

# Start Coding

Some basic information here!



*By Adam Higherstein*

## Table of Contents

Real values can help to solve the problem.....	1
Example 1: fractions.....	1
Example 2: check email.....	2
Example 3: Roman numerals.....	2
Program design.....	3
Example 4: Pseudocode.....	4
Example 5: flowchart.....	5
Example 6: what does that code do?.....	5
Example 7: simulating the code.....	6
Example 8: Updating a table.....	10
Coding pitfalls.....	12
Example 9: Printing takes time.....	12
Example 10: extra semicolon.....	14
Example 11: array index out of bounds.....	14
Example 12: floor division.....	15
Example 13: too big value to variable.....	17
Example 14: Unicode '0' is 48.....	18
Example 15: wrong datatype.....	19
Example 16: if else.....	20
Example 17: missing null.....	22

# Code Design Basics

To be able to code solution for some problem, you have to be able to solve that problem first yourself, without any coding at all!

## Real values can help to solve the problem

### Example 1: fractions

For example you have to create an app that calculates the product of two fractions, you have first be able to calculate that product "manually", may be using pen and paper! Or at least you have to know how that calculations is normally, with arithmetic methods, to be done.

So sometimes it is good to use a concret case to start with. We can have a real example first.

In this case we can have to fractions, e.g.  $5/6$  and  $11/18$ .

Then we calculate the product (as we know a bit math):

Product is  $5 \cdot 11 / 6 \cdot 18$  and it is  $55/108$

So to our code we have to create 4 integer variables, eg.  
x1, x2, y1, y2

And for the result may be r1 and r2.

And then we assign values:

x1 = 5

y1 = 6

x2 = 11

y2 = 18

And results:

r1 = x1\*x2

r2 = y1 \* y2

And we get the result.

Of course it may happen that fractions are given as strings: so we have to be able to split the strings and then convert parts to integers and assign to variables.

SO, may be we use strings and user gives e.g.  
"5/6" and "11/18".

Now we have first find the character '/', called slash: may be the name of that position is slashplace.

"5/6"      "11/18"

      \        /

      slashplaces

Then we remove numerators that are between position 0 and slashplace.  
 And then we remove denominators that are between slashplace and the ends of the strings:  
 end position is normally the length of the string...

## Example 2: check email

We take another example.

Problem:

Find the country code of url, normally we use the name "top-level domain".

How to do it? Start again with "paper and pen"!

Write any real url and study it a bit

E.g.

["http://www.nba.com"](http://www.nba.com)

Url has standard format

Country code is after the last dot.

SO:

Find the place of the last dot.

["http://www.nba.com"](http://www.nba.com)  
 /  
 search the position  
 of last dot

Catch the part between the end of url and the place of the last dot: you get "com"

## Example 3: Roman numerals

Your code converts Western (Arabic) numerals to Roman

First, we have to understand what western and roman numerals are...

Symbol	I	V	X	L	C	D	M
Value	1	5	10	50	100	500	1000

Then it is easy to notice that there are several choices, thus decision making is needed.  
If e.g. user gives 5, we can print V.

From 1 to 10 we can easily get  
by just using if statements  
In Pseudocode:  
If val is 1 then roman is I  
If val is 2 then roman is II  
If val is 3 then roman is III  
If val is 4 then roman is IV  
And so on

But what about bigger western values?

How can do if values are bigger?

We can not code thousands of if sentences!

Instead we can try to take divide the value in smaller pieces,  
convert pieces to roman type and combine them to get final value

Let's take (again) a real value and use "paper&pen" to solve problem first manually:

Arabic value is now 2222

Get thousands

$2222/1000$  we get 2

2) Calculate remaining part

$2222 \% 1000$  is 222

3) Get hundreds

$222/100$  is 2

4) Remains  $222 \% 100 \Rightarrow 22$

5) Get 50's  $\rightarrow$  no

6) Get 20's  $\Rightarrow 22/20$  is 1, remains 2

7) No tens and 2 is II

Now, the code is easier to write!!

## Program design

About program flow

In programs execution can go  
straight forward

or it contains decision making (branching)

or there are loops

Three different ways to go.

We can create the plan of the program or algorithm by using pseudocode or flowchart

In pseudocode we use clear text show to program flow, often structures that are used in programs: if, if – else, while, do – while, print and so on... We also add there indentations... From pseudocode it can be easier to create the code.

## Example 4: Pseudocode

We have to solve a quadratic equation: it is a second order equation.

$$ax^2 + bx + c = 0$$

Of course, we have to know what a quadratic equation is and what do characters a, b and c mean here. And is there a formula to solve equation and how many roots there may be.

