Assignment 1

Assignment: Story Book
File Name: storybook.c

Stories are something that everyone enjoys. The type of stories you enjoy make you unique. I want you to think of your favorite stories (moves, tv series, books, fairly tales, campfire stores, etc). I would like you to print out a formatted list of your three top favorite stories, what type of story is it, and the year it was published. (If it wasn't published put N.D.

Example:

```
Story                   Type        Date
Star Wars               Movie       1977
Wheel of Time           Book        1990
Battlestar Galactica    TV Show     2004
```

In this assignment you will need to align the table as shown. You will not be using spaces to align these and you must use the tab escape character to receive full credit.

Assignment 2 (Part A)

Problem A - Reflection Pond Measurements
Filename: pond.c

Welcome to UCF! In front of the library we have a beautiful reflection pond. It has a foundation and during Spirit Week students get to jump in and have a pep rally. It is a lot of fun!

You have been tasked to find out the area that the pond covers. You will be given the radius of the pond. In this example you can assume that the pond is a perfect half circle.

Input Specification

Note: It is guaranteed that whoever uses your program will adhere to these specifications. This means that you do NOT have to check for them!

The radius entered will be integers in between 1 and 100, inclusive.

Output Specification

Produce two lines of output with the following format:

area = A

where A is the desired area. Please use 3.14 as pi to match the sample output.

Output Sample

Below is one sample output of running the program. Note that this sample is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given above. In the sample run below, for clarity and ease of reading, the user input is given in italics while the program output is in bold. (Note: When you actually run your program no bold or italics should appear at all. These are simply used in this description for clarity's sake.)

Sample Run #1

```
Enter the radius of the pond.
9
area = 127.17
```

Assignment 2 (Part B)

Filename: field.c

You are now in charge of intramural sports and your job to draw the lines on the field of each sport. You want to know how much paint you will need for the outer border of the field. In addition, you want to know what is the area of the field you are to paint. Write a program that asks the user to enter the length and width of a field, then uses this information to print out both the area and perimeter of the field.

Input Specification

Note: It is guaranteed that whoever uses your program will adhere to these specifications. This means that you do NOT have to check for them!

The length and width entered will be **integers** in between 1 and 100, inclusive.

Output Specification

Produce two lines of output with the following format:

area = A
perimeter = P

where A is the desired area and P is the desired perimeter.

Output Sample

Below is one sample output of running the program. Note that this sample is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given above. In the sample run below, for clarity and ease of reading, the user input is given in italics while the program output is in bold. (Note: When you actually run your program no bold or italics should appear at all. These are simply used in this description for clarity's sake.)

Sample Run #1

```
Enter the length of the field:
5
Enter the width of the field:
9
area = 45
perimeter = 28
```

Assignment 3 (Part A)

Problem A - Movie Going
Filename: movies.c

Last week I was able to go to see the new Dr. Strange movie. I thought it was pretty awesome. When I was looking at buying tickets I wanted to make sure I got as many discounts as I could get. By default a movie ticket is $15. If you go before 5pm, the ticket price is $12.50. If you plan on buying in bulk (5 or more tickets) you will receive a 10% discount. If you are a student, you get a 15% discount.

Write a program that asks the user if they are going to the movie before 5PM (1 for yes, 0 for no), how many tickets they are going to buy (an integer greater than 1), and if they are a student (1 for yes, 0 for no). Print out the total cost for the amount of tickets you would like to buy.

Sample Case 1:

Are you going to the movie before 5PM? (1 - Yes, 0 - No)
0
How many people are going?
2
Are you a student? (1 - Yes, 0 - No)
0
The group's price is $30.00.

Sample Case 2:

Are you going to the movie before 5PM? (1 - Yes, 0 - No)
1
How many people are going?
1
Are you a student? (1 - Yes, 0 - No)
1
The group's price is $10.62.

