

The screenshot displays the Visual Studio IDE with a VB.NET code file open. The code defines two parallel arrays, `deptNumArray` and `deptArray`, and a search method `btnSearch_Click`. The search method iterates through the `deptNumArray` and checks if the value in `txtDeptNum.Text` matches any element in the array. If a match is found, it displays the corresponding department name from `deptArray`.

```
3 Dim deptNumArray(3) As Integer
4 Dim deptArray(3) As String
5
6 Windows Form Designer generated code
7
8 References
9
102 Private Sub frmDept_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
103     deptNumArray(0) = 15
104     deptNumArray(1) = 26
105     deptNumArray(2) = 37
106     deptNumArray(3) = 56
107     deptArray(0) = "Books"
108     deptArray(1) = "Toys"
109     deptArray(2) = "Gifts"
110     deptArray(3) = "Cookware"
111 End Sub
112
113 Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSearch.Click
114     Dim indFound As String = "N"
115     Dim wkSub As Integer = 0
116     Do While indFound = "N" And wkSub < deptArray.Length
117         If CInt(txtDeptNum.Text) = deptNumArray(wkSub) Then
118             indFound = "Y"
119         Else
120             wkSub = wkSub + 1
121         End If
122     Loop
123     If indFound = "Y" Then
124         lblShowDeptName.Text = deptArray(wkSub)
125     Else
126         lblShowDeptName.Text = "No match found"
127     End If
128 End Sub
129 End Class
```

A small application window titled "Dept Arra" is overlaid on the code. It contains a text box for "Enter Dept Number", a "Search Dept" button, and a label for "Dept Name". The label displays the value "15" with handwritten blue annotations "86", "37", and "56" next to it, indicating a search for the number 15.

Basic search of two parallel arrays

The screenshot displays the Visual Studio IDE with a VB.NET code file open. The code defines an array of department names and a search function. A small application window titled 'Dept Arra' is running, showing the search results for department 37.

```
3 Dim deptNumArray(3) As Integer
4 Dim deptArray(3) As String
5
6 'Windows Form Designer generated code
7
8
9
102 Private Sub frmDept_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
103     deptNumArray(0) = 15
104     deptNumArray(1) = 26
105     deptNumArray(2) = 37
106     deptNumArray(3) = 56
107     deptArray(0) = "Books"
108     deptArray(1) = "Toys"
109     deptArray(2) = "Gifts"
110     deptArray(3) = "Cookware"
111 End Sub
112
113 Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSearch.Click
114     Dim indFound As String = "N"
115     Dim wkSub As Integer = 0
116     Do While indFound = "N" And wkSub < deptArray.Length
117         If CInt(txtDeptNum.Text) = deptNumArray(wkSub) Then
118             indFound = "Y"
119         Else
120             wkSub = wkSub + 1
121         End If
122     Loop
123     If indFound = "Y" Then
124         lblShowDeptName.Text = deptArray(wkSub)
125     Else
126         lblShowDeptName.Text = "No match found"
127     End If
128 End Sub
129 End Class
```

