

MIDTERM EXAM

CS 217
October 26, 2000

Name:

Precept:

Honor Code:

Score:

Problem	Score	Max
1		5
2		5
3		5
4		10
5		10
6		15
7		15
Total		65

1. For the arithmetic operation: $75 - 91$, translate the decimal numbers to 2's complement form, complete the arithmetic operation, and translate the result back into decimal. Indicate if overflow occurs. If not, sign extend the 2's complement representation of the result to 16 bits.

2. The following line of code is obtuse. Write an equivalent statement that more clearly states the programmer's intent. Use an accepted idiom if possible.

```
key = key >> (bitoff - (bitoff >> 3) << 3);
```

3. Suppose the following macro is defined:

```
#define isupper(c) ((c) >= 'A' && (c) <= 'Z')
```

Which of the following two uses of this macro is better. Give two reasons why.

- (a) `while (isupper(c=getchar()))`
- (b) `while ((c=getchar()) != EOF && isupper(c))`

4. What is the output of the following program?

```
#include <stdio.h>

main(void) {
    static char *array[3][3]=
        {{":|(", "Hope", "you"},
         {"have", "a", "good"},
         {"fall", "break", ":",|)"}};

    char *(*p)[3]=array;
    char **q=(array+1);

    printf("1 : %s\n",array[0][2]);
    printf("2 : %s\n",*(array[1]));
    printf("3 : %s\n",*(*(array+1)+1));
    printf("4 : %s\n",*(q+2));
    printf("5 : %s\n",*(*(p+2)+1));
}
```

5. The program shown on the next page should output:

```
This is a fall00 midterm question.
his is a fall00 midterm question.T
is is a fall00 midterm question.Th
s is a fall00 midterm question.Thi
 is a fall00 midterm question.This
is a fall00 midterm question.This
s a fall00 midterm question.This i
 a fall00 midterm question.This is
a fall00 midterm question.This is
 fall00 midterm question.This is a
```

However, the program core dumps.

- (a) Identify and fix the bug that makes the program core dump.
- (b) After fixing the bug, explain what problem might arise if `printMatrix` is provided as a library function to be used in big programs.

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef char * CHARBUF;

/* this function prints a 10-line matrix based on the content of
 * argument a. If strlen(a)<10, it will print an error message.
 */
void printMatrix(CHARBUF a)
{
    CHARBUF b[10];
    int i;
    char *to_ptr, *from_ptr;

    if (strlen(a) < 10) {
        fprintf(stderr, "string length must be larger than 10\n");
        return;
    }
    for (i=0;i<10;i++) {
        b[i] = (CHARBUF)malloc(sizeof(char) * (sizeof(a)+1));
        if (b[i]==NULL) {
            fprintf(stderr, "no more memory available.\n");
            return;
        }
        for (to_ptr = b[i], from_ptr = a+i; (*from_ptr) != 0; )
            *to_ptr++ = *from_ptr++;
        for (from_ptr = a; from_ptr != a+i; )
            *to_ptr++ = *from_ptr++;
        *to_ptr = 0;
    }
    for (i=0;i<10;i++)
        printf("%s\n",b[i]);
}

void main(void)
{
    printMatrix("This is a fall00 midterm question.");
}

```

6. Translate each of the following English-language descriptions into a C declaration, and each C declaration to an English-language description.

(a) `static int c = 1;`

(b) `const int i;`

(c) `const int * pc;`

(d) `int * const cp;`

(e) `int *a[10];`

(f) `int **ip;`

(g) `int *f(void *);`

(h) `int (*f)(void *);`

(i) Structure `person` has three components: `name` (a pointer to a structure that contains two components `fname` (a character string), and `lname` (a character string)); `dob` (an array of 3 integers), and `parent` (a pointer to a `person`).

(j) Declare variable `employees` as an array of 10 pointers to structure `person`.

(k) Suppose variable `employees` is a pointer to a `person` structure, and variable `numemps` is an integer. Show the code to dynamically setup `employees` as an array of size `numemps`.

(l) Function `p` that takes `person` as a call-by-reference argument, and returns a generic pointer.

(m) Procedure `map` that takes a pointer `fp` to a function that returns an integer pointer.

(n) Show how you would invoke the function referenced by `fp`, and print out its result.

7. Suppose you are given the type definitions and function interface for a binary tree ADT shown below in file `tree.h`. The function `size` takes a binary tree as an argument and returns the number of nodes in the tree. The function `tree2array` takes a binary tree as argument and returns an array that contains the tree elements (the integer values) listed in depth first traversal order. You need to write the functions `size` and `tree2array`. The `clienttree.c` program (given below) should work with your implementations. Note, however, that your functions should work with ANY size tree, not just trees with 5 nodes. You do not need to write the `insert` function.

File `tree.h`

```
typedef struct tree {
    int val;
    struct tree *left, *right;
} tree;

typedef tree *ptree;

int size (ptree p);
int * tree2array(ptree p);
ptree insert (ptree p, int nval);
```

File `clienttree.c`:

```
#include tree.h;
int main() {
    int * a; /* stores flattened tree */
    int i;
    ptree p = NULL;

    /* insert values into tree */
    p = insert (p, 3);
    p = insert (p, 4);
    p = insert (p, 2);
    p = insert (p, 1);
    p = insert (p, 5);
    a = tree2array(p);
    for (i = 0; i < size(p); I++)
        printf("%d\n", *a++);
    return 0;
}
```