CS16 Midterm Exam 1
E01, 10S, Phill Conrad, UC Santa Barbara
Wednesday, 04/21/2010, 1pm-1:50pm

Name: _____

Umail Address: _____@ umail.ucsb.edu

Circle Lab section: 9am 10am 11am noon

(Link to Printer Friendly-PDF version)
Please write your name **only** on this page. That allows me to grade your exams without knowing whose exam I am grading.

This exam is **closed book, closed notes**, **closed mouth, cell phone off**, except for:

- You are permitted **one sheet of paper** (max size 8.5x11") on which to write notes
- These sheets will be collected with the exam, and might not be returned
- Please write your name on your notes sheet

There are 100 points worth of questions on the exam, and you have 75 minutes to complete the exam.

A hint for allocating your time:

- if a question is worth 10 points, spend no more than 5 minutes on it
- if a question is worth 20 points, spend no more than 10 minutes on it
- etc.

You will then complete the exam in 50 minutes, and have 25 minutes remaining to check your answers, or go back and work on problems you were unable to complete the first time through.

---

1. (20 pts) In lab03, you were given an example of a function `starL` that produced ASCII art like this:

Function call	Output	Function call	Output	Function call	Output

starL(3,4)	`*`
	`*`
	`*`
	`***`
		starL(3,5)	`*`
			`*`
			`*`
			`*`
			`***`
				starL(5,3)	`*`
					`*`
					`*****`

And you had an assignment to write a function starT that produced output like this:

Function call	Output	Function call	Output	Function call	Output

starT(3,4)	`***`
	`  *`
	`  *`
	`  *`
		starT(3,5)	`***`
			`  *`
			`  *`
			`  *`
			`  *`
				starT(5,3)	`*****`
					`   *`
					`   *`

Your job now is to write a function that produces output like this:

Function call	Output	Function call	Output	Function call	Output

starC(3,4)	`***`
	`*`
	`*`
	`***`
		starC(3,5)	`***`
			`*`
			`*`
			`*`
			`***`
				starC(5,3)	`*****`
					`*`
					`*****`

On the next page, you'll find the entire source code for the starC.c program, except for the body of the function definition of starC. Please fill in that body.

Be sure to read the comments (in **`bold`**) immediately before the function definition.

To help you, along with this exam, you should have recieved a handout with the entire source code of the `starL.c` program.

Put your answer on the next page

Your answer to the starC.c problem goes here—fill in the function body

```
// starC.c   A demonstration of ASCII Art printing C characters
// Exam question for CS16, Spring 2010, UCSB


#include <stdio.h>
#include <stdlib.h>
void starC(int width, int height); // function prototype

int main(int argc, char *argv[])
{
  int width, height;

  if (argc!=3)
    {
      printf("Usage: ./starC width height\n");
      return 1;
    }

  // get command line params
  width = atoi(argv[1]);
  height = atoi(argv[2]);

  starC(width,height);   // call the function

  return 0;
}

// A C function that prints the letter C with stars,
// at any width or height, provided both width and height are >= 3
// If either is less than 3, the function should print nothing.)

void starC(int width, int height)
{


  // your answer goes here









}
```

2. (15 pts) Write the definition of a C function `ratioSquares` that takes two parameters with the names *a* and *b*, both of type double.

   The function should return the value of: $\frac{a^2}{b^2}$

   Write ONLY the function definition—for this question, I do NOT want a complete C program, so do NOT include any extraneous stuff such as #include <stdio.h> or a main function.

   Be sure that you write your answer using only valid C—not in math notation.

3. (20 pts) Write a complete main program that calls the ratioSquares function you wrote for question 1, allowing the person running the program to specify values for *a* and *b*.

   You have two choices:

   - Use `printf`/`scanf` to ask the user to input values for `a` and `b`—***OR***
   - Get the values of `a` and `b` from the command line (using `argc`, `argv`)

   Either method is ok, but choose one and stick with it—don't mix the two.

   Your program should then pass the values of `a` and `b` to a function call to `ratioSquares`
   Store the value returned in a variable of type double called `result`.
   Then print the value of `result` with this line of code:

```
printf("Result is %lf\n", result);
```

   See handout that comes with this exam for sample output and additional hints.