So we can remember that in terms of a, b and c, the formula is.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

And that the square root expression is called discriminant:

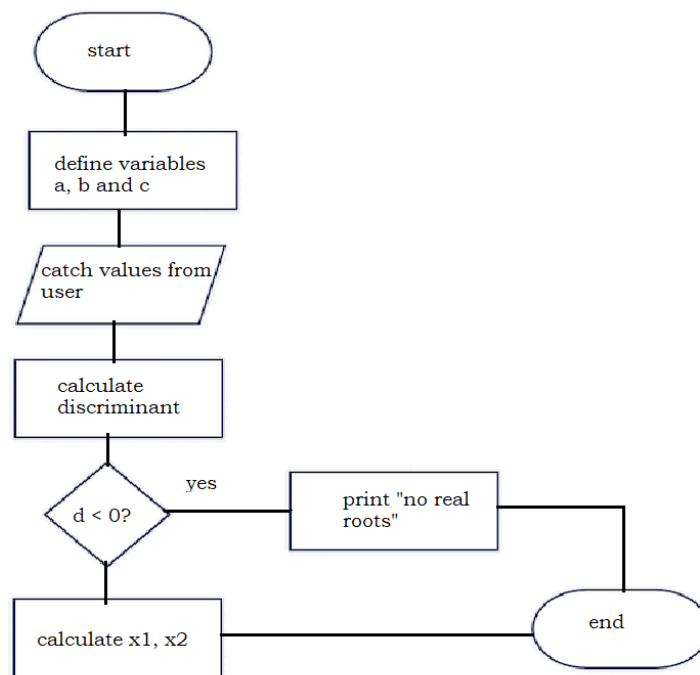
$$\sqrt{b^2 - 4ac}$$

So, we can first create the pseudocode.

```
Add values to a, b and c
calculate the discriminant (d)
if d < 0
    print no real roots
else
    calculate x1 and x2 using the formula
    print x1 and x2
```

and with flowchart next...

## Example 5: flowchart



So, maybe the coding is easier now!

## Example 6: what does that code do?

What about understanding some code or program flow?  
Let's try to find out helping methods.

Example 5

What does the code do?

e.g. we have now a simple swapping code here:

In C# the code could be

```
int a = 5;
```

```
int b = 99;
```

6

```
int temp = a;  
a = b;  
b = temp;
```

Let's study the code:

```
a = 5  
b = 99
```

swap values: we can not write `a = b`,  
because then we lose the original value of `a` (5)  
we need an extra variable, name could be `temp`, and then:

```
temp = a  now temp is 5, a is 5, b is 99  
a = b     now a is 99, b is 99, temp is 5  
b = temp  now b is 99, temp is 99, a is 5
```

## Example 7: simulating the code

We take a look at the selection sort method

Here is the C code:

```
int values[] = {20, 30, 5, 9, 2, 0, 22};  
  
int i, j;  
for (i = 0; i < 7; i++)  
    for (j = i + 1; j < 7; j++)  
    {  
        if (values[j] < values[i])  
        {  
            int temp = values[i];  
            values[i] = values[j];  
            values[j] = temp;  
        }  
    }
```

Let's try to understand the method by simulating that code.

Here we have values:

20	30	5	9	2	0
----	----	---	---	---	---

Round 1:

We want to find the smallest value and add it to the beginning of this array  
SO, the first value is now 20, place is `values[0]`



7

Now we compare 20 to the next value of the array (place is values[1]  
if the next value, 30, is smaller, we swap values

Round 1:

SO, let's go on:

$30 < 20$ ?

No, we do nothing

$5 < 20$ ?

Yes, we swap values and get:

5	30	20	9	2	0	22
---	----	----	---	---	---	----

Round 1:

We go on with value 5 now:

$9 < 5$ ?

No, we do nothing

$2 < 5$ ?

Yes, values are swapped and we get

2	30	20	9	5	0	22
---	----	----	---	---	---	----

Round 1:

We go on with value 2 now:

$0 < 2$ ?

Yes, we swap values and get

0	30	20	9	5	2	22
---	----	----	---	---	---	----

Round 1:

Finally we compare 0 to 22 and do nothing.

So the result of first round is here!

0	30	20	9	5	2	22
---	----	----	---	---	---	----

0	30	20	9	5	2	22
---	----	----	---	---	---	----

Round 2:

we compare the second value (values[1], 30 now) to other values:

$20 < 30$ ?

Yes, swapping and we get

0	20	30	9	5	2	22
---	----	----	---	---	---	----

$9 < 20$ ?

