

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808 ) By Durga sir Demo On 02-01-2018  
1. Simple:  
-----  
2. Secure:  
-----  
Test.java==>Test.class(compiler)==>Bytecode==>Machine code(JVM)  
Inside JVM: ByteCode Verifier + Security Manager  
Sandbox Checking  
3. Object Oriented  
-----  
in terms of Object  
  
Encapsulation==>Security  
Inheritance==>Reusability  
Polymorphism==>Flexibility  
34  
35  
36
```

```
1. Robust:  
-----  
The chance of failing is very very rare....  
Type Checking: Strongly Typed PL  
Garbage Collector--->memory related problem  
Exception Handling :  
try  
{  
}  
catch(X e)  
{  
}
```

A screenshot of a Java code editor window. The code in the editor is:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int i =10.0;
6         System.out.println(i);
7     }
8 }
```

The line `int i =10.0;` has a red squiggly underline under the number `10.0`, indicating a syntax error. The status bar at the bottom shows "In 5 col 20 9 3B PC ANSI".

Compile time error

A screenshot of a Windows command prompt window titled "cmd.exe". The command entered is `D:\durgaclasses>javac Test.java`. The output shows a compilation error:

```
D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from double to int
        int i =10
                           ^
1 error
```

A context menu is open over the error message, showing options like "Mark", "Copy", "Paste", "Select All", "Scroll", and "Find...". The status bar at the bottom shows "PM 9: 02-01-2018".

Since java is platform independent its easily portable and can be executed in any machine

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808 ) By Durga sir Demo On 02-01-2018
```

```
50 5. Platform Independent:  
51 -----  
52 Write Once and Run Anywhere(WORA)  
53  
54 6. Architecture Neutral:  
55 -----  
56  
57 7. Portability:  
58 -----  
59 mobile portability  
60 IDEA==>Airtel==>  
61  
62 10  
63 10  
64
```

java programming language and all APIs are compiled into bytecodes. Bytecodes are effectively platform-independent. The virtual machine takes care of the differences between the bytecodes for the different platforms

```
For Help, press F1 In 70 col 17 74 00 PC ANSI
```

```
10  
11  
12 10  
13 10  
14  
15  
16 Simple  
17 Secure  
18 OOP  
19 Robust  
20 Platform Independent  
21 Architecture Neutral  
22 Portable  
23 -----  
24 Multi Threaded  
25 Compiled and Interpreted  
26 High Performance  
27 Distributed  
28 Dynamic
```

With architecture-neutral, the book means that the byte code is independent of the underlying platform that the program is running on. For example, it doesn't matter if your operating system is 32-bit or 64-bit, the Java byte code is exactly the same. You don't have to recompile your Java source code for 32-bit or 64-bit. (So, "architecture" refers to the CPU architecture).

"Portable" means that a program written to run on one operating system works on another operating system without changing anything. With Java, you don't even have to recompile the source code; a *.class file compiled on Windows, for example, works on Linux, Mac OS X or any other OS for which you have a Java virtual machine available.

```
For Help, press F1 In 88 col 8 5 00 PC ANSI
```

```
1 Java Basics
2 Operators
3 Flow Control
4 Java Source File Structure
5 OOPs
6 Exception Handling
7 Date and Time API
8 String,StringBuilder
9 Wrapper classes: Integer,Double and Boolean
10 ArrayList
11 Lambda Expressions
12
13
```

For Help, press F1

In 12 col 1 13 00 PC ANSI

```
OCJA 1.8 Java SE 8 Programmer - I(120 - 808 ) By Durga sir Demo On 03-01-2018
Press Esc to exit full screen
42
43 5. Secure:
44 -----
45 Byte code ==>Machine Code==>JVM
46 ByteCode verifier+Security Manager
47 java.lang.VerifyError
48
49 6. Object Oriented :
50 -----
51 Inheritance==>Reusability
52 Encapsulation==>Security
53 Polymorphism==>Flexibility
54 ...
55 7. Robust:
56 -----
57 GC
58 Compiler
59 platform
60 ....
61
```

For Help, press F1

In 55 col 10 66 3A PC ANSI

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) By Durga sir Demo On 03-01-2018
62 8. Multi Threaded:
63 -----
64 tasks==>simultaneously
65 java program==>multiple parts
66
67 9. Distributed:
68 -----
69 10. Compiled and Interpreted:
70 -----
71 |
72
73
74
75
```

The Features of Java.pdf - Adobe Acrobat Reader DC
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga sir Demo On 03-01-2018

Home Tools The Features of Java... x Sign In

10) Compiled and Interpreted:

Java is both Compiled and Interpreted Programming language. First Java compiler compiles java code and generates machine independent Byte Code. At runtime JVM interprets this byte code into machine code and executes that machine code.

11) High Performance:

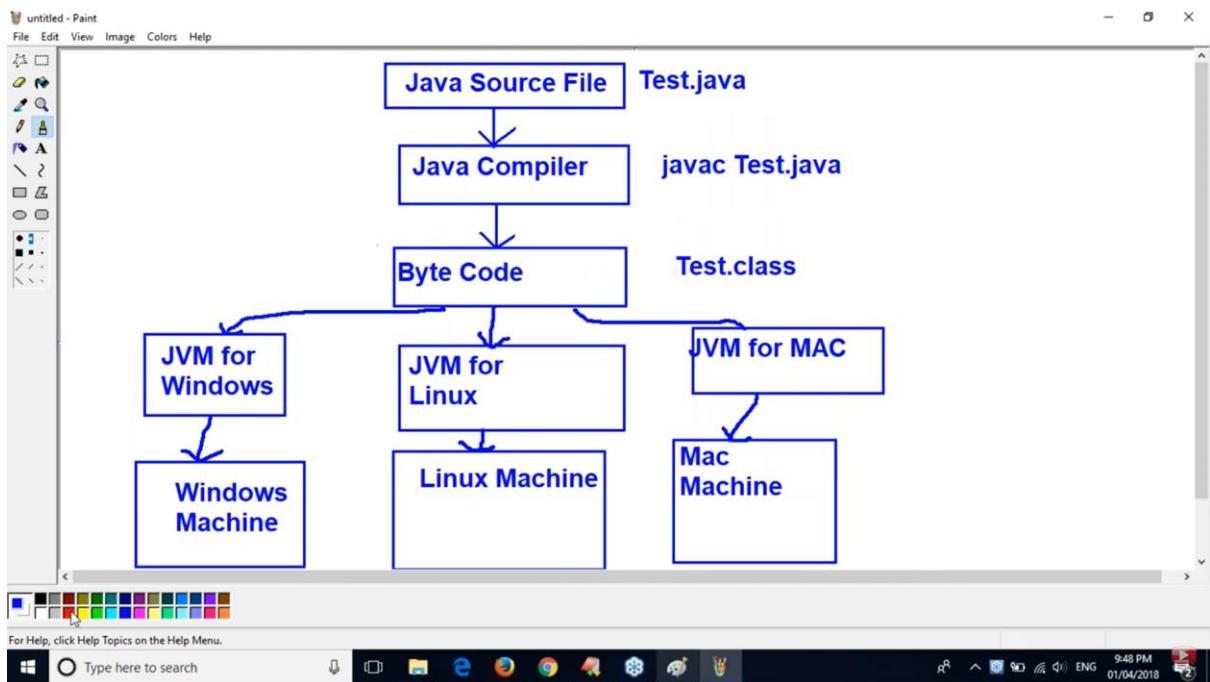
Java is relatively faster than traditional interpreted languages, since byte code is "close" to native code. But Java is still somewhat slower than C or C++.

