1. Handout - TODO

Debugging:

F5: Start debugging
Shift F5: End debugging
F9: Add/Remove breakpoint at the current line
Ctrl + Shift + F9: Removes all breakpoints
F10: Step over
F11: Step into
Shift F11: Step over

Ctrl + K + C: Comment all
Ctrl + K + U: Uncomment all

Breakpoints types:

- **Simple**
- **Conditional**
- **Hit count**
- **Filter**

Autos:
Locals:
Watch and Add Watch:
Call stack:

Errors:

***Syntax error*:**
This error occurs due to following reason.
I. Not following the grammatical rules used in declaration of identifier.
II. Not declaring an identifier used in program.
III. Not terminating statement by semicolon.
IV. Not providing equal number of opening and closing braces etc.
These errors can be rectified by the user as it is displayed while compiling the program.

***Runtime error*:**
This error occurs while running a program by displaying the message listed below.

I. Division by 0.
II. Overflow
III. Underflow

***Logical error*:**
This error won't be displayed on the screen. However it will lead to display wrong results. Example: An infinite loop. This error lead to abnormal termination of a program or infinite loop.

2. Live debugging exercise:

- **breakpoints (normal, conditional, hitcount, action - log)**
- **Call stack**
- **Locals, Autos, Watch**
- **Editing in the watch**
- **Function**
- **For / while example**
- **Errors: move the sum function at the end of the file, (+ overflow in the for)**

```
int sum(int a, int b)
{
        return a + b;
}

int main()
{
        cout << sum(20, 10) << endl;
        int sumOfThree = sum(sum(1, 2), 100);
        cout << sumOfThree<< endl;
        cout << sum(sumOfThree, sumOfThree) << endl;

        int a[5] = { 1,3,5,7,9 };
        for (int i = 0; i < 5; ++i)
        {
                cout << sum(a[i], a[i+1]) << endl;
        }
        cout << "Hello" << endl;}
```

3. Hands on

a.) For debug:

The program takes two numbers from the user (a base number and an exponent) and calculates the power. (manually)
After this, calculate the power using the “pow” function. Debug it and step into this function.

Solutions:

```
while (exponent != 0)
{
    result *= base;
    --exponent;
}
```

```
// for(int i = 0; i < exponent; ++i) { result *= base; }
```

```
// result = pow(base, exponent);
```

b.) For debug:

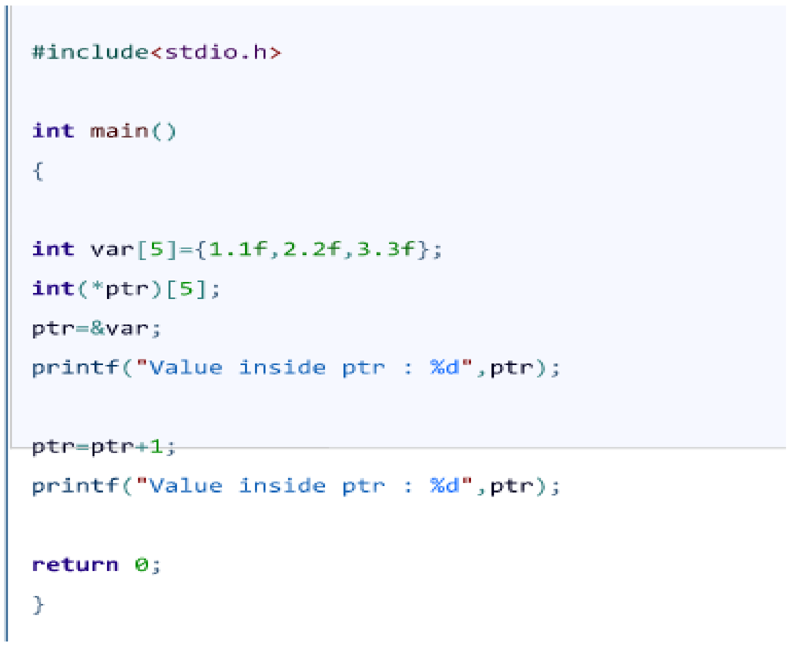
1.