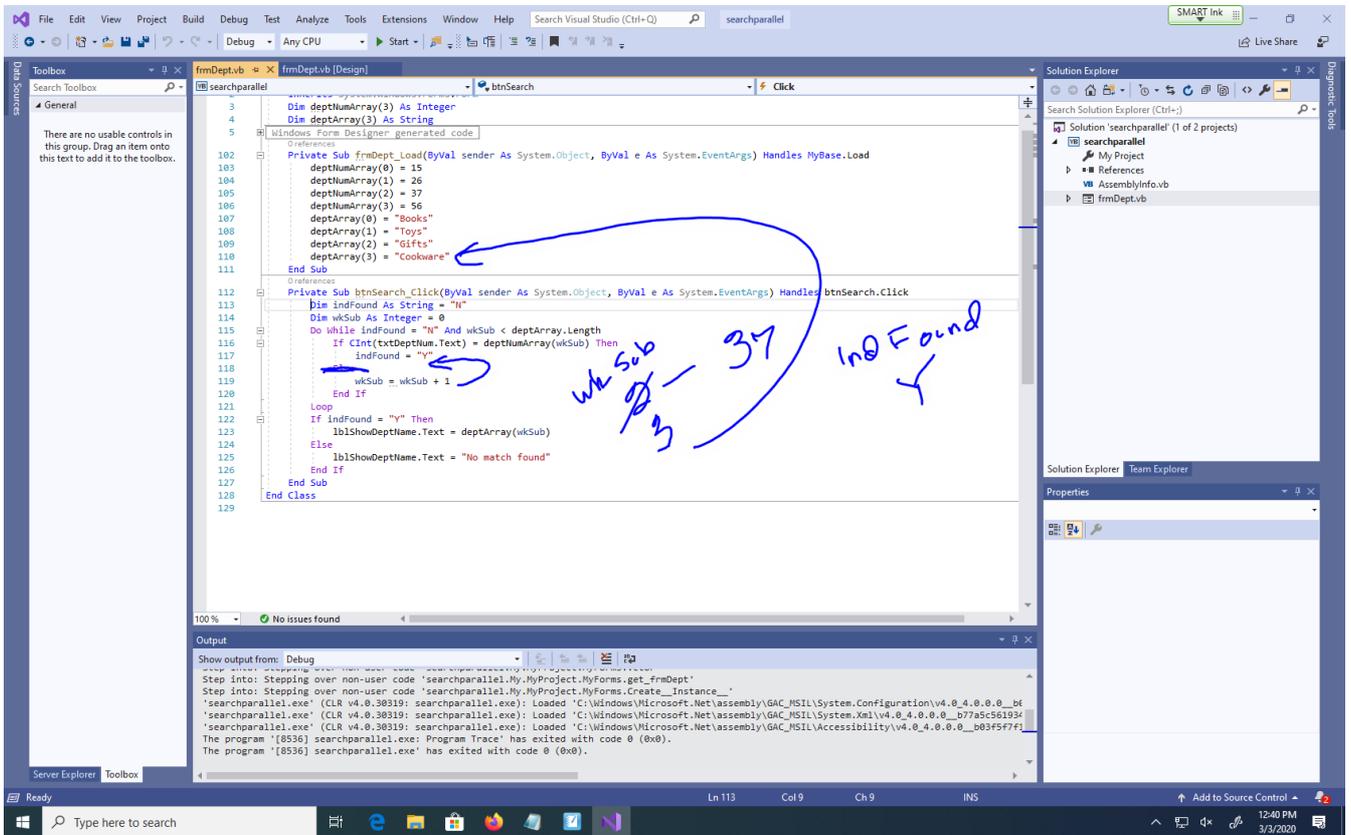
The application window shows:

