Name:_____

CSE 303, Winter 2007, Midterm Examination
9 February 2007

Please do not turn the page until everyone is ready.

Rules:

- The exam is closed-book, closed-note, except for one 8.5x11in piece of paper (both sides).
- **Please stop promptly at 1:20.**
- You can rip apart the pages, but, if you do, please write your name on each page.
- There are **65 points** total, distributed **unevenly** among 6 questions (many of which have multiple parts).

Question	Max	Grade
1	8	
2	9	
3	15	
4	12	
5	12	
6	9	
Total	65	

Advice:

- Read questions carefully. Understand a question before you start writing.
- **Write down thoughts and intermediate steps so you can get partial credit.**
- The questions are not necessarily in order of difficulty. **Skip around.**
- If you have questions, ask.
- Relax. You are here to learn.

Name:_____

1. (**8** points) Consider the following subdirectory structure inside a user's home directory:

```
documents/
documents/friends/
documents/friends/birthdays.txt
documents/friends/emails.txt
documents/friends/notes.txt
documents/home/
documents/home/bills.txt
documents/home/todo.txt
documents/office/
documents/office/report.doc
documents/office/sales.xls
```

   (a) (**4** points) The user executes the following sequence of commands. What is the final content of the file `resultA`:

```
cd ~/documents/
ls home > intermediate
ls office >> intermediate
sort intermediate > resultA
```

   Solution:
   The content of file `resultA` is:

```
bills.txt
report.doc
sales.xls
todo.txt
```

Name:_____

(b) (**4** points) The user executes the following sequence of commands. What is the final content of the file `resultB` (Reminder: `wc -w` will read from `stdin` and will print the number of words that it read):

```
cd ~/documents/friends/
mv notes.txt reminders.txt
rm emails.txt
ls | wc -w > ../resultB
```

Solution:
The content of file `resultB` is the number 2

Name:_____

2. (**9** points) What does each one of the following commands do? Please circle the appropriate answer.

   (a) `egrep '[0-9][0-9]$' input`

   i. Searches the file `input` for lines containing at least two digits.
   ii. Searches the file `input` for lines containing at most two digits.
   iii. Searches the file `input` for lines that end with two consecutive digits.
   iv. Searches the file `input` for lines that only hold two digits and nothing else.

   Solution:
   (iii)

   (b) `egrep 'hello.world' input`

   i. Searches the file `input` for lines containing the string "hello world"
   ii. Searches the file `input` for lines containing the string "hello.world"
   iii. Searches the file `input` for lines containing the words "hello" and "world" in that order, separated by zero or more other characters.
   iv. Searches the file `input` for lines containing the words "hello" and "world" in that order, separated by exactly one character.

   Solution:
   (iv)

   (c) `sed -n 's/Java/C/gp' input`

   i. For each line in file `input`, replaces all occurrences of the string `Java` with the character `C`. Writes only the affected lines to `stdout`. Does not modify the original file.
   ii. For each line in file `input`, replaces all occurrences of the string `Java` with the character `C`. Writes only the affected lines to `stdout`. Modifies the original file.
   iii. For each line in file `input`, replaces all occurrences of the string `Java` with the character `C`. Writes only the count of affected lines to `stdout`. Does not modify the original file.
   iv. For each line in file `input`, replaces all occurrences of the string `Java` with the character `C`. Writes only the count of affected lines to `stdout`. Modifies the original file.

   Solution:
   (i)

Name:_____

3. (**15** points) Consider the following bash script. Indicate the output of this script using the provided blanks. Assume the script is run as follows:

```
./scrip.sh script-input.txt
```

where the file `script-input.txt` contains the following sequence of digits: `1 1 2 3 4 5 4 3 2 1`

```
#!/bin/bash

if [ $# -lt 1 ]
then
    echo "usage: ‘basename $0‘ input_file" >&2
    exit 1
fi

x=0
for i in ‘cat $1‘
do
  let "array[i] = 0"
  if [ $x -lt $i ]
  then
      let "x = i"
  fi
done

echo "$x"                                # Output: _________________

y=0
for i in ‘cat $1‘
do
  let "array[i] = array[i] + 1"
  let "y = y + 1"
done

echo "$y"                                # Output: _________________

i=1
while [ $i -lt $((x+1)) ]
do
  echo "$i ${array[i]}"                  # Output: Please write the output below
  let "i = i+1"
done
```