Yes, swapping

0	9	30	20	5	2	22
---	---	----	----	---	---	----

0	9	30	20	5	2	22
---	---	----	----	---	---	----

$5 < 9$ ?

Yes, swapping and we get:

0	5	30	20	9	2	22
---	---	----	----	---	---	----

$2 < 5$ ?

Yes, swapping and we get:

0	2	30	20	9	5	22
---	---	----	----	---	---	----

0	2	30	20	9	5	22
---	---	----	----	---	---	----

Next round:

$20 < 30$ ?

Yes, swapping and we get:

0	2	20	30	9	5	22
---	---	----	----	---	---	----

$9 < 20$ ?

Yes, swapping and we get:

0	2	9	30	20	5	22
---	---	---	----	----	---	----

0	2	9	30	20	5	22
---	---	---	----	----	---	----

$5 < 9$

Yes, swapping and we get:

0	2	5	30	20	9	22
---	---	---	----	----	---	----

No more changes ...

0	2	5	30	20	9	22
---	---	---	----	----	---	----

Next round:

$20 < 30$ ?

Yes, swapping and we get:

0	2	5	20	30	9	22
---	---	---	----	----	---	----

$9 < 20$ ?

Yes, swapping and we get:

0	2	5	9	30	20	22
---	---	---	---	----	----	----

No more changes...

0	2	5	9	30	20	22
---	---	---	---	----	----	----

Next round:  $20 < 30$ ?

Yes, swapping and we get:

0	2	5	9	20	30	22
---	---	---	---	----	----	----

No more changes...

0	2	5	9	20	30	22
---	---	---	---	----	----	----

Last round:

22 < 33?

Yes, swapping and we get:

0	2	5	9	20	22	30
---	---	---	---	----	----	----

That's is!

Array is sorted!

May be it is now easier to remember the method code or create it from scratch ...

## Example 8: Updating a table

Other ways to try to understand code: you can use a table that has variable names as table headers. Then you can simulate the code and enter values to variables when they change... This is quite good e..g when we have repeating parts, loops, in the code...

Example loop:

```
int sum = 0;
int n = 0;
for (int a = 2, a < 6; a++)
{
    sum = sum + 2;
    n++;
}
```

```
float average = sum/n;
print(sum);
print(average);
```

Round	a	a < 6??	sum	n
0	?	?	0	0
1	2	true	2	1
2	3	true	5	2
3	4	true	9	3
4	5	true	14	4
5	6	False, break the loop	14	4

Then we calculate  
average =  $14/4$   
sum was 14

Instead of table you can also have some boxes for variables and during simulation you can update the values of boxes...

## Coding pitfalls

### Example 9: Printing takes time

Avoid printing if it not absolutely needed.

Here is an examnle

```
// About unnecessary printing
// Printing is time comsuming!!

// Example: calculate sum of values 1 ... n
int n = 100;
int sum = 0;
Stopwatch stopWatch = new Stopwatch();
stopWatch.Start();
for (int k =1; k <= n; k++)
{
    sum = sum + n;
}
stopWatch.Stop();
TimeSpan ts = stopWatch.Elapsed;
Console.WriteLine("Elapsed time is " + ts);
Console.WriteLine("Sum is " + sum);
```

It did not take much time:

```
Elapsed time is 00:00:00.000042
Sum is 10000
```

Now add printing

Example: b)  
- printing the  
accumulated

```
// About unnecessary printing
// Printing is time consuming!!

// Example: calculate sum of values 1 ... n
int n = 100;
int sum = 0;
Stopwatch stopWatch = new Stopwatch();
stopWatch.Start();
for (int k = 1; k <= n; k++)
{
    sum = sum + n;
    Console.WriteLine("Sum is now " + sum);
}
stopWatch.Stop();
TimeSpan ts = stopWatch.Elapsed;
Console.WriteLine("Elapsed time is " + ts);
Console.WriteLine("Sum is " + sum);
```

Result is seen here:

Before:

```
Elapsed time is 00:00:00.0000042
Sum is 10000
```

After:

```
Sum is now 9600
Sum is now 9700
Sum is now 9800
Sum is now 9900
Sum is now 10000
Elapsed time is 00:00:00.0637042
Sum is 10000
```

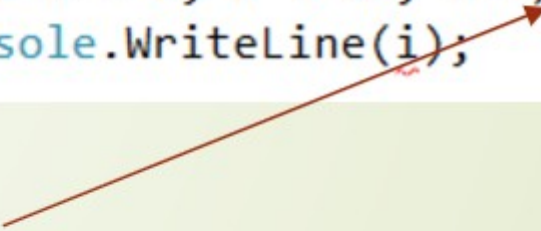
We wasted much time!!

