Regarding atomic functions and good programming practice

Pls review the following code which is logically and syntactically fine but functions are not atomic. Lets explore the functions one by one and correct them..

```cpp
#include<iostream>
using namespace std;
int hour, minute, second;
int input(int, int, int);
int addition(int, int, int);
int charges(int total_hours, int total_minutes, int total_seconds);
int main()
{
	input(0, 0, 0);
}
int input(int, int, int)
{
	int total_hours = 0, total_minutes = 0, total_seconds = 0;

	cout << "Enter the time the internet was used for in each sitting(in hours, 
minutes and seconds).\nEnter -1 to end.\nEnter time of First Sitting : ";
	cout << "\nHours: ";
	cin >> hour;
	do
	{
		total_hours = total_hours + hour;
		cout << "Minute: ";
		cin >> minute;
		total_minutes = total_minutes + minute;
		cout << "Seconds: ";
		cin >> second;
		total_seconds = total_seconds + second;
		cout << "Enter time of next Sitting : ";
		cout << "\nHours: ";
		cin >> hour;
	} while (hour != -1);
	addition(total_hours, total_minutes, total_seconds);
	return 0;
}
int addition(int total_hours, int total_minutes, int total_seconds)
{
	total_minutes = total_minutes + (total_seconds / 60);
	total_hours = total_hours + (total_minutes / 60);
	charges(total_hours, total_minutes, total_seconds);
	return 0;
}
int charges(int total_hours, int total_minutes, int total_seconds)
{
	cout << "Total time of usage is " << total_hours << endl;
	cout << "Total charges are Rs. " << total_hours * 25;
	return 0;
}
```

Input function:

This function should logically help in taking input from the user. The input should obviously must be stored in variables and this function should not return anything because this is not required!

Input/Output: technically this function does not need any thing from us (the user) to take input. And it also does not need to return anything. E.g. in case of FindSum function we need a **sum** from this function. So, ideally the function should look like:

```
void input()
{
	//declare relevant variables
	//and take input in then through cin
}
```

But, we also know that we need the values that you take from the user (hours, minutes, seconds) in other functions (add to total and calculate charges). Thus, we need to do some adjustments and ordering in our code lines to make these values available to the other functions. There are three possibilities for it:

1) **Declare hours, minutes, seconds as global variables**
   But, we already studied that we should not use global variables extensively in our code. We need it in scenarios when we have to deal with constants that are technically common to whole application. .e.g interest rate in bank application. We cannot declare our regular used variables (used in our logic design) as global variables.
2) **Call the other function that need input values in the input function itself like:**
   ```
   void input()
   {
   	//declare relevant variables
   	//and take input in then through cin
   	//call the function(s)
   	addition(total_hours, total_minutes, total_seconds);

   }
   ```
   This is also not a good programming practice. The reason is that input function must only consists of those lines of code/ logic that is related to input only. It should not contain any other functionality, like in this case we are adding function call to achieve addition.
3) **Use reference variables**

   The third and acceptable option is to use reference variables. Pass reference of variables to function so that the input function takes input in original variables (that are referred).

```
int main()
{
	input(0, 0, 0);
}
int input(int, int, int)
{
	int total_hours = 0, total_minutes = 0, total_seconds = 0;

	cout << "Enter the time the internet was used for in each sitting(in hours, minutes and seconds).\nEnter -1 to end.\nEnter time of First Sitting : ";
	cout << "\nHours: ";
	cin >> hour;
	do
	{
		total_hours = total_hours + hour;
		cout << "Minute: ";
		cin >> minute;
		total_minutes = total_minutes + minute;
		cout << "Seconds: ";
		cin >> second;
		total_seconds = total_seconds + second;
		cout << "Enter time of next Sitting : ";
		cout << "\nHours: ";
		cin >> hour;
	} while (hour != -1);
	addition(total_hours, total_minutes, total_seconds);
	return 0;
}
```

Thus better implementation of this function would be:

```
void input(int &hour,int &minute, int &second)
{
              cout << "Enter the time the internet was used for in each
sitting(in hours, minutes and seconds).\nEnter -1 to end.\nEnter time of First
Sitting : ";
              cout << "\nHours: ";
              cin >> hour;
              cout << "Minute: ";
              cin >> minute;
              cout << "Seconds: ";
              cin >> second;
}
```

addtoTotal/addition function:

This function should logically help in adding the inputs taken from the user to some variable. That is, hours, minutes, and seconds for each sitting must add up to total_hours, total_minutes, and total_seconds, respectively. Lets explore the following function.

```
int addition(int total_hours, int total_minutes, int total_seconds)
{
       total_minutes = total_minutes + (total_seconds / 60);
       total_hours = total_hours + (total_minutes / 60);
       charges(total_hours, total_minutes, total_seconds);
       return 0;
}
```

Input/Output: technically this function does not need anything from us (the user) to add up the inputs gradually. However, it do need from us the values of input hours/minutes/seconds because that needs to be added up for each sitting. With the same motivation of reference variables used for input function, we can use the same concept here and remove the call to **charges** function that is irrelevant for the scope of this function. The addition function should only contain functionality related to addition. Nothing more and nothing less!. Thus, a better version would be:

```
void addition(int &total_hours, int &total_minutes, int &total_seconds, int
hour, int second, int minute)
{
       total_hours = total_hours + hour;
       total_seconds = total_seconds + second;
       total_minutes = total_minutes + minute;
}
```