** *If you are getting 10.63 this is also correct. The way you round does not matter.*

Sample Case 3:

Are you going to the movie before 5PM? (1 - Yes, 0 - No)
0
How many people are going?
5
Are you a student? (1 - Yes, 0 - No)
1
The group's price is $57.37.

Assignment 3 (Part B)

Problem B - The Better Team
Filename: team.c

One of the craziest things about college sports is trying to rank them. There are so many teams, and many never play each other. For example, in division 1 basketball (which UCF plays in) has 350 different schools in it! There are human and computer rankings that get published each week. You assume that your school (UCF) will win the game if they have the lowest overall ranking. In general, UCF basketball teams play 2 games a week.

Your goal is to determine based on your ranking and the other two school's ranking how many games you will win and who is the best team.

Sample Case 1:

What is UCF's ranking?
50
What is team #1's ranking?
75
What is team #2's ranking?
25
UCF's Record: 1-1
Best Team: Team #2

Sample Case 2:

What is UCF's ranking?
1
What is team #1's ranking?
75
What is team #2's ranking?
25
UCF's Record: 2-0
Best Team: UCF

Sample Case 3:

What is UCF's ranking?
100
What is team #1's ranking?
15
What is team #2's ranking?
25
UCF's Record: 0-2
Best Team: Team #1

Assignment 4 (Part 1)

Loops 1

Filename: loops.c

In this program you will ask the user if they would like to create one of two patterns. The first pattern will be a triangle, and another will have alternating stripes.

For the triangle, you will ask the user how many rows. The first row will start with one star (*) and each row afterwards will increase by one additional star. For example if the user inputs 4 the triangle will appear like:

```
*
**
***
****
```

For the stripes, you will ask how many rows and how many items are in the first row. The second row will have one less. The third row will return back to the same amount that the 1st row, and the 4th will be the same as the 2nd row. This pattern will continue for how every many rows the user asked for. For example if the user says there will be 5 rows and 3 stars in in the first row, the output will look like:

```
***
**
***
**
***
```

Example Run:

```
What shape would you like to make? (1- Triangle, 2 - Stripes)
1
How many rows would you like?
6
*
**
***
****
*****
******
```

Example 2:

```
What shape would you like to make? (1- Triangle, 2 - Stripes)
2
How many rows would. you like?
6
```

```
How many stars in the first row?
4
****
***
****
***
****
***
```

Assignment 4 (Part 2)

Loops 2

Filename: bands.c

Each night when I leave campus, there is a band on campus playing something. They are pretty good, and it is one of the things I enjoy when walking back to my car.

Let's say you are the manager to see when the bands are playing. Your job is to create a list of people who are playing and when. To keep this simple, you are tracking time by the number of minutes that past since class got out. Other words, the time class got out is minute number 0.

You will need to ask the user the following question:

- When then first band will come out? (As an integer)
- How many bands will be playing tonight? (As an integer)
- For each band, ask the user how long that band will be playing.

Print out for each band when they go onto the stage. At the end print out the end time of the set.

Example Running

When does the first band come out?
30
How many bands will be playing tonight?
3
How long does band number 1 play?
15
Band #1 came out at 30 minutes.
How long does band number 2 play?
45
Band #2 came out at 45 minutes.
How long does band number 3 play?
5
Band #3 came out at 90 minutes.
The total set ended at 95 minutes.

Assignment 5 (Part 1)

Functions 1

Filename: functions(part1and2).c

You will write two functions. Each function is in two parts. Use the main function from the first part and replace it when working on the second.

Part 1

Write a function that calculates an employee's regular pay and returns that value. If an employee works less than the number of hours in a regular work week, then her pay is just the number of hours worked times her pay rate. If she works a number of hours equal to or greater than the regular work week, her pay is the regular work week multiplied by her regular pay rate. (Note: She will get paid more than this overall, but that extra pay is called overtime pay, for the purposes of this problem.)

