Duration: 50 minutes
Aids Allowed: One 8.5 x 11 inch paper

Student Number: 
Last (Family) Name: SOLUTION
First (Given) Name(s): 

Tutorial Section: BA–1200 BA–2179
(circle one)
Xiaoyang Madalin
Guan Mihaliescu

Do not turn this page until you have received the signal to start.
(In the meantime, please fill out the identification section above,
and read the instructions below carefully.)

This term test consists of 5 questions on 6 pages (including this one). When you receive the signal to start, please make sure that your copy of the test is complete.

Answer each question directly on the test paper, in the space provided. If you need more space for one of your solutions, use the extra page at the end. Indicate clearly the part of your work that should be marked.

General Hint: We were careful to leave ample space on the test paper to answer each question.

Marking Guide

# 1: ____/ 6
# 2: ____/ 5
# 3: ____/ 6
# 4: ____/ 6
# 5: ____/ 7
TOTAL: ____/30

Good Luck!
Question 1. [6 marks]

The current working directory contains 3 files: coffee, milk, and tea. The contents of each file are shown below:

```markdown
<table>
<thead>
<tr>
<th>coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat milk</td>
</tr>
<tr>
<td>cat coffee</td>
</tr>
<tr>
<td>tea</td>
</tr>
<tr>
<td>cream</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>milk</th>
</tr>
</thead>
<tbody>
<tr>
<td>cream</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tea</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat sugar</td>
</tr>
<tr>
<td>cat coffee</td>
</tr>
</tbody>
</table>
```

The program listing for `drinks.sh` is shown below (' is a single quote, and ' is a backquote):

```bash
#!/bin/sh

echo "Part 1:"
for x in $*
do
grep $x *
done

echo "Part 2:"
grep "cat $1" coffee
grep 'cat $1' coffee
grep 'cat $1' coffee
```

Fill in the missing parts of the output when the program is run as `drinks.sh milk cream sugar`.

If a `grep` statement would produce no output, explain why.

Part 1:
- coffee: cat milk
- coffee: cream
- milk: cream
- tea: cat sugar

Part 2:
- cat milk
- cream

The second `grep` does not produce output because there is no string "cat $1" in coffee.
Question 2. [5 marks]
Assume that the program below runs to completion without errors:

```c
int main() {
    int result;
    printf("A\n");
    if((result = fork()) > 0 ) {
        printf("B\n");
    } else if(result == 0) {
        printf("C\n");
        execlp("runprog", 0);
        printf("D\n");
    } else {
        perror("fork");
        exit(1);
    }

    printf("F\n");
    wait(0);
    printf("Done\n");
    return 0;
}
/* runprog.c */
int main() {
    printf("E\n");
    return 0;
}
```

Part (a) [1 mark]
Which, if any, `printf` lines are executed more than once?

None of the `printf` lines are executed more than once.

Part (b) [4 marks]
Write down the output of this program in 4 different correct orderings.

```
A A A A A A A
B B B C C C C
C C F E E B B
E F C B E F F
F E E F F E E
Done Done Done Done Done Done
```
Question 3.  [6 marks]
The following declarations are used for both parts of this question.

```c
char *a = "Firefox";
char *b = "Safari";
char *c = "IE 7";
char **p;
char *files[3];
```

Part (a)  [3 marks]
Give the type and value of each of the following 3 expressions. If an error would occur, explain what the error is.

There was a typo in the last one. I meant to write *(p+1) which would have been an error. Naturally, it was marked as printed.

```c
p = &a;

p[0][3]  char  'e'

(*p)+1       char*    irefox

(*p+1)     char*    irefox
```

Part (b)  [3 marks]
For each of the following, either explain the problem that occurs or state that the sequence is correct. (The variables a, b, c are defined above.)

<table>
<thead>
<tr>
<th>Expression</th>
<th>Problem/Correctness</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>files[0] = malloc(8)</code></td>
<td>memory leak</td>
</tr>
<tr>
<td><code>files[0] = a</code></td>
<td></td>
</tr>
<tr>
<td><code>files[1] = NULL</code></td>
<td>okay</td>
</tr>
<tr>
<td><code>files[1] = b</code></td>
<td></td>
</tr>
<tr>
<td><code>files[2] = NULL</code></td>
<td></td>
</tr>
<tr>
<td><code>files[2] = strncpy(files[2], c, 5)</code></td>
<td>No memory allocated to copy into.</td>
</tr>
<tr>
<td><code>files[2] = strcpy(files[2], c, 5)</code></td>
<td></td>
</tr>
</tbody>
</table>
Question 4. [6 marks]
Without using any string functions, write the function `strncat`, described below:

DESCRIPTION

The `strncat()` function appends up to count characters of the null-terminated string `t` to the end of the null-terminated string `s`, then adds a terminating `\0`. The string `s` must have sufficient space to hold the result.

```c
char * strncat(char *s, const char *t, size_t count) {
    int i;
    char *result = s;
    while(*s != '\0') {
        s++;
    }
    for(i = 0; i < count; i++) {
        *s = t[i];
        if(t[i] == '\0') {
            return result;
        }
        s++;
    }
    *s = '\0';
    return result;
}
```
Question 5. [7 MARKS]

Suppose you are given a file in the following format. Each line of the file starts with a directory name followed by a space and a colon. The remaining elements of the line are the names of files in that directory.

Write a Bourne shell program that takes a file in the above format. For each line of the file:

```
dir : f1 f2
```

it prints a message if `dir` is not a directory in the current working directory, and it prints a message if any of the files `f1`, `f2` (etc.) are not files in the directory `dir`.

```bash
#!/bin/sh
while read line
do
  set $line
  dir=$1
  if [ ! -d $dir ]; then
    echo $dir is not a directory
  else
    shift
    shift
    while [ $1 ]
do
      if [ ! -f $dir/$1 ]; then
        echo $dir/$1 is not a file
      fi
      shift
done
  fi
done < $1
```