SOFTWARE – Ph.D. Qualifying Exam Fall 2013

SOFTWARE Exam Cover Page.

Do not write on this page.

The rest of this page is intentionally left blank.
(i)  (5 pts.)

Consider the following C program, which includes two function definitions, including the `main` function.

```c
#include <stdio.h>

struct vector {
    int length;
    int *values;
};

int modified_inner_product(struct vector *v1, struct vector *v2) {
    int i = 0, sum = 0;

    if (v2->length < v1->length) {
        printf("Computing in first mode.\n");
        return v1->values[0] * v2->values[0];
    } else {
        printf("Computing in second mode.\n");
        for (i = 0; i < v1->length; i++) {
            sum += (v1->values)[i] * (v2->values)[i];
        }
        while (i < v2->length) {
            sum += (v2->values)[i++];
        }
        return sum;
    }
}

int main(void) {
    int a1[] = {4, 3, -1, 7, 0};
    int a2[] = {7, 2, 1};
    int a3[] = {1, 2, 0, 1, 2, 0};
    struct vector v1 = {5, a1};
    struct vector v2 = {3, a2};
    struct vector v3 = {6, a3};
    int res1 = 0, res2 = 0, res3 = 0;

    res1 = modified_inner_product(&v1, &v2);
    res2 = modified_inner_product(&v1, &v3);
    res3 = modified_inner_product(&v2, &v1);

    printf("results: %d, %d, %d\n", res1, res2, res3);
    return 0;
}
```

Show the complete output as it appears on standard output.
Write your solution to Question (i) on this page. Please clearly indicate your solution and show all work on this page.
(ii)  (5 pts.)

Consider the following C program, which includes two function definitions, including the \texttt{main} function.

```c
#include <stdio.h>
#define ROWS 3
#define COLUMNS 3

void create_values(int x[ROWS][COLUMNS], int *y) {
    int i = 0, j = 0;
    for (i = 0; i < ROWS; i++) {
        for (j = 0; j < COLUMNS; j++) {
            x[i][j] = y[ROWS * COLUMNS - (i + j + 1)];
        }
    }
}

int main(void) {
    int i = 0, j = 0, iterations = 0;
    int x[ROWS][COLUMNS];
    int y[] = {2, -3, 3, 4, -1, 7, 17, -2, 4};
    create_values(x, y);
    for (i = 0; i < ROWS; i++) {
        for (j = 0; j < COLUMNS; j++) {
            printf("%d, ", x[i][j]);
        }
        printf("\n");
    }
    iterations = 0;
    for (i = 0; i < ROWS; i++) {
        for (j = 0; j < COLUMNS; j++) {
            if (x[i][j] > 0) {
                j++;
            } else {
                i++;
            }
            iterations++;
        }
    }
    printf("The iteration count is: %d\n", iterations);
    return 0;
}
```

Show the complete output as it appears on standard output.
Write your solution to Question (ii) on this page. Please clearly indicate your solution and show all work on this page.
(iii) (5 pts.)

Consider the following C program, which includes two function definitions, including the `main` function.

```c
#include <stdio.h>

struct element {
    char c;
    struct element *x;
    struct element *y;
};

void traverse(struct element *element, int i) {
    int j = 0;
    struct element *p = element;
    for (j = 0; j < i; j++) {
        printf("%d: \%c\n", j, p->c);
        if (j % 3 == 0) {
            p = p->y;
        } else {
            p = p->x;
        }
    }
}

int main(void) {
    const int length = 3;
    struct element collection[length];
    char s[] = "Maryland";
    int i = 0, j = 0;
    for (i = 0; i < length; i++) {
        j = (i + 1) % length;
        printf("j = %d\n", j);
        collection[i].x = &collection[j];
    }
    for (i = 0; i < length; i++) {
        j = (i * i) % length;
        printf("j = %d\n", j);
        collection[i].y = &collection[j];
    }
    for (i = 0; i < length; i++) {
        collection[i].c = s[2 * i];
    }
    traverse(collection, 5);
    return 0;
}
```

Show the complete output as it appears on standard output.
Write your solution to Question (iii) on this page. Please clearly indicate your solution and show all work on this page.
(iv) (5 pts.)

Consider the following function prototype and associated header comment.

```c
//******************************************************************************
// Perform a specified operation on two given strings, s1 and s2, and display the result on standard output. The name of the specified operation is specified by the string pointed to by the "operation" argument. The following are the valid (case-sensitive) operation names: "compare", "concatenate", "flip", and "measure". If the specified operation name is not one of these valid names, then print a meaningful error message to standard error. Otherwise, print to standard output the result of applying the specified operation to the given pair of strings. The result of the "compare" operation is the string "same" if s1 and s2 are identical, and is the string "different" if they are not identical. The result of the "measure" operation is the sum of the string lengths of (numbers of characters in) s1 and s2, excluding their terminating null characters. The result of the "concatenate" operation is the string s1 followed by s2 with no space in between them. Similarly, the result of the "flip" operation is the string s2 followed by s1 with no space in between them.
******************************************************************************
void stringops(char *s1, char *s2, char *operation);
```

Note that no error checking is required in this function beyond what is stated in the header comment.

**Part A:** Consider the C program below, which illustrates correct usage of the function `stringops`.

```c
int main(void) {
    char *s1 = "hello";
    char *s2 = "goodbye";

    stringops(s1, s2, "measure");
    stringops(s1, s2, "compare");
    stringops(s1, s2, "concatenate");
    stringops(s1, s2, "flip");
}
```

Show the complete output of this program as it appears on standard output.

**Part B:** Develop a complete C code implementation of the function `stringops`. In your implementation, make appropriate use of features that are provided through the standard library header file `<string.h>`.
Write your solution to Question (iv) on this page. Please clearly indicate your solution and show all work on this page.
Software Qualifying Exam Solutions
Fall 2013
Dept. of ECE, University of Maryland, College Park
05/19/2013

Problem 1:
Computing in first mode.
Computing in second mode.
Computing in second mode.
results: 28, 17, 40

Problem 2:
4, -2, 17,
-2, 17, 7,
17, 7, -1,
The iteration count is: 4

Problem 3:
j = 1
j = 2
j = 0
j = 0
j = 1
j = 1
0: M
1: M
2: r
3: l
4: r

Problem 4, Part A:
12
different
hellogoodbye
goodbyehello
void stringops(char *s1, char *s2, char *operation) {
    int length = 0;

    if (strcmp(operation, "compare") == 0) {
        if (strcmp(s1, s2) == 0) {
            printf("same\n");
        } else {
            printf("different\n");
        }
    } else if (strcmp(operation, "measure") == 0) {
        length = strlen(s1) + strlen(s2);
        printf("%d\n", length);
    } else if (strcmp(operation, "concatenate") == 0) {
        printf("%s%s\n", s1, s2);
    } else if (strcmp(operation, "flip") == 0) {
        printf("%s%s\n", s2, s1);
    } else {
        fprintf(stderr, "invalid strops operation\n");
    }
}