

Name		ID		Class ID	
Date				Time	

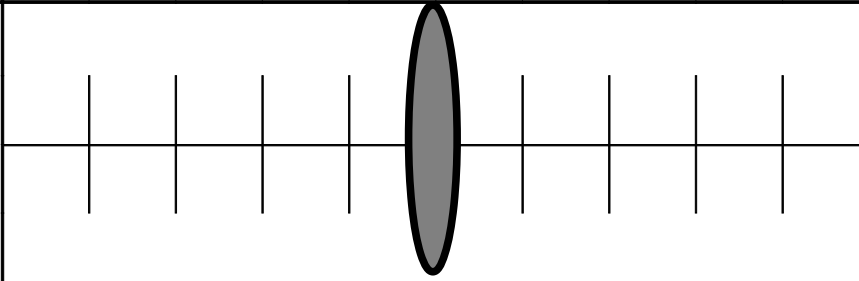
OPTICAL RAY TRACING

Table of Contents

Optical Ray Tracing.....	1
1 (Ray Tracing) Simple Lens Formula: Hand Calculation	2
2 (Ray Tracing) Preparing to Code the Simple Lens Formula: UML.....	3
3 (Ray Tracing) Simple Lens Formula: Direct Application Code.....	4
3.1 Commented Code.....	4
3.2 ANSI-C Code	4
3.3 Calculation with Code.....	4
4 (Ray Tracing) Simple Lens Formula w/ Coordinate System.....	5
4.1 Thinking – Hand Calculation (2 points).....	5
4.2 Preparing to code: Draw the UML Activity Diagram.....	5
4.3 Commented Code (save as mopt2com.c).....	5
4.4 ANSI-C Code (save as mopt2flt.c).....	5
4.5 Run Code to make sure calculation is correct.	5
5 (Ray Tracing) Using Pointers to get Input Data.....	6
5.1 Write the C-CODE to implement the input functions.	6
5.2 Use your program to check a few cases.....	6
6 (Ray Tracing) Conditional Flow, Cleaning & Repetition.....	7
6.1 Calculations with current version of code.	7
6.2 Correct the C-CODE to give correct answers for both cases.	7
7 (Ray Tracing) Split File & Add Repetition For Multiple Lenses	8
7.1 Split the code into three separate files.....	8
7.2 Modify the code to handle a multiple lens systems.....	8
7.3 Use your code to find the final image position and height. (units = mm).....	8
8 (Ray Tracing) Arrays: Separating Input and Output Phases.....	9
8.1 Modify existing UML diagram to separate input, calculation and output.	9
8.2 Modify the program mo8loop.c. Call the new program mo9arr.c.	9
8.3 Use your program to obtain the final image position/height after 4 lens	9
8.4 Open the output file “table.csv” in Microsoft Excel or OpenOffice Calc	9
9 (Ray Tracing) Records: Linking the data.....	10
9.1 Replace the four arrays with two arrays of records.....	10
9.2 Create a new function getOptR() to replace getOpt().	10
9.3 Check to make sure that your program is running correctly.....	10

Name		ID		Class ID	
Date		Time			

1 (Ray Tracing) Simple Lens Formula: Hand Calculation

o	f	i	m	Picture	
2	+ 3				
6	+ 3				
2	- 3				
6	- 3				

When you have completed the work, ask the TA to stamp. Then please help other students.	TA Stamp
---	-----------------

Name		ID		Class ID	
Date		Time			

2 (Ray Tracing) Preparing to Code the Simple Lens Formula: UML

Based on your hand calculations, summarize the procedure using a UML Diagram.

When you have completed the UML diagram. Ask a TA to stamp.	TA Stamp
---	-----------------

Name		ID		Class ID	
Date		Time			

3 (Ray Tracing) Simple Lens Formula: Direct Application Code

3.1 Commented Code

Write a commented version (not real C-code) based on the UML presented in class. When your code compiles without error, ask a TA to stamp.	TA Stamp
--	----------

3.2 ANSI-C Code

Write the C-CODE for solving the problem. When your code compiles, linked and runs without error, ask a TA to stamp	TA Stamp
---	----------

3.3 Calculation with Code

o	f	i	m	picture	TA Stamp
2	3				
6	3				
2	-3				
6	-3				
3	3				
3	-3				

Once you have completed this worksheet and have TA to stamp, you may either Go home, or, Help your fellow students to solve the problems with their code. I hope that you will choose the second option. This will not only help your friends but also help yourself in identifying coding errors	TA Sign
---	---------

Name		ID		Class ID	
Date		Time			

4 (Ray Tracing) Simple Lens Formula w/ Coordinate System

4.1 Thinking – Hand Calculation (2 points)

object		lens		Calculate		image		Picture											
x	h	x	f	o	i	x	h												
0	1	4	+2																
1	1	4	+2																

4.2 Preparing to code: Draw the UML Activity Diagram

4.3 Commented Code (save as mopt2com.c)

Write a commented version of the code (not real C-code). When your code compiles without error, ask the TA to stamp.	TA Sign
--	----------------

4.4 ANSI-C Code (save as mopt2flt.c)

Write the C-CODE for solving the problem. When your code compiles, linked and runs without error, ask the TA to stamp.	TA Sign
--	----------------

4.5 Run Code to make sure calculation is correct.

object		lens		Calculate		image		TA initialize							
x	h	x	f	o	i	x	h								
0	1	4	+2												
1	1	4	+2												

Name		ID		Class ID	
Date				Time	

5 (Ray Tracing) Using Pointers to get Input Data

5.1 Write the C-CODE to implement the input functions.

Replace `getFloat()` in `jlib.h` with your own subroutines to retrieve input data. One subroutine should prompt the user for the object information: position and height. One subroutine should prompt the user for the lens information: position and focal length.