The program takes 2 numbers from the user and swaps them (multiple solutions are available).
Please write a function for the swapping and call it from the main.
Solutions:

- Using a third variable
- Without using a third variable

Important: The parameters should be passed by reference /address!

2. Pointers



```
#include<stdio.h>

int main()
{

int var[5]={1.1f,2.2f,3.3f};
int(*ptr)[5];
ptr=&var;
printf("Value inside ptr : %d",ptr);

ptr=ptr+1;
printf("Value inside ptr : %d",ptr);

return 0;
}
```

c.) Error handling: Resolve the given code errors

1.
- What is the output if a = 5, b = 5?
- What is the output if a=5, b = 6?
- Resolve the logic error!

```
int main()
{
        int a, b;
        cin >> a;
        cin >> b;

        if (a = b)
        {
                cout << "a = " << a << " is equal to b= " << b << endl;
        }
}
```

Solution: The "=" vs "=="

4. Resolve the errors

```
int main()
{
        // This should print the sum of 2 numbers
        int a, b;
        int sum=a+b;
        cout<<"Enter two numbers to add: ";
        cin>>a;
        cin>>b;
        cout<<"The sum is: "<<sum;

        // This should print numbers from 0 to 9.
        int x;
        for (x = 0; x<10; x++);
                cout << x;
}
```

Solution:

- initialization of variables before usage or calculate sum after the input is done
- ";" after the for is not needed

d.) Debug quiz:

- Which is the value of the element at the 700 th index? → 290
- Which is the index of the element = 11? → 334
- Which is the index of the element = 67? → None. It is not in the list
- How many "11" are in the vector? → 3

```
#include <iostream>
using namespace std;

int main()
{
        int numbers[1000] = {
                10      ,600,   381,    206, 46         ,75     ,321,
                628     ,15     ,  343    ,817, 249     ,86     ,416      ,
                523     ,784,   279,    19      ,271    ,495    ,295,
                833     ,633,   611,    96      ,660    ,377    ,409,
                184     ,328,   586,    658, 54         ,446    ,826,
                235     ,155,   478,    50      ,327    ,817    ,163,
```

125	,617,	572,	650, 315	,764	,72 ,	
414	,834,	85	,343, 508	,402	,164,	
474	,707,	475,	548, 643	,277	,770,	
111	,349,	564,	573, 661	,112	,666,	
291	,87	, 849	,533, 615	,843	,697,	
362	,467,	734,	705, 236	,568	,256,	
576	,737,	810,	305, 19,	113	,152	,
797	,801,	746,	189, 51,	771	,764	,
244	,749,	877,	878, 644	,384	,568,	
292	,98	, 72	,331, 117	,154	,145,	
295	,435,	239,	30 ,543	,494	,855,	
651	,382,	188,	201, 581	,733	,397,	
550	,226,	125,	365, 105	,455	,98 ,	
417	,417,	398,	41 ,384	,703	,209,	
431	,442,	402,	645, 438	,487	,395,	
10	,499,	144,	570, 326	,728	,523,	
15	,425,	570,	858, 202	,746	,503,	
638	,147,	754,	33 ,258	,655	,490,	
287	,84	, 36	,356, 162	,579	,497,	
426	,848,	788,	696, 45,	201	,385	,
864	,602,	701,	415, 417	,133	,337,	
786	,812,	49	,843, 857	,183	,136,	
602	,339,	20	,491, 545	,380	,820,	
717	,228,	141,	744, 280	,43,	731 ,	
247	,744,	874,	48 ,788	,395	,74 ,	
387	,287,	87	,626, 795	,311	,584,	
730	,601,	383,	130, 141	,549	,196,	
479	,51	, 820	,650, 493	,517	,652,	
320	,620,	420,	837, 83	,322	,879	,
713	,556,	537,	447, 874	,500	,679,	
503	,171,	467,	19 ,169	,240	,216,	
58	,183,	485,	340, 162	,316	,331,	
367	,32	, 443	,878, 589	,649	,119,	
443	,768,	615,	44 ,70	,550	,658,	
496	,309,	428,	571, 432	,204	,264,	
183	,753,	877,	237, 152	,452	,807,	
758	,210,	227,	542, 868	,409	,440,	
296	,430,	813,	73 ,44	,562	,755,	
815	,812,	416,	32 ,775	,518	,790,	
541	,105,	145,	535, 578	,700	,157,	