12) Dynamic:

In the case of Java programs, all .class files won't be loaded at the beginning. At runtime if JVM required any class then only the corresponding .class file will be loaded(Dynamic Loading).The main advantage is program will always get latest version of .class file and memory utilization will be improved.

Explain Platform 58:39 / 1:04:07

Export PDF Convert Create PDF Store and share files in the Document Cloud Learn More

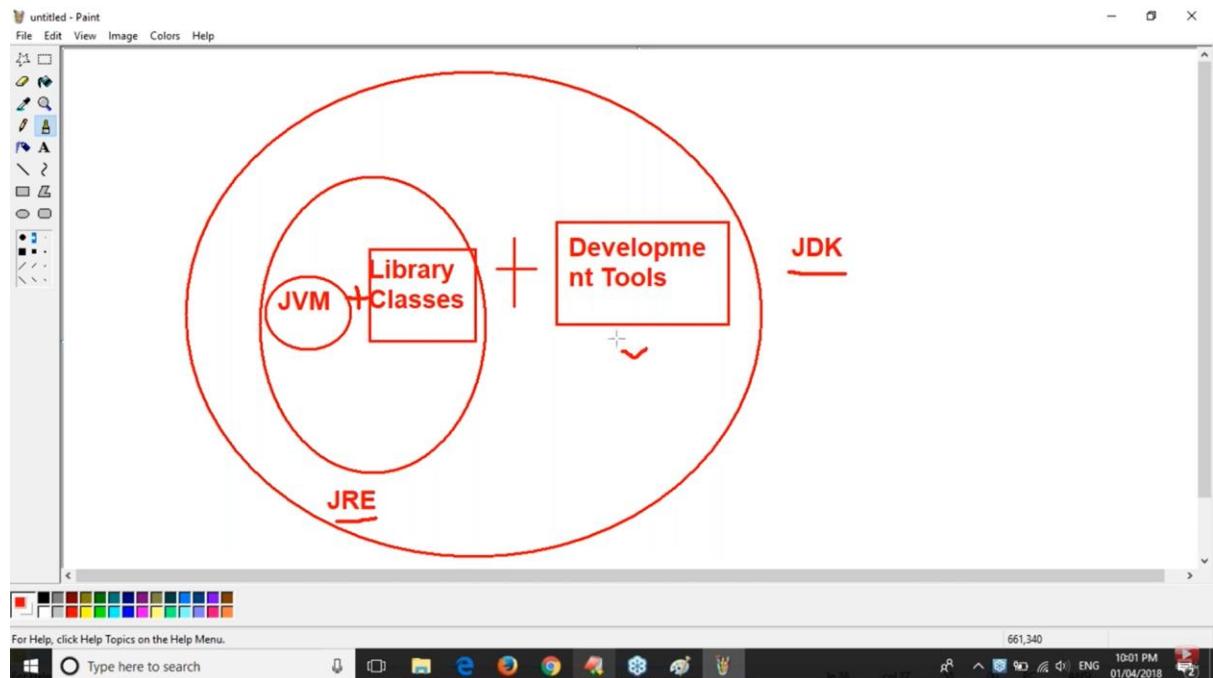


```

OCJA 1.8 Java SE 8 Programmer - I (120 - 808 ) By Durga sir Demo On 04- 01- 2018
1
2
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 JDK vs JRE vs JVM:
29 -----
30 JDK(Java Development Kit):
31 -----
32 To develop and run java applications
33
34 JRE(Java Runtime Environment):
35 -----
36 To run java applications
37
38 JVM(Java Virtual Machine):
39 -----
40 is an interpreter to run program line by line
41
42
43
44
45
46
47
48
49

```

All development tools in bin folder compiler and debugger part of Development tools



```
OCJA 1.8 Java SE 8 Programmer - 1 (1Z0-808) By Durga sir Demo On 04-01-2018
122 Q. Which of the following is true?
123
124 A) Java is platform dependent but JVM is platform independent
125 B) Java is platform independent but JVM is platform dependent
126 C) Java Byte code is platform dependent but JVM is platform independent
127 D) Java Byte code is platform independent but JVM is platform dependent
128
129 B and D
130
131
132 Q. Which Statement is true about Java Byte code?
133
134 A) It can run on any platform
135 B) It can run on any platform only if it was compiled for that platform
136 C) It can run on any platform that has the Java Runtime Environment (JRE)
137 D) It can run on any platform that has a Java Compiler
138 E) It can run on any platform only if that platform has both JRE and Java Compiler
139
140
141
```

This is a screenshot of a terminal window displaying Java SE 8 Programmer - 1 (1Z0-808) demo code. The code consists of several numbered lines, likely representing a script or a series of commands. Lines 122 through 129 are a question about Java being platform-dependent or independent. Lines 130 through 138 are a question about Java Byte code running on different platforms. Lines 139 through 141 are blank. The terminal window has a standard Windows-style interface with a title bar, menu bar, and status bar at the bottom.

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808 ) By Durga sir Demo On 04-01-2018
```

130
131
132 Q. Which Statement is true about Java Byte code?
133
134 A) It can run on any platform
135 B) It can run on any platform only if it was compiled for that platform
136 C) It can run on any platform that has the Java Runtime Environment (JRE)
137 D) It can run on any platform that has a Java Compiler
138 E) It can run on any platform only if that platform has both JRE and Java Compiler
139
140
141

Invalid type missing

```
Untitled1 Java_Features.txt Untitled2.java
```

For Help, press F1 28:57 / 54:16

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         x=10;
6         System.out.println(x);
7     }
8 }
```

In 137 col 1 2 44 PC ANSI

A screenshot of a Java code editor window. The code in the editor is:

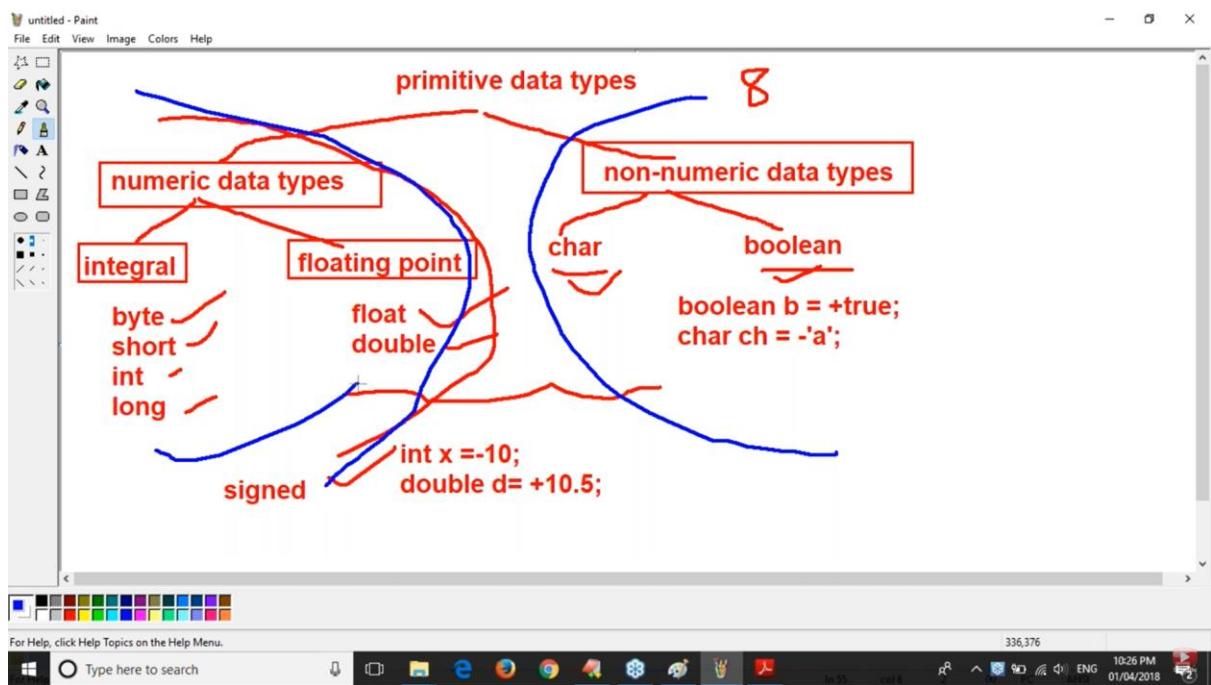
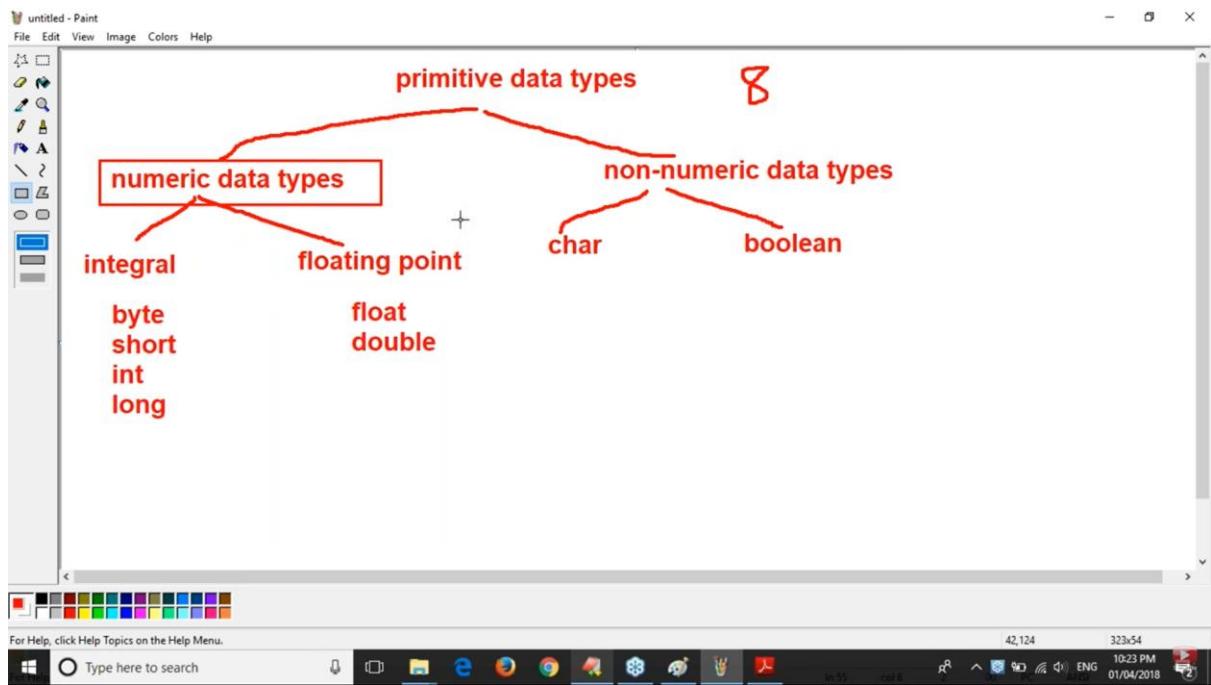
```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x = 10.5;
6         System.out.println(x);
7     }
8 }
```

The code editor has a toolbar at the top with various icons. Below the toolbar is a menu bar. The status bar at the bottom shows "For Help, press F1", "In 5 col 21 9 00 PC ANSI", and a video camera icon.

A screenshot of a presentation slide titled "OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga sir Demo On 04-01-2018". The slide content is:

```
36 To run java applications
37
38 JVM(Java Virtual Machine):
39 -----
40 is an interpreter to run program line by line
41
42
43 Strongly Typed PL I
44
45 Java is Pure OOP?
```

The slide has a navigation bar at the bottom with icons for back, forward, and search. The status bar at the bottom shows "For Help, press F1", "In 46 col 1 61 00 PC ANSI", and a video camera icon.

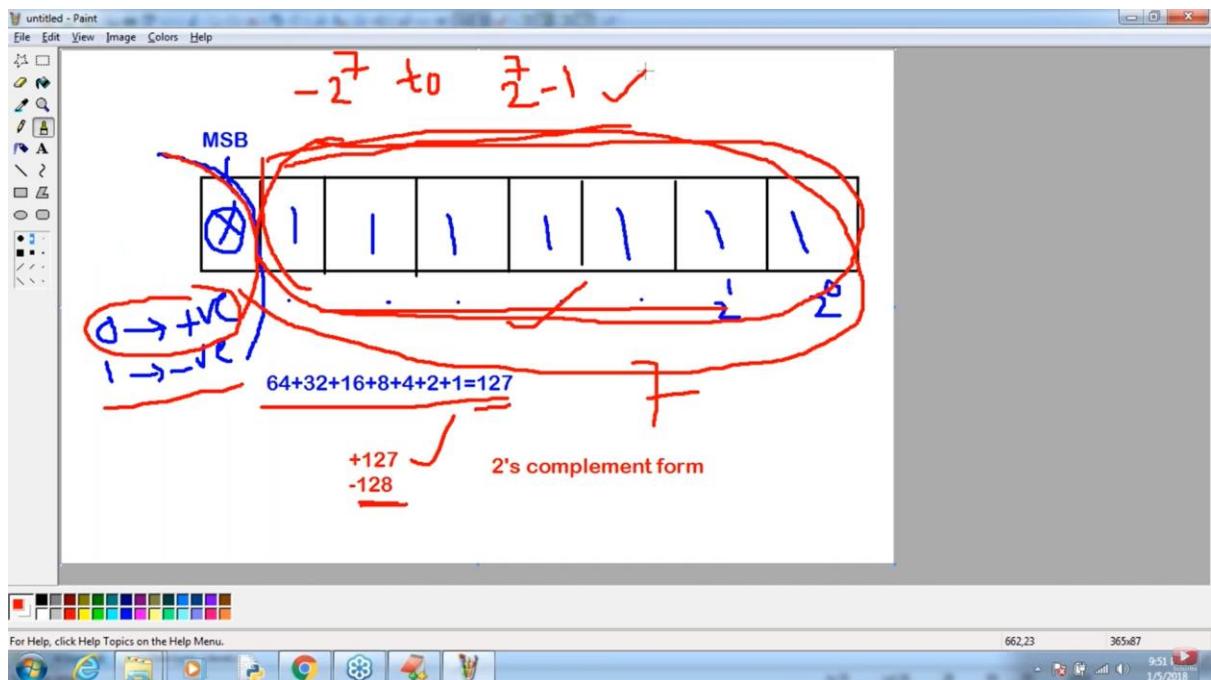


```

1 OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) By Durga sir Demo On 05-01-2018
2 -----
3 Data Types and Literals:
4 -----
5 Numbers:
6 1. Integral Types
7 10,20,100,100000
8 byte,short,int,long
9
10 2. Floating Point Types
11 10.0,12.123,65.78
12 float and double
13
14 Non-numeric data types
15
16 2: char and boolean
17 'a', '@'
18 true,false ==>boolean
19
20
21