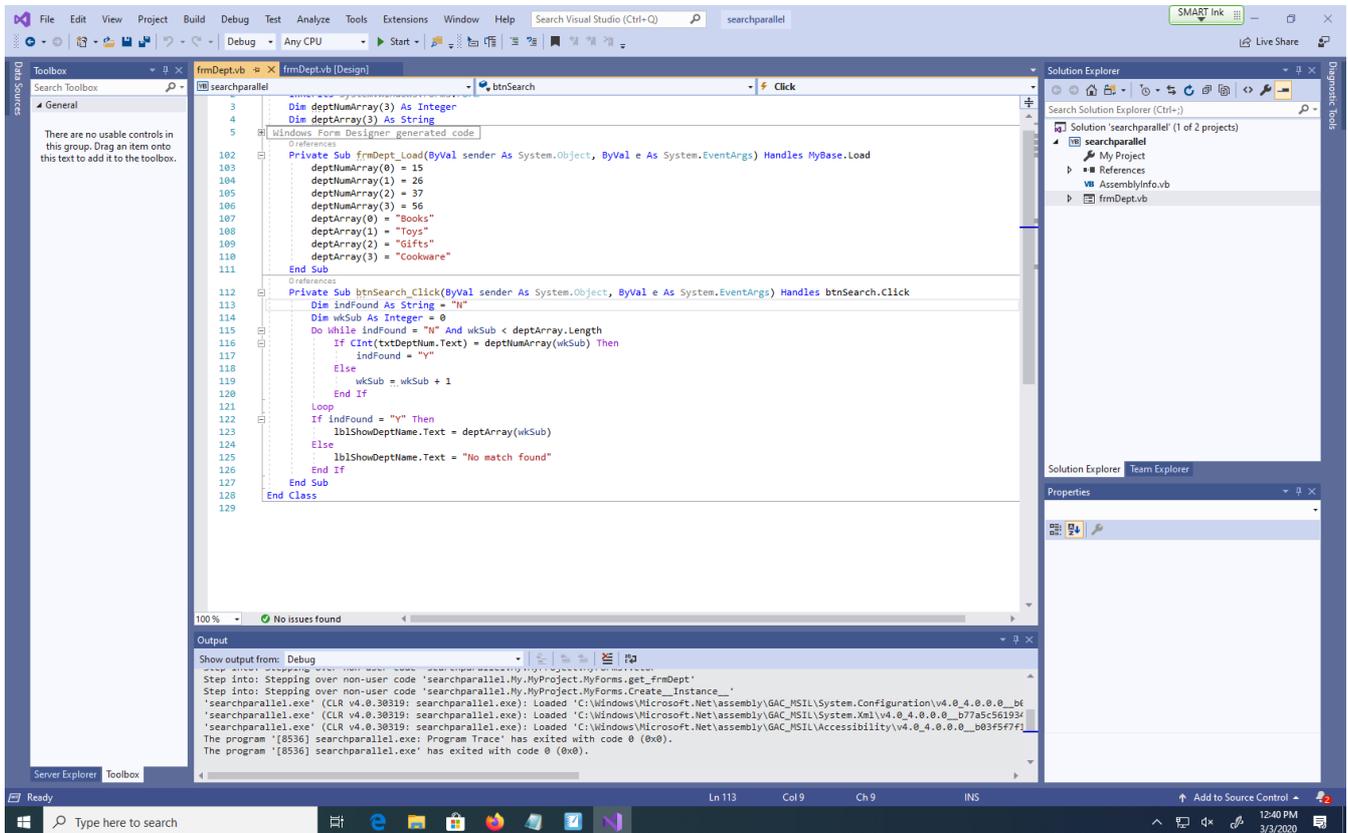
Enter Dept #

Dept Name: **Gifts**

The screenshot displays the Visual Studio IDE with a VB.NET code file named 'frmDept.vb'. The code defines an array of department numbers and names, and a search function. A dialog box titled 'Dept Arra' is overlaid on the code, showing a search input field with the value '2' and the message 'No match found'. The dialog also has a 'Search Dept' button.

```
3 Dim deptNumArray(3) As Integer
4 Dim deptArray(3) As String
5
6 'Windows Form Designer generated code
7
8
9
102 Private Sub frmDept_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
103     deptNumArray(0) = 15
104     deptNumArray(1) = 26
105     deptNumArray(2) = 37
106     deptNumArray(3) = 56
107     deptArray(0) = "Books"
108     deptArray(1) = "Toys"
109     deptArray(2) = "Gifts"
110     deptArray(3) = "Cookware"
111 End Sub
112
113 Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSearch.Click
114     Dim indFound As String = "N"
115     Dim wkSub As Integer = 0
116     Do While indFound = "N" And wkSub < deptArray.Length
117         If CInt(txtDeptNum.Text) = deptNumArray(wkSub) Then
118             indFound = "Y"
119         Else
120             wkSub = wkSub + 1
121         End If
122     Loop
123     If indFound = "Y" Then
124         lblShowDeptName.Text = deptArray(wkSub)
125     Else
126         lblShowDeptName.Text = "No match found"
127     End If
128 End Sub
129 End Class
```