451	,545,	149,	626, 592	,24	,493	,
200	,862,	521,	611, 444	,11	,770	,
531	,766,	412,	157, 536	,188	,289,	
484	,192,	670,	477, 325	,69	,179	,
541	,756,	300,	402, 677	,61,	44	,
310	,355,	218,	49 ,665	,375	,581,	
160	,361,	669,	830, 641	,786	,484,	
410	,409,	463,	529, 776	,542	,670,	
852	,400,	522,	428, 221	,117	,378,	
72	,600,	689,	208, 127	,571	,762,	
89	,174,	218,	36 ,402	,431	,452,	
443	,358,	523,	307, 260	,730	,187,	
623	,381,	620,	751, 122	,84	,391	,
390	,375,	614,	690, 718	,468	,691,	
230	,504,	452,	747, 772	,117	,748,	
171	,752,	399,	628, 693	,564	,419,	
803	,823,	55	,357, 80	,798	,752	,
593	,196,	396,	504, 61	,752	,819	,
444	,873,	379,	691, 111	,859	,766,	
546	,381,	636,	81 ,656	,786	,684,	
847	,246,	741,	115, 762	,107	,383,	
10	,751,	349,	217, 869	,408	,379,	
734	,827,	150,	734, 111	,80	,531	,
307	,314,	86	,730, 608	,251	,352,	
497	,382,	677,	206, 171	,247	,769,	
156	,673,	247,	386, 875	,758	,277,	
99	,615,	114,	765, 689	,65	,37	,
808	,125,	513,	585, 96,	240	,336	,
323	,299,	878,	611, 77,	535	,244	,
821	,779,	723,	87 ,252	,341	,523,	
331	,36	, 484	,390, 97	,363	,241	,
671	,454,	689,	821, 604	,682	,245,	
774	,429,	623,	684, 539	,279	,523,	
671	,78	, 624	,493, 602	,62	,315	,
634	,261,	394,	803, 448	,139	,199,	
88	,384,	517,	430, 578	,778	,189,	
745	,363,	507,	726, 143	,413	,261,	
507	,144,	519,	335, 390	,765	,366,	
857	,242,	244,	851, 389	,453	,225,	
831	,797,	757,	173, 311	,526	,761,	

37	,384,	804,	632, 11,	713	,722	,
730	,317,	339,	405, 408	,546	,548,	
451	,90	, 826	,608, 528	,660	,411,	
298	,375,	389,	473, 335	,96	,147	,
595	,752,	307,	248, 228	,13,	15	,
262	,221,	429,	694, 553	,455	,333,	
77	,422,	828,	478, 76,	497	,16	,
187	,588,	263,	471, 143	,123	,345,	
16	,555,	875,	802, 139	,672	,775,	
507	,317,	521,	63 ,877	,136	,191,	
688	,693,	359,	161, 587	,206	,730,	
294	,671,	829,	149, 323	,224	,544,	
304	,279,	870,	226, 129	,113	,84 ,	
576	,56	, 20	,844, 842	,876	,195,	
290	,100,	164,	264, 37,	127	,148	,
227	,636,	586,	302, 490	,764	,582,	
130	,517,	650,	196, 602	,280	,325,	
267	,553,	751,	348, 669	,794	,376,	
724	,188,	563,	80 ,36	,524	,802,	
232	,211,	873,	875, 707	,146	,646,	
424	,829,	148,	875, 728	,51,	173	,
504	,55	, 646	,285, 303	,797	,55 ,	
208	,237,	782,	640, 185	,523	,855,	
855	,88	, 325	,501, 648	,708	,226,	
532	,415,	398,	159, 798	,584	,291,	
709	,193,	30	,634, 877	,648	,827,	
34	,547,	305,	729, 529	,233	,647,	
140	,100,	839,	327, 269	,386	,198,	
135	,538,	870,	460, 654	,379	,324,	
663	,36	, 424	,135, 577	,724	,267,	
120	,349,	619,	276, 790	,329	,285,	
11	,413,	402,	249, 311	,345	,28 ,	
451	,40	, 317	,300, 403	,437	,592,	
764	,715,	41	,577, 233	,168	,617,	
230	,188,	484,	355, 562	,856	,741,	
631	,688,	38	,232, 701	,254	,206,	
79	,593,	671,	337, 65,	662	,844	,
799	,658,	90	,819, 732	,821	,296,	
50	,747,	436,	732, 224	,636	,599,	
783	,693,	494,	614, 394	,667	,527,	

```
                    34      ,526,   835,    533, 210        ,651    ,26 ,
                    490     ,474,   597,    259, 75,        265     ,841    ,
                    780     ,789,   65        ,662, 235     ,432    ,686,
                    654     ,879,   646,    445, 725        ,807    ,58 ,
                    407     ,281,   323,    64         ,429 ,377    ,624,
                    51      ,201,   329,    235, 819        ,861    ,620,
                    690     ,732,   669,    614, 699        ,735    ,557,
                    312     ,616,   284,    74         ,857 ,498    ,471,
                    473     ,673,   81        ,729, 694     ,265    ,123,
                    327     ,789,   146,    801, 418        ,811    ,647,
                    851     ,402,   434,    756, 726        ,320    ,90 ,
                    222     ,623,   140,    140, 764        ,542    ,317,
                    319     ,200,   481,    286, 677        ,828    ,98 ,
                    230     ,123,   679,    208, 476        ,99        ,373,
                    342     ,168,   110,    820, 246        ,831    ,489,
                    291     ,284,   367,    654, 289        ,564    ,503,
                    528     ,354,   164,    783, 586        ,505 };

                    for (int i = 0; i < 1000; i++)
                    {
                            cout << numbers[i] << " ";
                    }
}
```