```

Positive nos are directly stored in memory whereas negative nos are rep in 2's compliment so we can store upto 128



A screenshot of a Java IDE interface. The code editor window shows the following Java code:

```
1 ----
2 size=1 byte(8 bits)
3 Range: -128 to +127
4
5
6
7
8
```

The code editor has tabs for "Untitled1" and "Data types_Literals". The status bar at the bottom shows "In 26 col1 40 00 PC ANSI".

A screenshot of a Windows command prompt window titled "cmd.exe". The command line session is as follows:

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

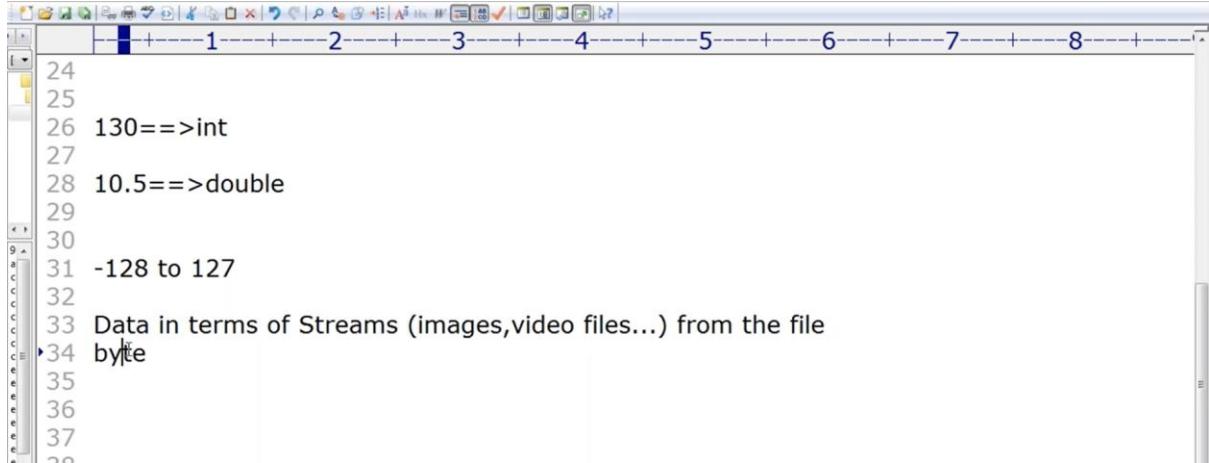
C:\Users\LENOVO>d:
D:>cd durgaclasses
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
10

D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from int to byte
        byte b =130;
                           ^
1 error
D:\durgaclasses>
```

The taskbar at the bottom shows various application icons.

By default every integral type is int and every floating point are double

Wherever files or streams are rep in bytes



```
24
25
26 130==>int
27
28 10.5==>double
29
30
31 -128 to 127
32
33 Data in terms of Streams (images,video files...) from the file
34 byte
35
36
37
38
```

Rarely used data type is Short – 2bytes

Olden days we were using 16 bit processor, now java 9 using 64 bits . we are not using short anymore. But can be used



```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) By Durga sir Demo On 05-01-2018
39 -139
40
41 short:
42 -----
43 size=2 bytes(16 bits)
44 Range: -2^15 to 2^15-1
45 : -32768 to 32767
46
```

```
1-----2-----3-----4-----5-----6-----7-----8-----9-----  
45 : -32768 to 32767  
46  
47 50000  
48 int:  
49 ----  
50 size=4 bytes(32 bits)  
51 Range: -2^31 to 2^31-1  
52 : -2147483648 to 2147483647I  
53  
54  
55 ----
```

```

84 char data type:
85 -----
86 char ch ='a';
87
88 size : 2 bytes
89 C,C++=>ASCII
90 <256 characters ==>8 bits
91 Java is unicode
92 >256 and <65536
93
94 boolean data type:
95 -----
96 true, false
97 size: NA
98 Range: NA
99
100
101
102

```

The screenshot shows a Java code editor with several lines of code. Lines 84-93 define the characteristics of a 'char' data type. Lines 94-98 define the characteristics of a 'boolean' data type.

data type	Size	Range	Corresponding Wrapper class	Default v
byte	1 byte	-2^7 to 2^7-1 (-128 to 127)	Byte	0
short	2 bytes	-2^{15} to $2^{15}-1$ (-32768 to 32767)	Short	0
int	4 bytes	-2^{31} to $2^{31}-1$ (-2147483648 to 2147483647)	Integer	0
long	8 bytes	-2^{63} to $2^{63}-1$	Long	0
float	4 bytes	-3.4e38 to 3.4e38	Float	0.0
double	8 bytes	-1.7e308 to 1.7e308	Double	0.0
boolean	Not applicable	Not applicable (but allowed values true/false)	Boolean	false
char	2 bytes	0 to 65535	Character	0 (represents blank space)

Data Type Default Value (for fields)

byte 0
short 0
int 0
long 0L
float 0.0f
double 0.0d
char '\u0000'
String (or any object) null
boolean false

Note:
The default value for the object references is "null"



```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 08 - 01 - 2018  
-----  
23 1. Decimal (base-10):  
24 -----  
25 0 to 9  
26 int x =12;  
27  
28 2. Octal(base-8):  
29 -----  
30 0 to 7  
31 int x =010;  
32  
33 3. Hexa decimal(base-16):  
34 -----  
35 0 to 9,a to f  
36  
37 int x = 0X10;  
38 int x = 0x10;  
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  
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  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1098  
1099  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1198  
1199  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1298  
1299  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1398  
1399  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1498  
1499  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1598  
1599  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1698  
1699  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1798  
1799  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1889  
1890  
1891  
1892  
1893  
189
```

```
C:\Users\LENOVO>d:  
D:>cd durgaclasses  
D:\durgaclasses>javac Test.java  
D:\durgaclasses>java Test  
D:\durgaclasses>javac Test.java  
Test.java:5: error: integer number too large: 0786  
        int x =0786;  
                  ^  
1 error  
D:\durgaclasses>
```