Calculate charges function:

This function should logically help in calculating the total charges. While calculating the charges the hours related calculation also needs to be performed (i.e. to convert minutes and seconds to hours). Any kind of conversion related logic should be the part of this function. You can think about inserting the **convert hours,minutes,seconds to HOURS** logic in addition function. But the purpose of addition function was to take hours one by one and add then to TOTAL HOURS. Same goes for minutes and seconds. Converting minutes to hours or seconds to minutes does not as such comes under its scope. So it is better to put that logic in charges function.

```
int charges(int total_hours, int total_minutes, int total_seconds)
{
	cout << "Total time of usage is " << total_hours << endl;
	cout << "Total charges are Rs. " << total_hours * 25;
	return 0;
}
```

A better implementation would be:

```
int charges(int total_hours, int total_minutes, int total_seconds)
{
	total_minutes = total_minutes + (total_seconds / 60);
	total_hours = total_hours + (total_minutes / 60);
	cout << "Total time of usage is " << total_hours << endl;
	cout << "Total charges are Rs. " << total_hours * 25;
	return 0;
}
```

You can have another print function that prints the final answer. But it is needless to do so as we discussed that too many smaller functions are actually an overhead to our code's performance. Therefore skip making too simple and too short multiple functions.

Now lets have a look at the main function (older and corrected version). The red version has only one call that meaninglessly takes three 0s as an input (Not required logically!).

```
int main()
{
	input(0, 0, 0);
}
```

Think of your main function as an interface. That is, the starting point of your application. Assume you start any game, lets say StreetFighter.. First, always an interface screen appears that takes input from you how you want to use the game application. You give input by pressing any button (settings, help, vs battle, team battle, quit). The main function then calls the relevant methods. Thus, ideally your main should only contain calls to functions. Yes, it CAN contain few other statements too, like variable declarations, or some messages to print anything on console.e.g welcome message. But you should try to keep it free from any manipulation or logic. Thus, a better implementation of main function would be:

```
int main()
{
	int hour=0, minute=0, second=0;
	int total_hours = 0, total_minutes = 0, total_seconds = 0;
	do
	{
		input(hour,minute,second);
		if(hour!=-1)
			addition(total_hours, total_minutes, total_seconds,
		hour,minute,second);
	} while (hour != -1);
	charges(total_hours, total_minutes, total_seconds);
}
```

To Summarize pls see full code versions (old + improved version)

```
#include<iostream>
using namespace std;
int hour, minute, second;
int input(int, int, int);
int addition(int, int, int);
int charges(int total_hours, int total_minutes, int total_seconds);
int main()
{
	input(0, 0, 0);
}
int input(int, int, int)
{
	int total_hours = 0, total_minutes = 0, total_seconds = 0;

	cout << "Enter the time the internet was used for in each sitting(in hours,
minutes and seconds).\nEnter -1 to end.\nEnter time of First Sitting : ";
	cout << "\nHours: ";
	cin >> hour;
	do
	{
		total_hours = total_hours + hour;
		cout << "Minute: ";
		cin >> minute;
		total_minutes = total_minutes + minute;
		cout << "Seconds: ";
		cin >> second;
		total_seconds = total_seconds + second;
		cout << "Enter time of next Sitting : ";
		cout << "\nHours: ";
		cin >> hour;
	} while (hour != -1);
	addition(total_hours, total_minutes, total_seconds);
	return 0;
}
int addition(int total_hours, int total_minutes, int total_seconds)
{
	total_minutes = total_minutes + (total_seconds / 60);
	total_hours = total_hours + (total_minutes / 60);
	charges(total_hours, total_minutes, total_seconds);
	return 0;
}
int charges(int total_hours, int total_minutes, int total_seconds)
{
	cout << "Total time of usage is " << total_hours << endl;
	cout << "Total charges are Rs. " << total_hours * 25;
	return 0;
}
```

```
#include<iostream>
using namespace std;
void input(int &, int &,int &);
void addition(int&, int&, int&,int,int,int);
int charges(int total_hours, int total_minutes, int total_seconds);
int main()
{
       int hour=0, minute=0, second=0;
       int total_hours = 0, total_minutes = 0, total_seconds = 0;
       do
       {
              input(hour,minute,second);
              if(hour!=-1)
                     addition(total_hours, total_minutes, total_seconds,
hour,minute,second);
       } while (hour != -1);
       charges(total_hours, total_minutes, total_seconds);
}
void input(int &hour,int &minute, int &second)
{
              cout << "Enter the time the internet was used for in each sitting(in
hours, minutes and seconds).\nEnter -1 to end.\nEnter time of First Sitting : ";
              cout << "\nHours: ";
              cin >> hour;
              cout << "Minute: ";
              cin >> minute;
              cout << "Seconds: ";
              cin >> second;
}
void addition(int &total_hours, int &total_minutes, int &total_seconds, int hour,
int second, int minute)
{
       total_hours = total_hours + hour;
       total_seconds = total_seconds + second;
       total_minutes = total_minutes + minute;
}
int charges(int total_hours, int total_minutes, int total_seconds)
{
       total_minutes = total_minutes + (total_seconds / 60);
       total_hours = total_hours + (total_minutes / 60);
       cout << "Total time of usage is " << total_hours << endl;
       cout << "Total charges are Rs. " << total_hours * 25;
       return 0;
}
```