' ');
pSpace = '\0';
char *pTheWord = *pCurrent;
pCurrent = pSpace + 1;
return pTheWord;
}
```

What is the output when the above function is used with the following code:

```c
char text[]="build a wall around it";
char * pCurrent = text;
printf("%s ", stringFunctions2( &pCurrent, text));
printf("%s ", stringFunctions2( &pCurrent, text));
```

A) build
B) build build
C) build a
D) build a wall around it