```
3. Hexa decimal(base-16):  
-----  
0 to 9,a to f  
-----  
int x = 0X10;  
int x = 0x10;  
-----  
int x =0777;  
int x = 0786;-->CE  
int x = 0XFace;  
int x = 0XBeef;  
int x = 0XBeer;I
```

This type or multiple arguments in line 9 are not allowed, we can pass only 1 variable like line 8

The screenshot shows a Java code editor with the following code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x =10;
6         int y = 010;
7         int z=0X10;
8         System.out.println(x+..+y+..+z);
9         System.out.println(x,y,z); // Error
10    }
11 }
12
```

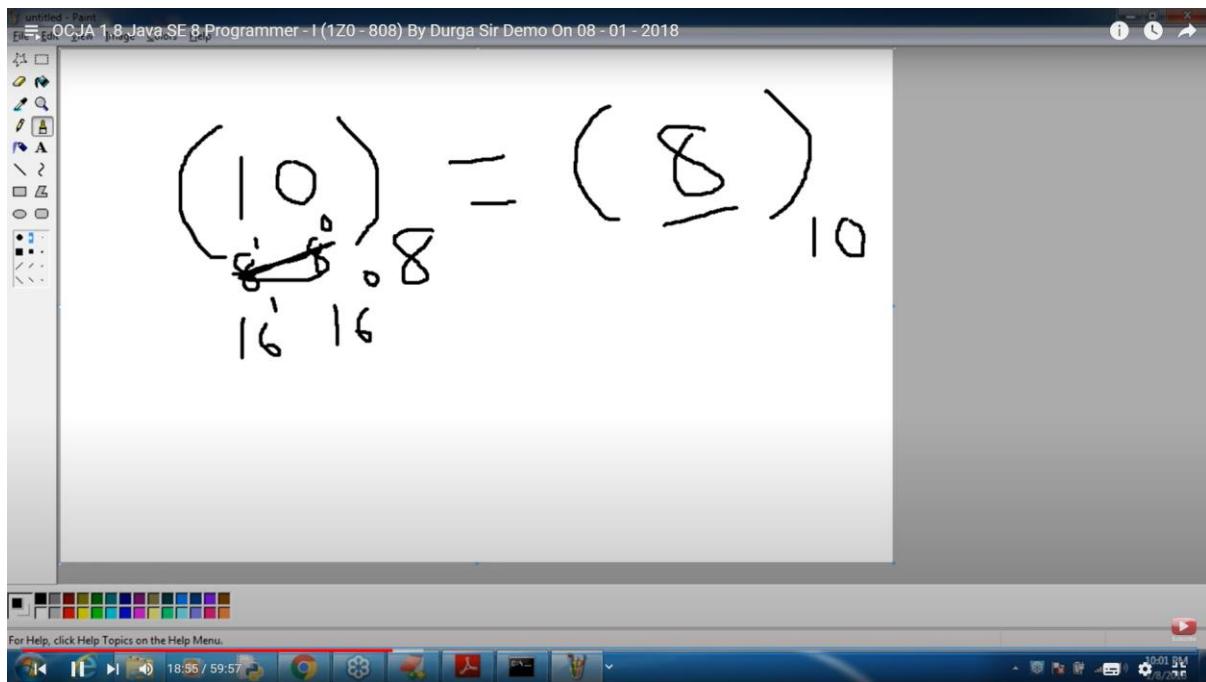
The line 9, `System.out.println(x,y,z);`, is highlighted in blue, indicating a syntax error. The status bar at the bottom shows "In 9 col 35".

The screenshot shows the same Java code editor after the error was fixed. The code now looks like this:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x =10;
6         int y = 010;
7         int z=0X10;
8         System.out.println(x+..+y+..+z);
9     }
10 }
11
```

The status bar at the bottom shows "In 8 col 45".

System will consider only in decimal value



```

1 error
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
10..8..16

```

int x = 0786;-->CE
int x = 0XFace;
int x = 0XBeef;
int x = 0XBeer;

10
0777
0XFace

int
10L long

X	Question	Asker
1	x=10 y=21 z=10	Shubham Gupta
2	16	Shubham Gupta
3	10..8..16	Shubham Gupta
4	no	Kalyan KK
5	understand	Kalyan KK
6	s	Pooja Chavan
7	3	Shubham Gupta
8	3	Kalyan KK
9	yes	Pooja Chavan
10	yes	Kalyan KK
11		Shubham Gupta

Good Evening Sir

Send Privately Send to All

A screenshot of a Windows Command Prompt window titled "Select C:\Windows\system32\cmd.exe". The command "javac Test.java" is run three times, resulting in the same error message each time:

```
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from long to int
        int x =10L;
                           ^
1 error
D:\durgaclasses>_
```

The window has a vertical scroll bar on the left and a horizontal scroll bar at the bottom. The taskbar at the bottom shows various icons for programs like File Explorer, Internet Explorer, and Google Chrome.

By Default every integral (octal, decimal and Hexa) is of int type. If you want to specify as Long suffix with L

A screenshot of an IDE window titled "OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 08 - 01 - 2018". The code editor displays the following Java code:

```
48 10
49 0777
50 0XFace
51
52 int x =10;
53 long l =10;
54
55 long l =10L;
56 int x =10L;
```

The code uses various integer literals (10, 0777, 0XFace) and demonstrates different ways to declare variables of type int and long. The variable declarations are highlighted in blue, while the literals are in black. The IDE interface includes a toolbar, a status bar at the bottom, and a menu bar at the top.

The below is wrong. Cant explicitly specify byte or short

A screenshot of a Java code editor window. The code is as follows:

```
59
60
61 100
62
63
64 byte b = 100b;
65 short s = 100s;
66
67
68
69
70
71
72
73
74
75
76
77
78
```

The line `byte b = 100b;` has a syntax error, indicated by a red underline. The line `short s = 100s;` also has a syntax error, indicated by a red underline. The status bar at the bottom shows "For Help, press F1".

A screenshot of a Java code editor window. The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         byte b =128;
6         System.out.println(b);
7     }
8 }
9
```

The line `byte b =128;` has a syntax error, indicated by a red underline. The status bar at the bottom shows "OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 08 - 01 - 2018", "For Help, press F1", and "31:33 / 59:57".

```
C:\Windows\system32\cmd.exe
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from long to int
        int x =10L;
                  ^
1 error
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
100

D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
127

D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from int to byte
        byte b =128;
                  ^
1 error
D:\durgaclasses>
```

```
1 59
2 60
3 61 100
4 62
5 63
6 64 -128 to 127
7 65
8 66 byte b = 100;
9 67
10 68 short: -32768 to 32767
11 69
12 70 short s=100;
13 71 short s = 32767;
14 72 short s = 32768;
15 73
16 74
17 75
18 76
19 77
20 78
```

By default floating point literals are of double type can't assign to float

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         float f = 123.456;
6     }
7 }
```

X	Question	Asker
invalid	Shubham Gupta	
double type	john acharya	
invalid	Shubham Gupta	
invalid..	Kunal Danch	
no	Pooja Chavan	
123.456f	Shubham Gupta	
double	Shubham Gupta	
int	john acharya	
double type	Shubham Gupta	
double	Pooja Chavan	
double typee	Kalyan KK	

Good Evening Sir

Send Privately Send to All

Suffix with small f or capital F

```
1 public static void main(String[] args)
2 {
3     float f = 123.456F;
4 }
5
6
7
8
9
```

Line 90 alone error

