1.) Given “char *a = “Hello”; char *b = “World;”, which of the following would result in an error?
   A) strlen(a)
   B) strcpy(a, b)
   C) strcmp(a, b)
   D) strstr(a, b)

2.) Given “int *p;”, which of the following is an invalid operation?
   A) p * 2
   B) &p
   C) *p
   D) p++

3.) Suppose you are reading a record from a file into a struct. One of the fields of that record may contain an integer or a string depending on the situation. What would be the best type for the member variable that holds the field that minimizes memory usage?
   A) struct
   B) array
   C) char*
   D) union

4.) Given variable declaration “const char *p;”, which of the following is a legal statement?
A) \&p = \text{NULL};
B) \ast p = 0;
C) p = \text{NULL};
D) None of the above

5.) Which of the following is \textbf{false} about storage classes?
A) A static global variable is visible across the entire program
B) An automatic local variable is deallocated at the end of its scope
C) A global variable is initialized when the program starts
D) There is no storage class with an automatic lifetime that is visible across the entire program

\textbf{Short Answer}

6.) Your program crashed due to an invalid pointer access. But strangely you don’t see output from a previous printf() call that should have executed before the pointer access. What might be the reason?

7.) Given “unsigned int c;”, write the correct statement using bitwise operators to do the following.
   a.) Set the 15\textsuperscript{th} bit of c to 1
   b.) Set the 15\textsuperscript{th} bit of c to 0
8.) Explain two distinct reasons why you would pass a pointer to a variable to a function, instead of the variable itself.

9.) For each of the below code snippets, 1) State what is wrong with the program and 2) State exactly how you would fix it by referencing the line numbers provided.

a.) The below program should print “Two”.

```c
1. void update(char *s) { s = "Two"; }  
2. int main()  
3. {  
4.   char *str = "One"; 
5.   update(str);  
6.   printf("%s\n", str);  
7.   return 0;  
8. }
```
b.) The below function `format(...)` should return a string formatted with the given number.

```c
1. char* format(int n)
2. {
3.     char str[100];
4.     sprintf(str, "%d\n", n);
5.     return str;
6. }
```

10.) Given the declaration “char a[2][3] = {{10, 20, 30}, {40, 50, 60}};” and the layout in memory given below, write the numerical value of the following expressions. If you believe the expression is illegal, write “illegal” and explain why.

<table>
<thead>
<tr>
<th>Address</th>
<th>Memory</th>
<th>Name of Storage Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x200</td>
<td>...</td>
<td>a[0][0]</td>
</tr>
<tr>
<td>0x204</td>
<td>10</td>
<td>a[0][1]</td>
</tr>
<tr>
<td>0x208</td>
<td>20</td>
<td>a[0][2]</td>
</tr>
<tr>
<td>0x20c</td>
<td>30</td>
<td>a[1][0]</td>
</tr>
<tr>
<td>0x210</td>
<td>40</td>
<td>a[1][1]</td>
</tr>
<tr>
<td>0x214</td>
<td>50</td>
<td>a[1][2]</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

a.) What is the value of “a”?

b.) What is the value of “a[0]”?

c.) What is the value of “a+1”?

d.) What is the value of “a[0]+1”?

e.) What is the value of “*a[0]+1”? (Hint: * operator has higher precedence compared to + operator.)
f.) What is the value of “*(a[0]+1)”?

**Tracing**

11.) What is the output when the following program is run? Assume sizeof(char) == 1, sizeof(long long) == 8, and sizeof(char *) == 8.

```c
#include <stdio.h>
struct {
    char x;
    long long y;
} a;
union {
    char x;
    long long y;
} b;
char buf[100];

int main()
{
    char *p = buf;
    printf("sizeof(a)=%d\n", sizeof(a));
    printf("sizeof(b)=%d\n", sizeof(b));
    printf("sizeof(buf)=%d\n", sizeof(buf));
    printf("sizeof(p)=%d\n", sizeof(p));
    return 0;
}
```
12.) If the program below, what would be the output?

```c
#include <stdio.h>
void even(int n) { printf("%d even\n", n); } 
void odd(int n) { printf("%d odd\n", n); } 
void (*f[4])(int n);

int main()
{
    int i;
    void (*foo)(int n);
    for(i = 0; i < 4; ++i) {
        if(i % 2)
            f[i] = odd;
        else
            f[i] = even;
    }
    for(i = 0; i < 4; ++i) {
        foo = f[i];
        foo(i);
    }
    return 0;
}
```
Coding

13.) Write a program that asks the user to input a word and then reverses the characters in the word. Match the example output. You can assume the word is less than 100 characters.

For example:

Enter a word: hello
Reverse of hello is olleh

or

Enter a word: world
Reverse of world is dlrow
14.) Write a number search program that takes two command line arguments: 1) length of the array to generate and 2) the number to search within the array. The program should define an array of 100 integers and automatically fill it with the powers of 2 (1, 2, 4, 8, 16 ...) up to the given length and the rest with 0s. You can assume that the given length is less than 100.

This is an example output assuming the name of the program is “search”:

```
./search 10 32
Number 32 found in index 5.
./search 10 50
Number 50 was not found.
```

Part of the program, specifically the search function, has already been implemented for you. Your job is to IMPLEMENT THE MAIN FUNCTION using the given search function as described in the comment. Check that there are exactly two command line arguments and if not, print an error message and return. (Hint: You are free to use the “int atoi(const char *nptr)” function defined in stdlib.h to convert argument strings to integers.)

```c
#include <stdio.h>
#include <stdlib.h>

// [Arguments]
// begin: pointer to the beginning of array
// end: pointer to the end of array
// needle: value that is being searched for
// [Return Value]
// If not found: NULL
// If found: pointer to element of array containing needle value
int* search(int* begin, int* end, int needle)
{
    int *p;
    for(p = begin; p <= end; p++) {
        if(*p == needle) return p;
    }
    return NULL;
}

// IMPLEMENT THIS
int main(int argc, char **argv)
{
    int nums[100] = {0};

    // 1. Check and convert command line arguments using atoi.
    // 2. Fill array “nums” with powers of 2 up to “length” numbers.
    // 3. Use above search(...) function to search array with needle number.
    // 4. Print search result using the value returned by search(...).
    return 0;
}
```