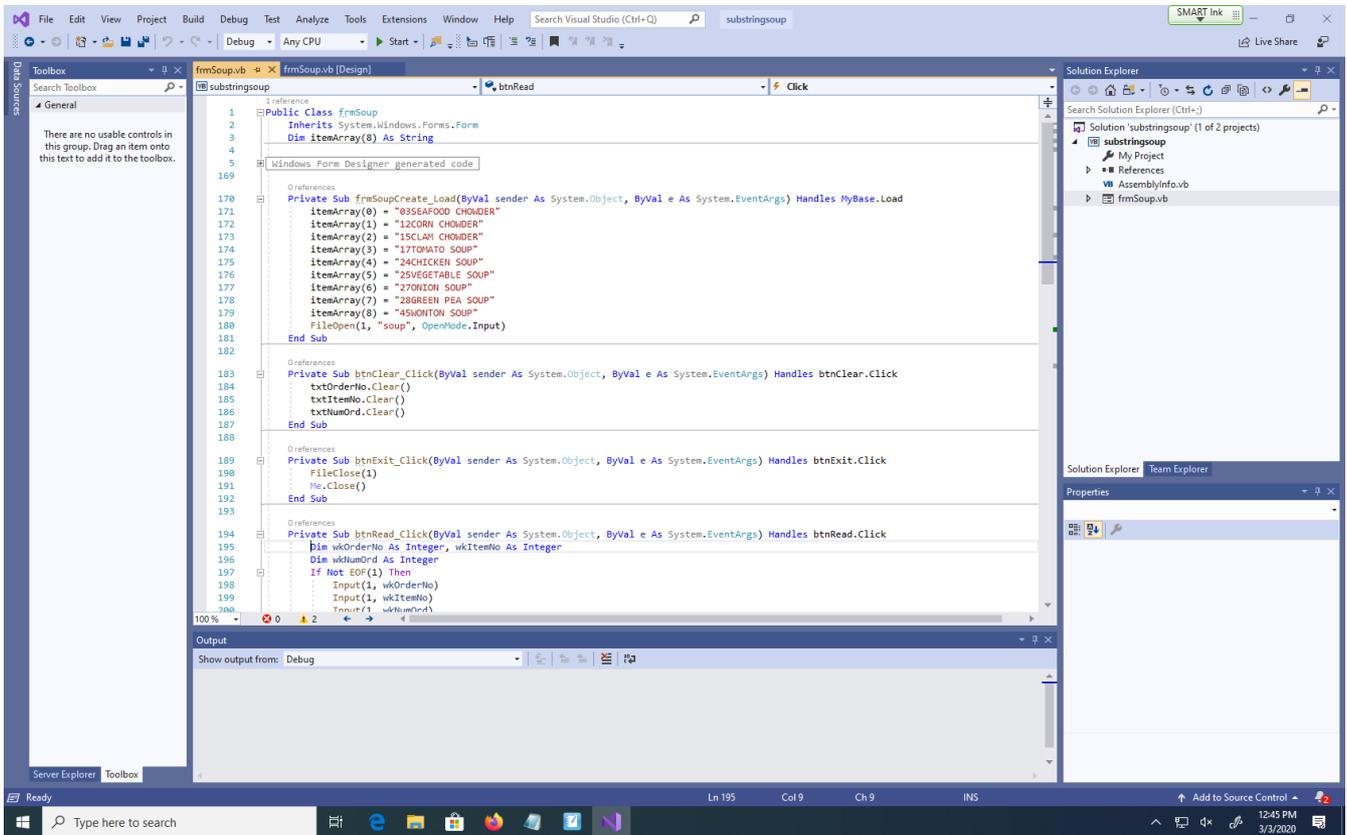
The screenshot shows the Visual Studio IDE with a VB.NET code file open. The code defines an array of department numbers and names, and a search method. A dialog box is displayed over the code, showing the search results for department number 36.

```
3 Dim deptNumArray(3) As Integer
4 Dim deptArray(3) As String
5
6 Windows Form Designer generated code
7
8
9
102 Private Sub frmDept_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
103     deptNumArray(0) = 15
104     deptNumArray(1) = 26
105     deptNumArray(2) = 37
106     deptNumArray(3) = 56
107     deptArray(0) = "Books"
108     deptArray(1) = "Toys"
109     deptArray(2) = "Gifts"
110     deptArray(3) = "Cookware"
111 End Sub
112
113 Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSearch.Click
114     Dim indFound As String = "N"
115     Dim wkSub As Integer = 0
116     Do While indFound = "N" And wkSub < deptArray.Length
117         If CInt(txtDeptNum.Text) = deptNumArray(wkSub) Then
118             indFound = "Y"
119         Else
120             wkSub = wkSub + 1
121         End If
122     Loop
123     If indFound = "Y" Then
124         lblShowDeptName.Text = deptArray(wkSub)
125     Else
126         lblShowDeptName.Text = "No match found"
127     End If
128 End Sub
129 End Class
```

The dialog box titled "Dept Array" contains the following text:

Enter Dept #

Dept Name: **No match found**



The screenshot shows the Visual Studio IDE with a VB.NET code file named `frmSoup.vb` open. The code defines a `Public Class frmSoup` that inherits from `System.Windows.Forms.Form`. It contains several private subroutines: `frmSoupCreate_Load`, `btnClear_Click`, `btnExit_Click`, and `btnRead_Click`. The `frmSoupCreate_Load` subroutine initializes an array of soup names. The `btnRead_Click` subroutine reads from a file named `soup` and displays the results in a dialog box.