```
86
87
88
89
90 float f = 123.456;
91 double d = 123.456;
92 float f = 123.456f;
93 double d = 123.456F;
94 double d = 123.456D;
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) By Durga Sir Demo On 08-01-2018  
1 2 3 4 5 6 7 8  
95 -----  
96 -----  
97 -----  
98 integral literals:  
99 -----  
100 decimal, octal and hexa decimal  
101  
102 floating point  
103  
104 double d = 123.456;  
105 double d = 0123.456;  
106 double d = 0X123.456;  
107
```

Floating Point only supports decimal.

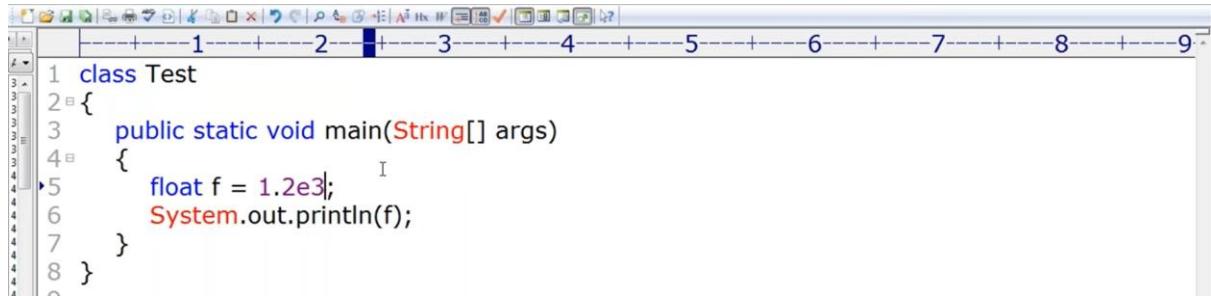
Double d = 0123.456 will be considered as floating point and ignore the 0 on printing the 0 will be truncated .So its valid.

Whereas 0x → x is an invalid character and will throw an error.

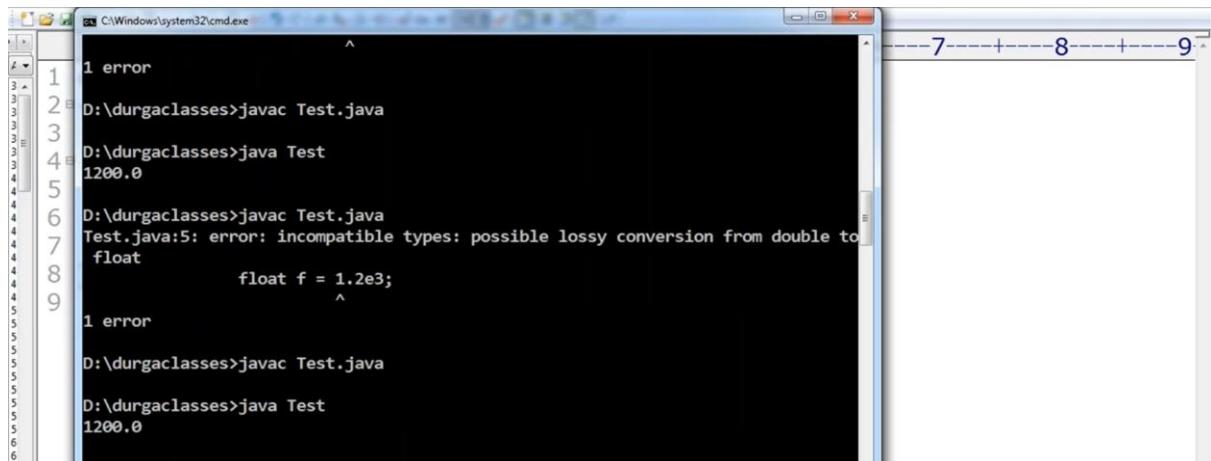
```
D:\durgaclasses>java Test  
123.456  
D:\durgaclasses>javac Test.java  
Test.java:5: error: malformed floating point literal  
      double d = 0X123.456;  
                         ^  
1 error
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) By Durga Sir Demo On 08-01-2018  
1 2 3 4 5 6 7 8  
101  
102 floating point==>only decimal  
103  
104 double d = 123.456;  
105 double d = 0123.456;  
106 double d = 0X123.456;  
107 -----  
108  
109 Exponential form  
110  
111 double d = 1.2e3;=>1.2*10^3==>1.2*1000==>1200  
112  
113
```

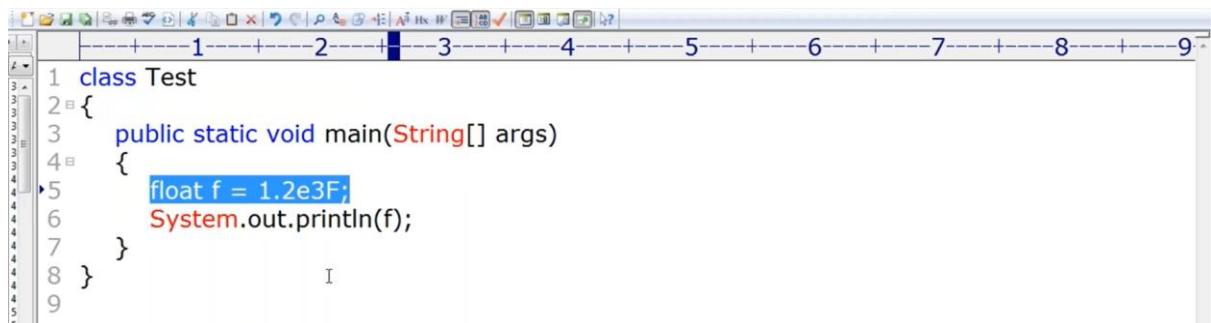
```
5 D:\durgaclasses>javac Test.java
5
5 D:\durgaclasses>java Test
5 1200.0
6
6 D:\durgaclasses>
```



```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         float f = 1.2e3;
6         System.out.println(f);
7     }
8 }
```



```
C:\Windows\system32\cmd.exe
1 error
2
3 D:\durgaclasses>javac Test.java
4
5 D:\durgaclasses>java Test
5 1200.0
6
7 D:\durgaclasses>javac Test.java
8 Test.java:5: error: incompatible types: possible lossy conversion from double to
9     float
          float f = 1.2e3;
           ^
1 error
2
3 D:\durgaclasses>javac Test.java
4
5 D:\durgaclasses>java Test
5 1200.0
6
```



```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         float f = 1.2e3F;
6         System.out.println(f);
7     }
8 }
9
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 08 - 01 - 2018
108
109 Exponential form
110
111 double d = 1.2e3;==>1.2*10^3==>1.2*1000==>1200.0
112
113 -----
114
115 sir can't we say for floating point decimal and octal are allow bcz we are always using 0
for octal
116
117 int x =010;
118 sop(x); //8
119
120 double d = 0123.456;
121 sop(d); 123.456
122
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 17 - 01 - 2018
10 10000 values
11
12 int[] x = new int[10000];
13 x[0],x[1],x[2]...
14 -----
15 Fixed in size
16 x[0]=10;
17 x[1]=20;
18 x[2]=true;
19 homogeneous==>same type
20
21 An array is an indexed collection of fixed number of homogeneous data elements
22 -----
23
```

Autify - Automation Testing
Autify, the most advanced AI-powered software testing automation platform
app.autify.com

* Untitled1 11:49 / 1:11:53

In 23 col 1 23 00 PC ANSI

```
21 An array is an indexed collection of fixed number of homogeneous data elements
22 -----
23 collections
24 -----
25 Array declaration:
26 -----
27 1 dimensional :
28
29 int[] x;
30 int []x;
31 int x[];
32
33
```

A red box highlights the code `int[] x;` in line 29. A red arrow points from the left towards this highlighted code.

The code editor interface includes tabs like Untitled1, Test.java, and a status bar showing 15:31 / 1:11:53. A sidebar on the right contains a "Questions" section with a list of user posts and answers.

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x;
6         int []y;
7         int z[];
8         int a[] [];
9     }
10 }
```

A red box highlights the code `int a[] [];` in line 8. A red arrow points from the left towards this highlighted code.