The input to the function is as follows:

payPerHour - dollars per hour (float)
regWorkWeek - the number of hours in a regular work week
hoursWorked - the number of hours worked in a single week

Here is the function prototype with pre-conditions and post-conditions specified:

```
#include <stdio.h>

// Pre-conditions: payPerHour is a positive floating pt
// number representing the pay per hour. regWorkWeek
// represents the number of hours in a regular work week
// (int). hoursWorked is the number of hours the employee
// worked (int).
// Post-condition: Returns the regular pay for the employee.
double regularPay(double payPerHour, int regWorkWeek, int hoursWorked){
    // Put your code here.
}

// Here is a testing function to test your code and what it should print out:
void testRegularPay() {
    printf("%.2lf\n", regularPay(8.25, 40, 30));
    printf("%.2lf\n", regularPay(12.50, 30, 40));
    printf("%.2lf\n", regularPay(9.99, 80, 80));
}

int main() {
    testRegularPay();
    return 0;
}
```

Here is the expected output:

```
247.50
375.00
799.20
```

Part 2

If an hourly employee works longer than the regular work week, then she gets overtime pay. For the purposes of this problem, there is an overtime rate which is greater than 1. For
each hour past the regular work week, instead of getting paid the regular hourly rate, an employee gets paid the regular hourly rate times the overtime rate, per hour. For example,
if the regular work week was 50 hours/week, the employee worked 60 hours for one week,
the employee had a regular hourly rate of $10/hour and the overtime rate was 2.5, then the
employee would get (60 hrs - 50 hrs) x $10/hr x 2.5 = $250 total in overtime pay. If an employee works less than the regular work week, she gets $0 in overtime pay.
Write a function that takes in these four quantities and calculates overtime pay:

payPerHour - dollars per hour (float)
regWorkWeek - the number of hours in a regular work week
hoursWorked - the number of hours worked in a single week
overtimeRate - overtime hour multiplier factor

Here is the function prototype with pre-conditions and post-conditions specified:

```
#include <stdio.h>

// Pre-conditions: payPerHour is a positive floating pt
// number representing the pay per hour. regWorkWeek
// represents the number of hours in a regular work week
// (int). hoursWorked is the number of hours the employee
// worked (int). overtimeRate is the multiplicative rate
// for overtime pay.
// Post-condition: Returns the overtime pay for the employee.
double overtimePay(double payPerHour, int regWorkWeek, int hoursWorked, double overti
meRate){
    // Put your code here.
}

// Here is a testing function to test your code and what it should print out:
void testOvertimePay() {
    printf("%.2lf\n", overtimePay(8.25, 40, 30, 1.5));
    printf("%.2lf\n", overtimePay(12.50, 30, 40, 2.0));
    printf("%.2lf\n", overtimePay(9.99, 80, 100, 2.5));
}

int main() {
    testOvertimePay();
```

```
    return 0;
}
```

Here is the expected output:

```
0.00
250.00
499.50
```

Assignment 5 (Part 2)

Functions 2

Filename: functions2.c

For each week, the total amount of regular pay plus overtime pay represents an employee's gross earnings. (In real life, unfortunately, money is taken from your gross pay and what ends up in your paycheck is called "net pay.") Write a program that calls the functions from parts 1 and 2 and calculates the gross earnings for an employee over several weeks. Your program should ask the user to enter each of the following pieces of information once, at the beginning:

1. Hourly Pay Rate in dollars
2. Regular Work Week
3. Overtime Rate
4. Number of Weeks they are working

Then, for each week, your program should ask the user how many hours they worked. At the end of each week, print out the employee's regular pay, overtime pay, and total payment.

After reading in information for each of the weeks, print out the employee's total payment over the whole program.

Your program should be written in a function called main, and main should call the functions from parts 1 and 2.

Input Specification

Note: It is guaranteed that whoever uses your program will adhere to these specifications. This means that you do NOT have to check for them!