The dialog box, titled "Search So...", has the following fields and buttons:

- Order #: 1212
- Item #: 27
- Number Ordered: 2
- Item Name: ONION SOUP
- Buttons: Read, Clear, Exit

The bottom of the screenshot shows the Windows taskbar with the system tray displaying the date and time as 12:48 PM on 3/2/2020.

The screenshot displays a Visual Studio IDE with a VB.NET project named 'substringsoup'. The code in the 'frmSoup.vb' file includes a 'btnRead_Click' event handler and a 'SearchArray' function. The event handler takes an order number and an item number as input, searches for the item, and updates the form's text boxes. The 'SearchArray' function iterates through an array of soup items to find a match. A 'Search Soup' dialog box is shown with the following values: Order #: 1212, Item #: 27, Number Ordered: 2, and Item Name: ONION SOUP. A Notepad window in the foreground shows the following array definition:

```
itemArray(0) = "03SEAFOOD CHOWDER"  
itemArray(1) = "12CORN CHOWDER"  
itemArray(2) = "15CLAM CHOWDER"  
itemArray(3) = "17TOMATO SOUP"  
itemArray(4) = "24CHICKEN SOUP"  
itemArray(5) = "25VEGETABLE SOUP"  
itemArray(6) = "27ONION SOUP"  
itemArray(7) = "28GREEN PEA SOUP"  
itemArray(8) = "45WONTON SOUP"
```

Logic for the Bubble Sort

www.pgrocer.net/Cis56/bubble.html

5	Compare:	1	2	1	2
2	SUB1 pts to 5				5
3	SUB2 pts to 2				3
4	5 not < 2 so flip				4
6					6
2	Compare:	2	3	2	2
5	SUB1 pts to 5				3
3	SUB2 pts to 3				5
4	5 not < 3 so flip				4
6					6
2	Compare:	3	4	3	2
3	SUB1 pts to 5				3
5	SUB2 pts to 4				4
4	5 not < 4 so flip				5
6					6
2	Compare:	4	5		2
3	SUB1 pts to 5				3
4	SUB2 pts to 6				4
5	5<6 leave alone				5
6					6
2	SUB2>END-PT	5	6		
3	so pass complete				
4	5 & 6 locked at				
5	end so 1				
6	subtracted				
	from END-PT				
	for Pass 3				

Bubble sort logic

The image shows a Visual Studio IDE with a C# program implementing bubble sort. The code is as follows:

```
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127
```

Handwritten annotations in blue and red include:

- A red circle around the `Pass` label in the code.
- Blue arrows pointing from the `holdSlot` variable to the `numArray(sub1)` and `numArray(sub2)` comparisons.
- Handwritten text: `Sort`, `Pass`, `holdSlot`, `5`, `sub1`, `sub2`, `flipct`, `endPt`.

The SMART Ink window shows a graphical representation of the array [5, 2, 7, 4, 9] and the sorted array [2, 4, 5, 7, 9].

Handwritten diagrams show the state of variables during a pass:

sub1	sub2	flipct	endPt
0	1	0	4
1	2	1	3
2	3	2	
3	4		
4			

Notes for CIS56

www.pgrocer.net/Cis56/cis56notes.html

Notes for CIS56 - Visual Basic

Notes	Click on links to retrieve
Relational database rules	Relational database - normalization rules
Notes on DreamSpark	DreamSpark Register DreamSpark Account Download Access The same techniques apply to other things you need to download Burn ISO to CD Install and Register (Access)
SQL Server	Using SQL server database Smartboard notes Wiki creating SQL Server Database
Information on color	Color Codes Color chart Color code information and links - interesting information Another color reference
Logic notes	Notes on breaks Notes on top down sort Notes on bubble sort Minor, Intermediate and Major break logic and processing Separate speaker notes to accompany Minor, Intermediate, and Major break logic and processing
Notes on ADO and Access	Notes on ADO Data Controls
Notes on Access	<p>Many examples taken from CIS120:</p> <p>Sample Access 2010 database Note there is an accompanying Presentation explaining this database under presentations (scroll down to see) Note that the Access 2007 databases can be accessed using Access 2010. Sample payroll database using Access 2007 In class project database using Access 2007 Access Database for asgn1 Download steps for downloading Access Database Sample payroll database using Access 2007 In class project database using Access 2007</p> <p>Presentations to accompany examples of Access 2010/2007:</p> <p>Access 2010 example explanation - database is under examples Zipped version of Access 2010 presentation and the database Zipped version of Access 2007 Introduction Access 2007 Inventory example Presentation for asgn1 - Assignment #1 Separate speaker notes to accompany asgn1 Assignment #1 for Access 2007 in pdf format Presentation for if queries in Access Separate speaker notes to accompany if queries in Access Creating a DB in Access 2007 Zipped Donor DB and PowerPoint</p> <p>Earlier versions of Access:</p> <p>Intro to Access 97 Introduction to relationships</p> <p>These notes are from another course. They are provided as a resource. They are not a requirement of this course!</p>