The code editor interface includes tabs like Untitled1, Test.java, and a status bar showing 15:31 / 1:11:53. A sidebar on the right contains a "Questions" section with a list of user posts and answers.

Space ignored by compiler

Cant specify size at time of declaration but can be done at the time of creation

```
2 --  
3  
4  
5  
6  
7  
8  
35 int[4] x;
```

```
4  
5  
6 } } D:\durgaclasses>javac Test.java  
7 Test.java:5: error: ]' expected  
8     int[x] x;  
      ^  
Test.java:5: error: not a statement  
     int[4] x;  
          ^  
Test.java:5: error: illegal start of expression  
     int[4] x;  
          ^  
3 errors
```

All the below are valid

```
37  
38  
39 int[][] x;  
40 int [][]x;  
41 int x[][];  
42  
43  
44 int[] []x;  
45 int[] x[];  
46 int []x[];  
47  
48  
49  
50
```

```
49 int[][][] x;  
50 int [][][]x;  
51 int x[][][];  
52  
53 int[][] []x; I  
54 int[][] x[];  
55  
56 int[] [][]x;  
57 int[] []x[];  
58 int[] x[][];  
59 int [][]x[];  
60 int []x[][];  
61
```

Sppace will be ignored both are 2D

A screenshot of a Java code editor window. The title bar reads "OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) By Durga Sir Demo On 17-01-2018". The code area shows the following lines:

```
61
62 int[] a,b;//a-1D, b-1D
63 int[] a,b[];a-1,b-2
64
65
66 int[] []a,b; a-2,b-2
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
```

The line "int[] []a,b; a-2,b-2" is highlighted with a blue selection bar under "int[]". The status bar at the bottom right shows "In 66 col1 81 69 PC ANSI".

Declaring the dimension is only allowed before the first variable, not before the second variable.

Below will throw an error.

A screenshot of a Java code editor window. The title bar has many icons. The code area shows the following lines:

```
61
62 int[] a,b;//a-1D, b-1D
63 int[] a,b[];a-1,b-2
64
65
66 int[] []a,b; a-2,b-2
67 int[] []a,b[]; a-2,b-3
68
69 int[] a,[]b;
70
71
72
73
74
75
76
77
78
79
80
81
```

The line "int[] a,[]b;" is highlighted with a yellow rounded rectangle. To its right, the text "you can have [] before the first variable not in the middle" is written. The status bar at the bottom right shows "In 69 col10 84 5B PC ANSI".

```
5  
5  
5 D:\durgaclasses>javac Test.java  
5 Test.java:5: error: <identifier> expected  
5     int[] []a,[]b;  
5                     ^  
5 Test.java:5: error: ';' expected  
5     int[] []a,[]b;  
5                     ^  
5  
5 2 errors  
5  
5 D:\durgaclasses>
```

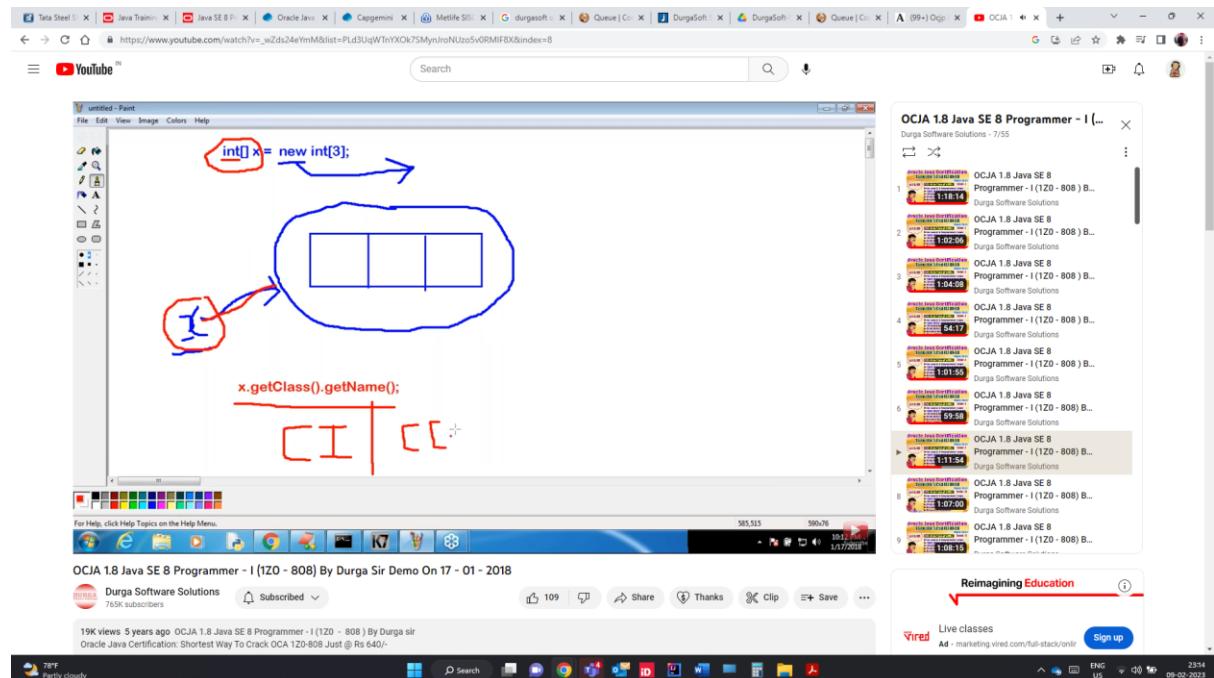
73- Error 74 right

The screenshot shows a Java code editor with the following code:

```
1 int[] []a,b[]; a-2,b-3  
2  
3 int[] a,[]b;  
4  
5  
6 int[] a,b,[]c,d[];  
7 int[] a[],b[],c[];  
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
```

The cursor is positioned at line 74, character 19, where the IDE highlights the closing brace of the previous line. The status bar at the bottom indicates "In 74 col 19".