5. Algorithm finder

- Try to understand the below code. You should use the debugger.
- Write the steps of the algorithm.
- What does each function?
- What is the order of the function calls? (write the code line too)
- This program solves a quadratic equation in standard form: Ax^2 + Bx + C = 0

```
#include <stdio.h>
#include <iostream>
#include <math.h>
using namespace std;


double determinant(double arg_a, double arg_b, double arg_c)
```

```
{
        return (arg_b * arg_b) - (4 * arg_a * arg_c);
}


void realRoots(double arg_a, double arg_b, double arg_sqrt)
{
        double firstRoot = (-arg_b / (2 * arg_a)) +
                (arg_sqrt / (2 * arg_a));
        double secondRoot = (-arg_b / (2 * arg_a)) -
                (arg_sqrt / (2 * arg_a));
        cout << "\nFirst Real Root: \t" << firstRoot << "\n";
        cout << "Second Real Root: \t" << secondRoot << "\n";
}

void output1(double first, double second)
{
        cout << "\nFirst Imaginary Root: \t" << first << " + " << second << "i" << "\n";
        cout << "\nSecond Imaginary Root: \t" << first << " - " << second << "i" << "\n";
}

void output2(double first, double second)
{
        second = -(second);
        cout << "\nFirst Imaginary Root: \t" << first << " - " << second << "i" << "\n";
        cout << "\nSecond Imaginary Root: \t" << first << " + " << second << "i" << "\n";
}

void imaginaryRoots(double arg_a, double arg_b, double arg_imag_sqrt)
{
        double two_a = 2 * arg_a;
        double first_term = (-arg_b) / two_a;
        double second_term = (arg_imag_sqrt) / two_a;

        if (second_term >= 0)
        {
                output1(first_term, second_term);
        }
        else
        {
                output2(first_term, second_term);
        }

}

int main()
{
        double a, b, c;
        char n;
        do
        {
                cout << "\nEnter the value of 'a': ";
                cin >> a;
                cout << "Enter the value of 'b': ";
                cin >> b;
                cout << "Enter the value of 'c': ";
                cin >> c;
                double det = determinant(a, b, c);

                if (det >= 0)
                {
                        realRoots(a, b, sqrt(det));
                }
                else
                {
                        imaginaryRoots(a, b, sqrt(-det));
                }
                cout << "\nWould you like to solve another?";
                cout << "\nEnter 'y' for 'Yes' -- 'n' for 'No': ";
                cin >> n;
                cout << "\n";

        } while (n == 'y' || n == 'Y');

        return 0;

}
```