   You do not need to recopy your entire function definition from the previous question—wherever you would have written that, to save time, you can just write a comment that says:

```
// function definition for ratioSquares goes here
```

4. (20 pts) For each of the following C expressions, fill in the value, and then circle the type. The first two are done for you as an example.

   Don't forget to fill in BOTH the value AND circle the type.

expression	value	type
3	3	[int] double char char *
0.5 * 0.5	0.25	int [double] char char *
17 % 5		int double char char *
23 % 100		int double char char *
4 + 7 * 2		int double char char *
2 + 1 / 5.0		int double char char *
6.7/10		int double char char *
"6 % 2"		int double char char *
'7'		int double char char *
3/5		int double char char *
5/3		int double char char *
16 % 2		int double char char *

5. (16 pts) For each of the for loops below:
   a. Circle **infinite** if it is an infinite loop, or **finite** if it NOT an infinite loop
   b. Check the in the **no output column (☑)** if the loop has no output
   c. If the loop has output, put it in the box.
   Note: if the output will be infinite, just write the **output of the first three times through the loop**, then put three dots like this: ...

code	Infinite or finite?	No output?	write the output here (if any)

`int i;`
`for (i=0; i<4; i--)`
`  printf("%d ",i);`
	infinite finite	☐	

`int i;`
`for (i=4; i>1; i--)`
`  printf("%d ",i);`
	infinite finite	☐	

`int i;`
`for (i=1; i>=4; i++)`
`  printf("%d ",i);`
	infinite finite	☐	

`int i;`
 	
`for (i=1; i<=4; i++)`
`  printf("%d ",i);`
	infinite finite	☐	

6. Suppose we have a program where the main starts with the line:

```
int main(int argc, char *argv[])
```

and the program is run with the following command line:

```
./myprog 12 dozen eggs
```

   a. (3 pts) What is the value of `argc`?

   b. (3 pts) What is the value of `argv[2][2]`

   c. (3 pts) If you had a variable of type `int` called `quantity`, how could you assign it to the value that immediately follows the program name (in this case it is `12`)?

End of Exam

Total Points: 100

CS16 Midterm Exam 1
E01, 10S, Phill Conrad, UC Santa Barbara

Handout with starL.c

```
// starL.c   A demonstration of ASCII Art printing L characters
// P. Conrad for CS16, Winter 2010, UCSB

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

void starL(int width, int height);




int main(int argc, char *argv[])
{
  int width, height;

  if (argc!=3)
    {
      printf("Usage: ./starL width height\n");
      return 1;
    }

  // remember argv[0] is the name of the program
  width = atoi(argv[1]);
  height = atoi(argv[2]);

  // call the function
  starL(width,height);

  return 0;
}

// A C function that prints the letter L with stars,
// at any width or height, provided both width and height are >= 2
// If either is less than 2, the function should print nothing.)


void starL(int width, int height)
{
  int i;

  // check if parameters are valid
  if ((width<2) || (height < 2))
    return;  // return without printing anything

  // print height-1 rows of *\n
  for (i=0; i<height-1; i++)
    printf("*\n");

  // print width stars, followed by a final \n
  for (i=0; i<width; i++)
    printf("*");

  printf("\n");

  return;   //  we are finished
}
```

CS16 Midterm Exam 1
E01, 10S, Phill Conrad, UC Santa Barbara

Question 2

Sample output

bold indicates what the user types
grey indicates the Unix prompt.
Everything else is output from the program.

Sample Output (printf/scanf method)	**Sample Output** (argc, argv method)
-bash-2.05b$ **./calcSR** Enter a: **2.0** Enter b: **4.0** Result is 0.25 -bash-2.05b$	-bash-2.05b$ **./calcSR 2.0 4.0** Result is 0.25 -bash-2.05b$

Hints for full credit

- Use correct % specifiers (%d or %i for int, and %lf for double)
- Remember the special syntax detail that applies to using scanf.
- Remember your #includes
- Use "function prototypes" when function calls precede function definitions
- When the main() is finished, remember to signal success/failure to the OS