Note that you will need to use pointers in the subroutine as you are trying to return two pieces of information and not just one piece of information with each function call.

Write down a copy of your new version of `getLens()` here:

5.2 Use your program to check a few cases.

This is to check that after changing your code the calculation remains correct.

object		lens		Calculate		image		TA initialize
x	h	x	f	o	i	x	h	
0	1	4	+ 2					
1	1	4	+ 2					

Name		ID		Class ID	
Date		Time			

6 (Ray Tracing) Conditional Flow, Cleaning & Repetition

6.1 Calculations with current version of code.

Download the pointer version (mo5ptr.c). Are the results reasonable?

ox	oh	lx	lf	o	i	ix	ih	picture
0	1	5	+ 5					
10	1	5	+ 5					

6.2 Correct the C-CODE to give correct answers for both cases.

Note that you will need to use the “IF and ELSE” statement to modify the function containing the thin lens equation.

ox	oh	lx	lf	o	i	ix	ih	picture
0	1	5	+ 5					
10	1	5	+ 5					

When your code compiles, linked and runs without error, and you have checked that the image position and heights, ask the TA to stamp.	TA Sign
--	----------------

Name		ID		Class ID	
Date		Time			

7 (Ray Tracing) Split File & Add Repetition For Multiple Lenses

7.1 Split the code into three separate files

Once we are sure that functions are working correctly, it is a good idea to place working functions in a separate file. After observing how the teacher splits the code into three files:

1. one file to hold the main code (mo7split.c)
2. one file to hold the functions called by main (molib.c)
3. one file to hold the function prototypes (molib.h), This file is called a header file.

Observe how the teacher splits the code into separate files. Similarly split your code into three separate files. Compile, link and run.	TA Sign
--	---------

7.2 Modify the code to handle a multiple lens systems

Modifications should be made in the main function (i.e. mo7split.c) Use either the **while()** or **do...while()** repetition structures to allow multiple lens to be used. Your program should print the final image position and height. Fix the original object's position at $x=0$ and its height as $h=1$. Name your final file **mo8loop.c**

7.3 Use your code to find the final image position and height. (units = mm)

Place the object at the location $x=0$ mm with a height $h=1$ mm. Assuming you have a 2 lens system, Perform the calculation. Draw the picture to scale.

Item	x [mm]	f/h [mm]	Picture (Scale 1mm = 1mm)
Object	$x=0$	$h=1$	
Lens	$x=10$	$f=5$	
Lens	$x=35$	$f=10$	
Image			

When your code compiles, linked and runs without error, and you have checked that the image position and height is correct, ask TA to stamp.	TA Sign
--	---------

Name		ID		Class ID	
Date		Time			

8 (Ray Tracing) Arrays: Separating Input and Output Phases

8.1 Modify existing UML diagram to separate input, calculation and output.

8.2 Modify the program mo8loop.c. Call the new program mo9arr.c.

1. Declare four arrays (lens position, lens focal length, image position, image height)
2. Initialize the original object position at x=0 and height as h=1.
3. Use a **do... while...** loop to enter the lens data into 2 arrays.
4. Use a **for** loop to calculate the position of the next image
5. Use a **for** loop to print out...
 - a. initial object location and height
 - b. location and focal length of each lens
 - c. image location and height after each lens

You should only modify the function **main()**. The format of your final output should be similar to the table in the following section. Do not modify any other function.

8.3 Use your program to obtain the final image position/height after 4 lens

Lens	ix	ih	lx	lf	Picture (Scale 1mm = 1mm)
1	0	1	5	2	
2			7	-3	
3			8	4	
4			10	2	
final					

Use your program to calculate the image position and height after each lens. Last week we saw that if you place your data in a file, you can have the C-program read the data from the file using input redirect (you may need to comment out the prompt in getFloat()). For example:

```
mo8arr < molens.csv
```

We can also write the data to a file and not to the screen. This is done using the output redirect command. To do both at once we can write:

```
mo8arr < molens.csv > mopath.csv
```

8.4 Open the output file "table.csv" in Microsoft Excel or OpenOffice Calc

When your code runs without error, and you have checked and you have opened the table in a spreadsheet file, ask TA to stamp.	TA Sign
---	---------

Name		ID		Class ID	
Date		Time			

9 (Ray Tracing) Records: Linking the data

9.1 Replace the four arrays with two arrays of records.

9.1.1 Define a new type, *Optdata*, that links each quantity with its *x* position

9.1.2 Declare two arrays of records of type *data*

1. Use one array of records to hold the lens data
2. Use one array of records to hold the image positions.

9.1.3 Modify the rest of function *main()* accordingly.

When your code compiles, linked and runs without error and you have verified your calculations, ask the TA to stamp. If this is being done as a homework assignment, please copy and print your code on the back of this sheet of paper.	TA Sign
--	----------------

9.2 Create a new function *getOptR()* to replace *getOpt()*.

Your new function should not use pointers but use “return” to get data back to your main function, I.e., Replace..

```
void getOpt(double *x, double *t);
```

with

```
struct Optdata getOptR(void);
```

9.3 Check to make sure that your program is running correctly

Use the same input file as you used last week in testing your array program. In other words get the data from the same file but save it to another file, e.g.

```
optics < molens.csv > optiRec.csv
```

Compare the output file this week with the file generated by the array program. They should be the same.

When your code compiles, linked and runs without error and you have verified your calculations, ask the TA to stamp. If this is being done as a homework assignment, please copy and print your code on the back of this sheet of paper.	TA Sign
--	----------------