## Example 10: extra semicolon

Common mistake

```
// print values 0 to 9
// local variable is not seen outside the block


for (int i = 0; i < 10; i++);
    Console.WriteLine(i);
```



Semicolon does  
not belong here!

## Example 11: array index out of bounds

```
int[] vals = {22, 44, 5, 6, 1111, 2, 9999};
int size = vals.Length;
for (int i = 0; i <= size; i++)
    Console.WriteLine(vals[i]);
```



Exception Unhandled

**System.IndexOutOfRangeException:** 'Index was outside the bounds of the array.'

Arrays: Index out of the range



The screenshot shows a code editor with the following C# code:

```
// 2 statements in the for body  
// missing program block (curly braces)  
int arraySum = 0;  
int p;  
for (p = 0; p < size; p++)  
    Console.WriteLine(vals[p]);  
    arraySum += vals[p];
```

Two red arrows point to the missing curly brace after the `Console.WriteLine` statement and the `arraySum += vals[p];` statement. A text label "Missing curly braces!" is positioned below the arrows.

On the right side, an "Exception Unhandled" dialog box is displayed, showing the error: `System.IndexOutOfRangeException` the array.' It includes links for "View Details" and "Copy Details", and a button for "Exception Settings".

## Example 12: floor division

Some languages have integer division: sometimes coder does not remember it.

Integer division:  
if not wanted...

```
int a = 10;  
int b = 3;  
Console.WriteLine("Result is " + a/b);
```

```
Result is 3
```

Solution

```
int a = 10;  
int b = 3;  
Console.WriteLine("Result is " + a/b);
```

```
Result is 3
```

```
int a = 10;  
int b = 3;  
Console.WriteLine("Result is " + (float)a/b);
```

```
Result is 3,333333
```



### Example 13: too big value to variable

Outside the range  
of integer type

$2^{16} =$   
**65 536**

Max value is 65535

```
unsigned short x1 = 70000;  
printf("%d \n", x1);
```

4464

```
unsigned short x1 = 70000 - 65536;
```

464

Same thing with signed  
types:

```
short x1 = 40000;  
printf("%d \n", x1);
```

-25536

## Example 14: Unicode '0' is 48

Commonly char type  
Is also an integer.  
But, is ASCII code 0 has  
value 48, 1 has value 49  
and so on!

```
printf("How many time do you eat meal daily: \n");  
int meal_times = getchar();  
  
int weekly_meal_times = 7 * meal_times;  
  
printf("OK, every week you eat %d meals! \n", weekly_meal_times);
```

```
How many time do you eat meal daily:  
3  
OK, every week you eat 357 meals!
```

Remember to subtract 48  
from given char value!

## Example 15: wrong datatype

C and C++ allow more  
than e.g Java or C#

```
int x2 = 3.65;
```

incompatible types: possible lossy conversion from double to int  
----  
(Alt-Enter shows hints)

```
int x2 = 3.65;
```

Same thing  
if we had used float  
here

```
int x2 = 3.65;
```

struct System.Double

Represents a double-precision floating-point number.

Cannot implicitly convert type 'double' to 'int'. An explicit conversion

## Example 16: if else

Many options: else is not placed correctly

```
// else in wrong place
int value = 3;
if (value == 1)
    Console.WriteLine("Uno");
if (value == 2)
    Console.WriteLine("Due");
if (value == 3)
    Console.WriteLine("Tre");
if (value == 4)
    Console.WriteLine("Quattro");
if (value == 5)
    Console.WriteLine("Cinque");
else
    Console.WriteLine("Esulano dall'ambito di applicazione");
```

Belongs to this if only

Now we get extra output

Extra output

Tre  
Esulano dall'ambito di applicazione

## Solution

```
int value = 3;  
if (value == 1)  
    Console.WriteLine("Uno");  
else if (value == 2)  
    Console.WriteLine("Due");  
else if (value == 3)  
    Console.WriteLine("Tre");  
else if (value == 4)  
    Console.WriteLine("Quattro");  
else if (value == 5)  
    Console.WriteLine("Cinque");  
else  
    Console.WriteLine("Esulano dall'ambit
```

Tre

## Example 17: missing null

C/C++: missing null from the string

```
char city[10];  
city[0] = 'R';  
city[1] = 'e';  
city[2] = 'n';  
city[3] = 'o';  
printf("%s \n",city);
```

We get

Reno√



That's all, folks!

Hope to get feedback!