- Hourly Pay Rate will be a float in between 8.0 and 100.0.
- The regular work week will be an integer in between 30 and 80.
- The overtime rate will be a float in between 1.5 and 10.
- The number of weeks will be in between 2 and 10.
- For each week, the number of hours worked will be an integer in between 0 and 120.

Output Specification

Prompt the user to enter each of the four items designated above in the order specified in the input specification. (Come up with your own reasonable prompts.)
For each week, after asking the user to enter the number of hours they worked for that week, print out a statement with the following form:

Week k regular pay = $X, overtime pay = $Y, total pay = $Z.

where k is the week number starting with 1, X is the regular pay for week k, Y is the over time pay for week k and Z is the total payment. Each of X, Y and Z should be decimal values.

After the output for the last week, conclude with a single line with the following format:

Total pay for W weeks = $D.

Where W is the total number of weeks worked and D is the total pay.

Sample Run

```
How much (in dollars) are you paid per hour?
8.85
How many hours in the regular work week?
40
What is the overtime rate?
1.5
How many weeks are you working?
2
How many hours did you work in week 1?
20
Week 1 regular pay = $177.0, overtime pay = $0.0, total pay = $177.0.
How many hours did you work in week 2?
42
Week 2 regular pay = $354.0, overtime pay = $26.55, total pay = $380.55.
Total pay for 2 weeks = $557.55.
```

Assignment 6

Arrays

Filename: arrays.c

In this assignment you will be writing four functions that will modify an array. You will be given an array and the size in each of the functions and you write the features of that function. Your output should match the one listed below. I will be giving you a template. DO NOT modify the main function. All your code should be within the four functions.

```
// Print out the array in a single line
// For example if the array has the values 1, 2 and 3 it will print out:
// Array: 1 2 3
void print_array(int arr[], int size) {

}

// Add all the values together to get the sum of the entire array.
// Example: if the values are 1, 2 and 3 it would return 6.
int add_together(int arr[], int size) {

}

// find the lowest value in the array and return that value.
// Example: if the values are 3, 1, 2, the function returns 1
// All values will be under 1,000,000
int find_lowest(int arr[], int size) {

}

// Find the average value within the array with the lowest dropped
// You will have an array with size 2 or greater.
// Example if the values are 1, 2 and 3, the average would be 2.5

// You must call find_lowest, and add_together to receive full credit.
double average_drop1(int arr[], int size) {

}
```

Example Run:

```
Example 1
Array: 1 2 3 4 5
15
1
3.5000

Example 2
Array: 5 5 5 5 5 5 5
35
```

```
5
5.0000

Example 3
Array: 30 20 10 10
70
10
20.0000

Example 4
Array: 100 33 60 40
233
33
66.6667
```

Assignment 8

Pointers

Filename: pointers1.c

In this assignment I would like you write a few common pointer algorithms. You will need to write your own main function that tests out these functions.

Function 1:

- Returns a void
- Named F1
- Takes in two pointers to integers
- F1 swaps the value of the two integers

Example:

Start (main):

a = 10
b = 20

End (main):

a = 20
b = 10

Function 2:

- Returns an integer
- Named F2
- Takes in an integer array, and integer representing the size of the array.
- Iterate through the array and if any value is less than 10, set it to 10, if any values are greater than 20, set it to 20.
- Return how many changes you had to do.

Example:

Start
arr = [3, 5, 10, 15, 35, 17, 2]
size = 7

End:
arr [10, 10, 10, 15, 20, 17, 10]
size = 7
Function return 4

Assignment 9

2D Arrays

Filename: A9.c

During my time as an undergraduate student at UCF, I enrolled in 4-5 courses a semester. When I would enroll using myUCF I would enroll in one class at a time. One time, I tried to enroll into six courses and it told me I was unable to do so. This wasn't a technical limit, but a limit the university had placed on me. If I dropped a class, I would be able to enroll into the new class.