Type here to search

1:19 PM 3/2/2020

Logic of Top-Down Sort: **Top down sort**

15 These are the numbers I want to sort.
 36
 24
 12
 20

Pass 1:

Before	Processing	After
	Set up: Establish 2 subscripts: SUB1=1 and SUB2 = 2	
	Compare:	
15	15 - what SUB1 is pointing to	15
36	36 - what SUB2 is pointing to	36
24	15 < 36 so leave alone	24
12		12
20		20
15	Add 1 to SUB2 so, SUB2 = 3	15
36	Compare:	36
24	15 - what SUB1 is pointing to	24
12	24 - what SUB2 is pointing to	12
20	15 < 24 so leave alone	20
15	Add 1 to SUB2 so, SUB2 = 4	12
36	Compare:	36
24	15 - what SUB1 is pointing to	24
12	12 - what SUB2 is pointing to	15
20	15 not < 12, so flip meaning move what SUB1 is pointing to, to the spot where SUB2 is pointing and	20

→ 15 (12)

36 36 ← 24 (15)

24 24 36 ← 25 (26)

12 15 25 36 ← (25)

20 20 26 25 36

Logic of Top-Down Sort:

15 These are the numbers I want to sort.
 36
 24
 12
 20

Pass 1:

Before	Processing	After
	Set up: Establish 2 subscripts: SUB1=1 and SUB2 = 2	
	Compare:	
15	15 - what SUB1 is pointing to	15
36	36 - what SUB2 is pointing to	36
24	15 < 36 so leave alone	24
12		12
20		20
15	Add 1 to SUB2 so, SUB2 = 3	15
36	Compare:	36
24	15 - what SUB1 is pointing to	24
12	24 - what SUB2 is pointing to	12
20	15 < 24 so leave alone	20
15	Add 1 to SUB2 so, SUB2 = 4	12
36	Compare:	36
24	15 - what SUB1 is pointing to	24
12	12 - what SUB2 is pointing to	15
20	15 not < 12, so flip meaning move what SUB1 is pointing to, to the spot where SUB2 is pointing and	20

