

School of Computer Science and Software Engineering
Clayton Campus, Monash University

CSE1303 Part A
Summer Semester, 2002

Tutorial 1: Revision
Solutions

Exercise 1.

```
a = {  
    {0, 1, 1, 1},  
    {0, 0, 2, 2},  
    {0, 1, 0, 3},  
    {0, 0, 1, 0}  
};
```

Exercise 2.

```
/*  
 * Given a positive total number of secs, totalSecs,  
 * return the equivalent number of seconds, minutes,  
 * and hours.  
 */  
void  
getTime(int totalSecs, int* secs, int* minutes, int* hours)  
{  
    if (totalSecs < 0)  
    {  
        fprintf(stderr, "Total number of seconds < 0\n");  
        exit(1);  
    }  
  
    *minutes = totalSecs/60;  
    *hours = *minutes/60;  
    *minutes %= 60;  
    *secs = totalSecs % 60;  
}
```

```
/* Write a main function in C which implements a menu
 * that prompts the user to chose to enter a positive
 * number of seconds or exit the program.
 */

void main()
{
    int cont = 1;
    int choice = 0;
    int total = 0;
    int secs;
    int minutes;
    int hours;

    while (cont == 1)
    {
        printf("Press 1 to convert seconds to ");
        printf("hours/minutes/seconds.\n");
        printf("Press any other key to exit ");
        printf("this program.\n");
        scanf("%d", &choice);

        if (choice != 1)
        {
            cont = 0;
        }

        else
        {
            printf("\nPlease enter a positive number");
            printf(" of seconds: ");
            scanf("%d", &total);
            if (total < 0)
            {
                printf("\nA positive number please");
            }
            else
            {
                getTime(total, &secs, &minutes, &hours);

                printf("hours: %d minutes: %d seconds:
                %d\n", hours, minutes, seconds);
            }
        }
    }
    return;
}
```

Exercise 3.

```
/* Write fragments of C code showing how you might use the
 * string library functions strcmp(), strcpy() and strlen()
 */
char string1[100];
char string2[200];
int count;
int cmp;

strcpy(string1, "Apple");
strcpy(string2, string1);

count = strlen(string1);

cmp = strcmp (string1, string2);
if (cmp == 0)
{
    printf("Strings are the same\n");
}
else if (cmp < 0)
{
    printf("string1 sorts before string2\n");
}
else
{
    printf("string1 sorts after string2\n");
}
```

Exercise 4

```
/* Reverse a character string. */
void
reverse(char* s)
{
    int length = 0;
    int i;
    char tmp;

    /* Find the length of the character string */
    while (s[length] != '\0')
    {
        length++;
    }

    for (i = 0; i < length/2; i++)
    {
        tmp = s[i];
        s[i] = s[length-i-1];
        s[length-i-1] = tmp;
    }
}
```

Exercise 5

```
char next = 'A';
char current = 'g';
char *ptr;

ptr = &current;
*ptr = 'X';

printf("%p", ptr);

ptr = &next;
*ptr = 'd';

next is 'd'
current is 'X';
```

Exercise 6

```
/*
 * Sorts an array of students into alphabetical order
 * according to their names.
 */
void
sort(Student class[], int size)
{
    int j;
    int k;
    int index;
    char* last = NULL;
    Student tmp;

    for (k = size-1; k > 0; k--)
    {
        /*
         * Find the last name in the first k students
         * in the class.
         */
        index = 0;
        last = class[0].name;
        for (j = 1; j <= k; j++)
        {
            if (strcmp(class[j].name, last) > 0)
            {
                /*
                 * Keep track of the last name and
                 * and where it is.
                 */
                last = class[j].name;
                index = j;
            }
        }
        /*
         * Swap the record of the student with the last name
         * with the record of the kth student.
         */

        tmp = class[index];
        class[index] = class[k];
        class[k] = tmp;
    }
}
```

```
/*
 * Prints the array of structs to a file: classlist.txt
 */

void printclass(Student class[], char *filename, int size)
{
    FILE* outfile;
    int i;
    if((outfile = fopen(filename, "w")) == NULL)
    {
        printf("An error occurred opening the file.\n");
        return;
    }
    for (i=0; i < size; i++)
    {
        fprintf(outfile, "%s - %d", class[i].name, class[i].mark);
    }
    fclose(outfile);
    return;
}
```

Exercise 7.

```
/*
 * Compares character strings s and t, and returns a
 * negative value, zero, or a positive value if s is
 * lexicographically less than, equal to, or greater than
 * t, respectively.
 */
int
strcmp(char* s, char* t)
{
    for (; *s == *t; s++, t++)
    {
        if (*s == '\0')
        {
            return 0;
        }
    }

    return *s - *t;
}
```