Solution:

```
5
10
1 3
2 2
3 2
4 2
5 1
```

Name:_____

4. (**12** points) Consider the following function:

```
void to_uppercase(char *dst, char *src) {

  int i;
  for ( i = 0; i <= strlen(src); i++) {
    dst[i] = toupper(src[i]);
  }

}
```

where `int toupper(int c)` returns the upper-case letter corresponding to the given lower-case letter. If the character given as argument is not a lower-case letter, `toupper` returns that character unchanged.

(a) (**6** points) Next to each one of the following sequences of statements, indicate if the sequence is correct of if there is a bug. If there is a bug, **fix** the bug.

```
// Sequence 1
char string1[] = "Hello";
char *string2;
to_uppercase(string2,string1);
printf("%s and %s\n",string1, string2);
```

```
// Sequence 2
char string3[] = "Hello";
to_uppercase(string3,string3);
printf("%s\n",string3);
```

Name:_____

(b) (**6** points) The `to_uppercase` function makes several assumptions about its arguments. Write a sequence of statements that use this function in a manner that may "set the computer on fire" (i.e., that may cause the program to read or write invalid memory locations). One example would be to pass `NULL` as either one of the arguments. Please find another example. Your example must also be different from the two code snippets shown in the previous question.

Solution:

(a)
- Sequence 1 has a bug. No space is allocated for the destination string, `string2`. The code snippet must be fixed as follows:

```
// Sequence 1
char string1[] = "Hello";
char string2[strlen(string1)+1];
to_uppercase(string2,string1);
printf("%s and %s\n",string1, string2);
```

- Sequence 2 is correct.

(b) For this question, several answers are possible. Here is one where we both (1) provide a destination buffer that is too small causing the system to overwrite other items on the stack; and (2) overwrite the null character of the source string causing the system to read through memory and translate characters until it finds another null character:

```
char source[] = "Hello";
source[strlen(source)] = ' ';
char destination[1];
to_uppercase(destination,source);
```

Name:_____

5. (**12** points) Indicate the output of the program using the provided blanks.

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

void mystery(int **pp1, int **pp2) {

  int *temp = *pp1;
  *pp1 = *pp2;
  *pp2 = temp;

}

void main() {

  int *p1 = NULL;
  int *p2 = NULL;
  int *p3 = NULL;

  p1 = (int*)malloc(sizeof(int));
  *p1 = 5;
  p3 = p1;

  p2 = (int*)malloc(sizeof(int));
  *p2 = *p3;

  *p2 = (*p2)+1;

  printf("%d %d %d\n", *p1, *p2, *p3);    // Output: ____________________________

  mystery(&p1,&p2);

  printf("%d %d %d\n", *p1, *p2, *p3);    // Output: ____________________________

  free(p1);
  free(p2);
  p1 = NULL;
  p2 = NULL;
  p3 = NULL;

  return 0;
}
```

Solution:

```
5 6 5
6 5 5
```

Name:_____

6. (**9** points) The file `preprocessor.c` contains the following C code:

```
#include <stdio.h>

#ifdef BIGGER
#define DELTA 100
#else
#define DELTA 10
#endif

#define ADJUST(x)  (x) + DELTA

int main() {

  int v1 = 1;
  printf("v1: %d\n", ADJUST(v1));

  int v2 = 10;
  int result = 2 * ADJUST(v2);
  printf("v2: %d\n", result);

  int v3 = 20;
  result = ADJUST(v3+1) * ADJUST(v3+1);
  printf("v3: %d\n", result);

  return 0;
}
```

Indicate the output of the program if we compile and execute it as follows. Warning: be careful!

```
gcc -g -Wall -o preprocessor preprocessor.c
./preprocessor -D BIGGER
```

```
v1: __________________

v2: __________________

v3: __________________
```

Solution:

```
v1: 11
v2: 30
v3: 241
```