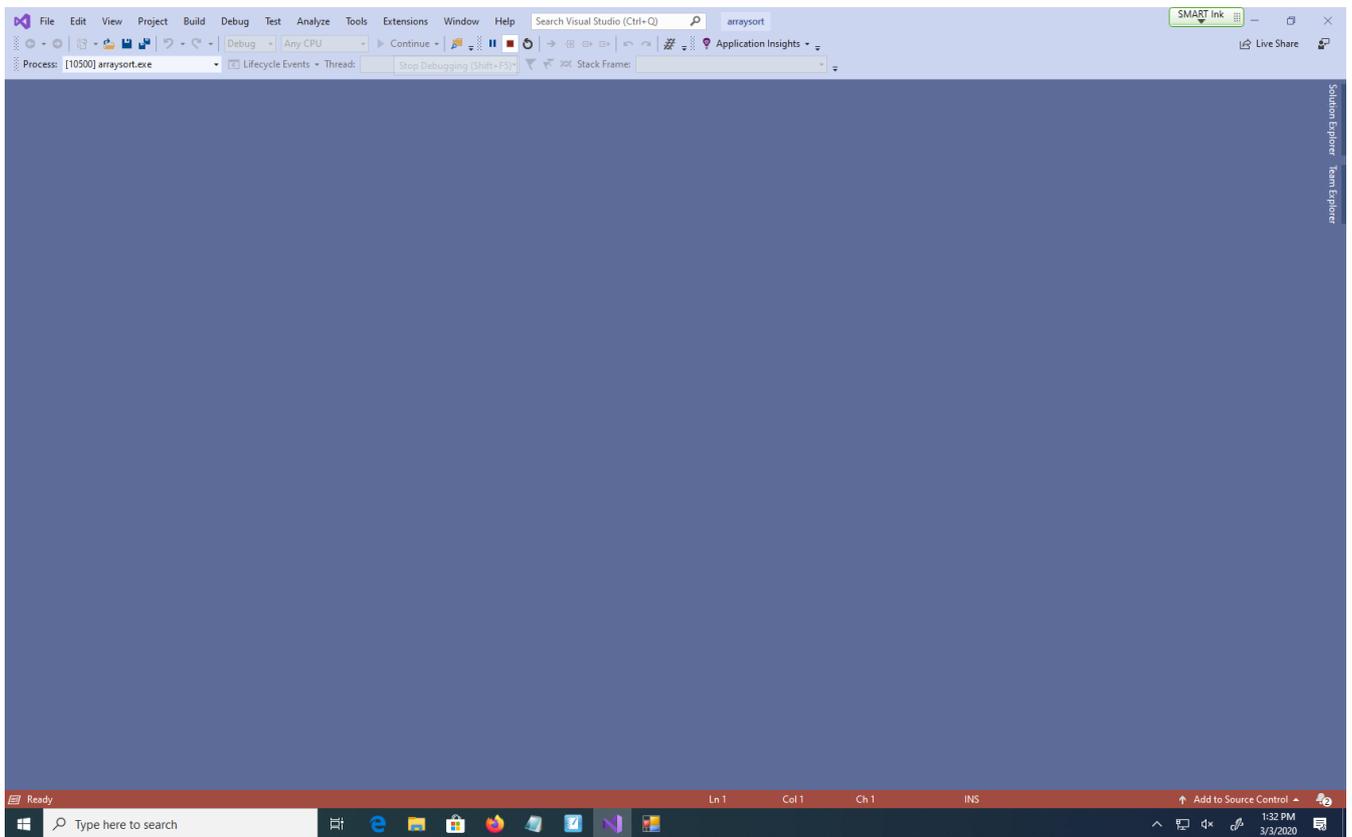
→ 15 (12)

36 36 ← 24 (15)

24 24 36 ← 25 (26)

12 15 25 36 ← (25)

20 20 26 25 36



```
2 Inherits System.Windows.Forms.Form
3 Dim numArray(4) As Integer
4
5 Windows Form Designer generated code
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90 Private Sub btnEnter_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnEnter.Click
91     Dim i As Integer
92     For i = 0 To 4
93         numArray(i) = InputBox("Enter Number", "Sort")
94         lstEnter.Items.Add(numArray(i))
95     Next
96 End Sub
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

Sort Meth SMART Ink

Enter numbers

5
1
9
2
6

Sort

1
2
5
6
9

```
1 reference
2 Public Class frmTestReDim
3     Inherits System.Windows.Forms.Form
4     Dim accumArray() As Integer
5     Dim ct As Integer
6     Dim wkHowMany As Integer
7     Dim wkMore As Integer
8
9     Windows Form Designer generated code
10
11     References
12     Private Sub btnStartGather_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnStartGather.Click
13         wkHowMany = InputBox("Enter number of original entries", "Start")
14         ReDim accumArray(wkHowMany - 1)
15         lstOriginal.Items.Add("Entries: " & Cstr(accumArray.Length))
16         ct = 0
17         btnGatherAmount.Enabled = True
18         btnClear.Enabled = True
19         btnStartGather.Enabled = False
20         txtAmount.Focus()
21     End Sub
22
23     References
24     Private Sub btnGatherAmount_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnGatherAmount.Click
25         accumArray(ct) = CInt(txtAmount.Text)
26         lstOriginal.Items.Add(accumArray(ct))
27         ct = ct + 1
28         btnGatherAmount.Enabled = False
29     End Sub
30
31     References
32     Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClear.Click
33         txtAmount.Clear()
34         If ct < accumArray.Length Then
35             btnGatherAmount.Enabled = True
36             txtAmount.Focus()
37         Else
38             btnClear.Enabled = False
39             btnContinue.Enabled = True
40         End If
41     End Sub
42
43     References
44     Private Sub btnContinue_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnContinue.Click
45         wkMore = InputBox("How many more do you want to enter", "Continue Array")
46         ReDim Preserve accumArray(wkHowMany - 1 + wkMore)
47         lstSecond.Items.Add("Entries: " & Cstr(accumArray.Length))
48         btnContGather.Enabled = True
49         btnClearSecond.Enabled = True
50     End Sub
51
52     References
53     Private Sub btnContGather_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnContGather.Click
54
55     End Sub
56 End Class
```

