Task 1:

Preprocessors include iostream and string and using std::options.  
Need enum position attached to the first character of each option, goalkeeper, midfielder, striker, winger, defender.  
Need struct called player with, int player number, float top speed, and enum type position pos.  
Need function prototype for getting a player instance with input of the player # in the loop to output 0, 1, 2 … etc. Returns a player.  
Need functions prototype for Displaying Player Stats which takes a player instance as input. No return.

Main.  
Create 2 separate player instances in their own variables (not an array etc) using Get Player function.  
Output the player stats using the Display Player Stats functions parsing in the player variable each time.

Get Player.  
Takes player number display (int) as input and displays this as “Player” + # before getting input.  
Create a blank player instance and go through 3 loops to get valid input for each type of input.  
Force correct input in each loop.  
return complete player instance with values now assigned.

Display Player Stats.  
Output each stat and also use switch case to display the player position type.

Task 2:

Preprocessors include iostream and string and using std::options.  
Need function prototype for Getting valid input, takes a string prompt and bool for newline or not, returns float value.  
Need function prototype for Getting Sum of Positive Only Numbers which takes a list pointer and list size to loop through the list. Returns float sum.

Main.  
Create valid array using user input for size for list of floats.   
Create array of user input size of type float in heap attached to pointer called list, getting user input from Get Valid Input Functions, and explicitly casting to an int.  
Loop through list and Get Floats from Get Valid Input Function and assign to index correctly  
Get Sum Of All Positive Numbers in list using respective function and params.  
Output sum.

Get Valid Input.  
Returns float value by getting string input (get\_line) removing ws to stop errors. Try cast to float using stof and catch conversion error, loop if wrong input. Return correct input.

Get Sum Positive.  
Create total starting at zero. Loop through list and add only the positive numbers to the total. Return the total.

Task 3:

Preprocessors include iostream and string and using std::options.  
Need function prototype for Getting matrix input takes parameters (double(&array)[3][3]), const int height and const int width. This is void return type.  
Need function prototype for adding two matrices together which takes parameters (double(&arrayOne)[3][3]), (double(&arrayTwo)[3][3]), const int height and width. Returns a new 2d array by a double\*\*.  
Need function prototype for multiplying a matrix by a constant which takes parameters double\*\* matrix, int multiple, const int height and width. Returns new array of type double \*\*  
Need function prototype for displaying a matrix which takes parameters double\*\* matrix, const int height and width. Returns void.

Main():  
create 2 double name[3][3] in stack  
Get input for each of them through GetMatrix functions.  
Create double\*\* c from Add Function which the two matrices as inputs.  
Multiply C matrix by 3 with Functions Multiply().  
Display the output matrix using function>  
Delete C because it was created new in function.

Get input for matrix is done by looping and getting double input for each one. This is done by using the address of array which is one of the params which makes input easy.

Add takes the two original arrays and creates a new double\*\* into heap. Loops through each array and adds values to new matrix value. Multiply does the same thing but with a constant and taking the double \*\* as param. Display loops through array as required.

Task 4

Task 5

Task 5

Task 6

\*\*Note create an expense for a day not an expense!! Each type of expense is in the expense struct and an expense is for a whole day or week. Store expenses in separate lists as theirs no point in having them together.