

Midterm Examination – October 11, 2011

Note: This examination has **four** questions for a total of 100 points. **Answer all questions. Write all the answers on your blue book.**

Question I (35 points total)

Note: For Parts (a) through (d) of this question, work must be shown to receive partial credit.

- (a) Convert the decimal integer 259 into octal. (5 points)
- (b) Convert the integer 123_5 to hexadecimal. (5 points)
- (c) Convert the decimal real number 70.15 into binary. (10 points)
- (d) Compute the 2's complement representation of the decimal integer -29 using 8 bits. *Give your answer in hexadecimal form.* (7 points)
- (e) Write a C function `count_digits` with the following header:

```
int count_digits (int x)
```

You may assume that the input parameter `x` is a strictly *positive* decimal integer; that is, it *won't* be negative or zero. Your function must return the number of decimal digits in `x`. For example, if the function is called with the value 1579, the value returned by the function must be 4. You need to show the C code only for the function. There is no need to include comments and you may use magic numbers in your code. (8 points)

Question II (25 points total)

- (a) Indicate the output produced by the following program. There are no syntax errors in the program. (8 points)

```
#include <stdio.h>

int main(void) {
    int r = 5, y = 5;
    printf("r = %d\n", r--);

    while ((r >= 3) && (y < 7)) {
        printf("y = %d\n", y++); --r;
    }

    printf("r = %d\n", r);
    return 0;
} /* End of main */
```

- (b) Indicate the output produced by the following program. There are no syntax errors in the program.
(9 points)

```
#include <stdio.h>
void mystery(int *, int *, int);

int main(void) {
    int p = 21, *q, r = 32;
    q = &p;
    printf("%d %d %d\n", p, *q, --r);
    mystery(q, &r, p);
    printf("%d %d %d\n", p, *q, r); return 0;
} /* End of main. */

void mystery (int *x, int *y, int z) {
    *x = -15; *y = ++(*x); z = *y;
    return;
} /* End of mystery */
```

- (c) Consider following C program. There are no syntax errors in the program.

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

int main(int argc, char *argv[]) {
    printf("%d\n", argc);
    if (argc >= 4)
        printf("%d\n", strlen(argv[3]));
    else
        printf("%d\n", strlen(argv[0]));
    return 0;
} /* End of main */
```

Assume the executable version of the above program is named `a.out`.

- (i) Suppose we run the program using the following command to Unix:

```
a.out ten twenty thirty
```

Indicate the output produced. (4 points)

- (ii) Suppose we run the program using the following command to Unix:

```
a.out csi333
```

Indicate the output produced. (4 points)

Question III (20 points)

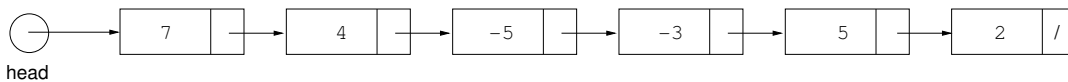
Assume that the following type declaration is available:

```
typedef struct node {
    int key; struct node *next;
} NODE, *PNODE;
```

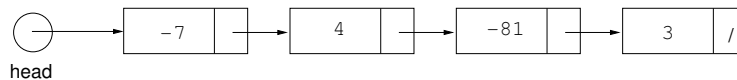
Notice that each node of type `NODE` contains an integer key value and a pointer to the next node of a list. We say that a list containing nodes of the above type is **special** if it satisfies both of the following conditions.

- (i) The list has two or more nodes and
- (ii) it contains two *successive* nodes whose **key** values are both *negative* (i.e., strictly less than zero).

For example, the list



is special while the list



is *not* special.

You are required to write a function `check_list` with the following header:

```
int check_list (PNODE head)
```

Here, the parameter `head` points to the first node of a list each of whose nodes is of type `NODE`. Your function must return the value 1 if the list is special and the value 0 otherwise.

Your need to show only the C code for the above function. There is no need to include comments in your code.

Question IV (20 points)

You are required to write a complete C program to do the following. Your program uses an input file called `"infile.dat"` and creates an output file called `"outfile.dat"`. The input file contains integer values separated by one or more white space characters. We don't know exactly how many integers are in the input file. Your program must read each integer from the input file; if the integer is negative (i.e., it is strictly less than zero) or it is a multiple of the decimal integer 17, then the integer must be written to the output file. (Thus, when the program ends, the output file must contain each integer from the input file which is either negative or a multiple of 17.)

If any error occurs while opening or closing the input or output files, your program must print a suitable error message to `stderr` and exit.

Your program may be written using just one function, namely `main`. There is no need to include comments in your program and you may use magic numbers in your code.