The Program

You are going to write a program that will ask the user which class they would like to take. They will input that class's name or course code as a string. Your program should be able to handle both names of the class (Introduction_to_C) or the course code (COP3233). The names of the courses will be less than 30 characters. After you insert the course in the list/array, you need to print out the course schedule. If you already have 5 courses in your list, you will print out the course schedule with the new class as the 6th course and ask the user which one they want to drop. They can input 6 to drop the one they just put in. You will need to not let the user put in any other number other than the 1-6, and will continue to prompt them until they put one in. The program ends when the user types in "EXIT" for the course name.

Example

Welcome to class registration!
Enter a course code:
COP3223
You have registered in the following courses:
1. COP3223
Enter a course code:
COP2500
You have registered in the following courses:
1. COP3223
2. COP2500
Enter a course code:
COT3100
You have registered in the following courses:
1. COP3223
2. COP2500
3. COT3100
Enter a course code:
MAC2312
You have registered in the following courses:
1. COP3223

2. COP2500
3. COT3100
4. MAC2312
Enter a course code:
PHY2048
You have registered in the following courses:
1. COP3223
2. COP2500
3. COT3100
4. MAC2312
5. PHY2048
Enter a course code:
CHM2048
You have registered in the following courses:
1. COP3223
2. COP2500
3. COT3100
4. MAC2312
5. PHY2048
6. CHM2048
You have registered for two many courses. Please pick one to delete.
Enter a number between 1 and 6.
2
Enter a course code:
ABC1234
You have registered in the following courses:
1. COP3223
2. CHM2048
3. COT3100
4. MAC2312
5. PHY2048
6. ABC1234
You have registered for two many courses. Please pick one to delete.
Enter a number between 1 and 6.
6
Enter a course code:
ABC2345
You have registered in the following courses:
1. COP3223
2. CHM2048
3. COT3100
4. MAC2312
5. PHY2048
6. ABC2345
You have registered for two many courses. Please pick one to delete.

Enter a number between 1 and 6.
6
Enter a course code:
EXIT
Goodbye

Assignment 10

Structs

Filename: 10A.c

We are going to revisit our course assignment. However, there is going to be some changes.

- We are going to create a menu of what they would like to do.
    - Add Courses
    - Delete Courses
    - Exit
- You will use a structure to store the course name and course number.
- You will need to use a linked list to store the values in order by course number.
- You will check to see if the course is already in the linked list (determine by checking course number), and if it is do not add it.
- You will determine if the course is in the list before you delete it. If it isn't display a message saying it isn't in the list.
- Print out the list. If there is nothing in your list print out "You aren't currently taking any courses."
- Course numbers will be less than 10 characters and course names will be less than 30 characters.

Here's what an example run can look like:

```
What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
3
You aren't currently taking any courses.

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
1
What course name would you like to add?
Intro_to_C
What course number would you like to add?
COP3223C
Added!

What courses would you like to do?
1. Add Course
2. Drop Course
```

```
3. Print Schedule
4. Exit
1
What course name would you like to add?
Computer_Science_1
What course number would you like to add?
COP3502C
Added!

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
3
Course Schedule:
1. COP3223C - Intro_to_C
2. COP3502C - Computer_Science_1

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
1
What course name would you like to add?
Concepts_in_Computer_Science
What course number would you like to add?
COP2500C
Added!

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
3
Course Schedule:
1. COP2500C - Concepts_in_Computer_Science
2. COP3223C - Intro_to_C
3. COP3502C - Computer_Science_1

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
2
What course code would you like to drop?
COP2100C
This course is not in your schedule.

What courses would you like to do?
1. Add Course
2. Drop Course
```

```
3. Print Schedule
4. Exit
2
What course code would you like to drop?
COP3223C
Course has been removed.

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
3
Course Schedule:
1. COP2500C - Concepts_in_Computer_Science
2. COP3502C - Computer_Science_1

What courses would you like to do?
1. Add Course
2. Drop Course
3. Print Schedule
4. Exit
4

Done!
```