

Name: _____

SSN: _____

Login: _____

CS 12 - Midterm February 18, 1998

Be sure to read each problem carefully and follow the directions. Points may be marked off if you do not follow the directions. For example, if the problem asks you to write a function, do not write an entire program. If you are asked to write code, only write the necessary code. You should write efficient code that has good style, but do not worry about commenting unless you are doing something obscure that needs explanation. Please feel free to ask if you have any questions.

Problem 1	10	
Problem 2	6	
Problem 3	5	
Problem 4	15	
Problem 5	13	
Problem 6	12	
Problem 7	10	
Problem 8	6	
Problem 9	10	
Problem 10	13	
TOTAL	100	

1. (10 pts) True/False (please write out “True” and “False”, do not just put a T or F).

(a) Every recursive function can be written iteratively.

(b) The *getline* function reads characters from the input stream until it finds a newline ($\backslash n$).

(c) When a width is set, the justification is set to right justified.

(d) A character array is a string.

(e) Classes and structures have the same capabilities.

2. (6 pts) List three uses of pointers.

3. (5 pts) Show how to output a floating point number y with a precision of 3 and a width of 10 using **printf**.

4. (15 pts) Define the following terms.

(a) Pointer

(b) String

(c) ifstream

(d) Pass-by-reference

(e) Abstract Data Type

5. (13 pts) Given the following code:

```
struct bunch_o_stuff {
    int x;
    double y;
    int *ip;
}gadget;

char name[20];
double scores[10][3];
bunch_o_stuff junk_pile[10];
```

What are the types of the following expressions? Write in the letter of the corresponding correct answer in the blank. An answer may be used more than once.

- | | | | |
|--------|---------------|---|--------------------------------|
| ___ 1 | gadget | A | integer |
| ___ 2 | gadget.x | B | double |
| ___ 3 | gadget.*ip | C | character |
| ___ 4 | *gadget.ip | D | character array |
| ___ 5 | name | E | array of doubles |
| ___ 6 | name[3] | F | 2-dimensional array of doubles |
| ___ 7 | scores | G | string |
| ___ 8 | scores[1] | H | pointer |
| ___ 9 | scores[1][1] | I | integer pointer |
| ___ 10 | junk_pile | J | character pointer |
| ___ 11 | junk_pile[3] | K | bunch_o_stuff |
| ___ 12 | junk_pile.ip | L | array of bunch_o_stuff |
| ___ 13 | bunch_o_stuff | M | invalid expression |

6. (12 pts) Show the values of each element in the array *A* after the following code has been performed.

```
int A[6] = {10, 20, 30, 40, 50, 60};
int *ip;
int x = 3;
int y = 2;

ip = A;
*ip = 4;
ip++;
*ip = *ip + 5;
ip += x;
(*ip)++;
*(ip+1) = 12;
ip = ip - y;
*ip = 2 * (*ip - 10);
```

7. (10 pts) Given the following code:

```
char str1[20];
char str2[20];

cout << "Enter a word: ";
cin >> str1;

cout << "Enter another word: ";
cin >> str2;
```

write code to dynamically allocate a string called **str3**. This string should be given **just enough** memory to contain **str1** concatenated with **str2**. Write code to set **str3**.

8. (6 pts) Assume each set of code below is executed separately and is given the input:

```
‘‘It’s midterm time.\n’’
```

Show the output and show what is left in the input buffer for each code segment. Be sure to be very clear about all characters.

(a)

```
char buf[30];  
cin >> buf;  
cout << buf;
```

(b)

```
char buf[30];  
cin.getline(buf, 30, '\n');  
cout << buf;
```

(c)

```
char buf[30];  
cin.get(buf, 30, '\n');  
cout << buf;
```

9. (10 pts) Write a recursive function that calculates and returns x raised to the power of n . The function should take two parameters, x and n . x is of type double and n is of type integer. For example 2.0 raised to the power of 2 is 4.0. You only need to take care of nonnegative values of n . The prototype for the function should be:

```
double power(double x, int n);
```

10. (13 pts) Write a program that reads from a user specified file. Each line in the file contains a double, referred to as *num*, followed by an integer, referred to as *power*. The amount of whitespace before, between, and after the numbers is arbitrary.

Assume you have access to a function that has already been written which calculates the value of some number x raised to some power n . This function has the following prototype:

```
// raise x to the power of n, return result
double power(double x, int n);
```

Your program should read each line of numbers from the file and show the value of *num* raised to the power of *power*. Each result should be printed to the screen on a separate line.

You do not need to write this function, just assume it has been written and use the function.