For every array type the class files are available at language level not at API level



X – is object reference . x.getClass().getName();

The class name is [[I – [ref 1 dimension I rep int; two dimensions [[I

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[][] x = new int[4][3];
6         System.out.println(x.getClass().getName());
7     }
8 }
9 }
```

```
1
2
3
4
5
6
7
8
9
10
```

```
D:\durgaclasses>java Test
[[[I
D:\durgaclasses>
```

Knowledge purpose

```
79 -----
80 int[] x = new int[4];
81
82
83
84 int[]-->[I
85 int[][] -->[[I
86 int[][][]-->[[[I
87 byte[]-->[B
88 long[] -->[L
89 boolean[] ---[Z
90
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 17 - 01 - 2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         boolean[] b=new boolean[3];
6         System.out.println(b.getClass().getName());
7
8     }
9 }
10
```

Dimension is mandatory at time or creation. 92 Compile error

```
91
92 int[] x = new int[];
93 int[] x = new int[4];
94
95
```

Syntactically valid but an array of 0 size cant store any value

```
95
96
97 int[] x = new int[0];
98
```

Index 0 out of bounds for length 0

JVM will create a String array with three values A B and C. String args[] = of length 3

A screenshot of a Java development environment. The code editor window shows a Java class named 'Test' with a main method that prints 'A B C'. The command-line interface window below shows the execution of the program with arguments 'A B C'.

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5
6     }
7 }
8
9 java Test A B C
```

D:\durgaclasses>java Test
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test A B C
A B C
D:\durgaclasses>java Test A
A
D:\durgaclasses>java Test
B
D:\durgaclasses>

Below says an array will be created with size 0

A screenshot of a terminal window showing the execution of a Java program. The program prints the value of the 'args' array, which is shown to be of size 0.

```
1
2
3
4
5
6
7
8
9
10
11
12
13
D:\durgaclasses>java Test
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test A B C
A B C
D:\durgaclasses>java Test A
A
D:\durgaclasses>java Test
B
D:\durgaclasses>
```

Compiler will always look for type here int and the value is int so no compile time error. JVM are run time allocate memory based on size , Negative array size exception

A screenshot of an IDE showing a Java code editor and a terminal window. The code editor contains the following Java code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x = new int[-3];
6     }
7 }
```

The terminal window shows the output of running the code:

```
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
Exception in thread "main" java.lang.NegativeArraySizeException
        at Test.main(Test.java:5)
```

If we want utilities method for array we need `java.util`

`il.arrays`

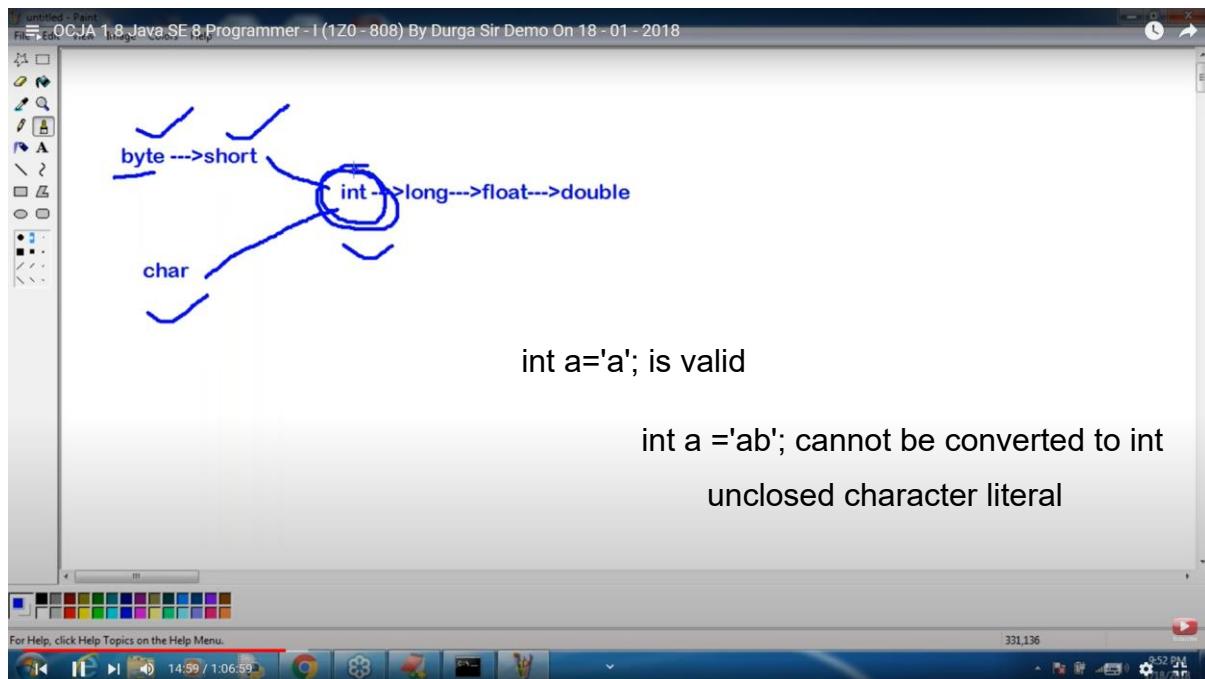
A screenshot of an IDE showing a Java code editor and a terminal window. The code editor contains the following Java code:

```
105 1.
106 2.
107
108 sir what is diff between array and java.util.Arrays?
109
110 Array
111 int[] x ={10,40,20,30};
112
113 Arrays.sort(x);
114 10,20,30,40
115
116
117
118
119
120
121
122
123
124
125
```

The terminal window shows the output of running the code:

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 17-01-2018
108 sir what is diff between array and java.util.Arrays?
109
110 Array
111 int[] x ={10,40,20,30};
112
113 Arrays.sort(x);
114 10,20,30,40
115
116
117
118
119
120
121
122
123
124
125
```

Byte short char are by default promoted to int



A -97 : size is 97

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x = new int['a'];
6         System.out.println(x.length);
7     }
8 }
```

Below are the size alloweds

```
35 int[] x = new int[0];
36 int[] x = new int[-5]; RE: NegativeArraySizeException
37
38
39 int
40
41
42 for array size:
43 -----
44 byte
45 short
46 char
47 int
48
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         short b = 10;
6         int[] x = new int[b];
7         System.out.println(x.length);
8     }
9 }
10
```

Compile error

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         long l = 10;
6         int[] x = new int[l];
7         System.out.println(x.length);
8     }
9 }
```

```
C:\Windows\system32\cmd.exe - OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 18 - 01 - 2018
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
10
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
97
D:\durgaclasses>javac Test.java
Test.java:6: error: incompatible types: possible lossy conversion from long to int
        int[] x = new int[1];
                           ^
1 error
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 18-01-2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         long l =10;
6         int[] x = new int[10.0];
7
8
9         System.out.println(x.length);
10    }
11 }
12
```

```
9
10 } } 1 error
11 } D:\durgaclasses>javac Test.java
12 Test.java:6: error: incompatible types: possible lossy conversion from float to
int
        int[] x = new int[10.0f];
                           ^
int[] x = new int[10.0f];
```

Datatype is double but the size is int

```
48
49
50 double[] d = new double[10];
51 |
```

```
49
50 double[] d = new double[10];
51
52 array type: double
53 array size : 10
54
```

Double casted to int is valid int accepted – Explicit type casting or implicit type case allowed

```
49
50 double[] d = new double[(int)10.5];
51
52 array type: double
53 array size : 10
54
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 18 - 01 - 2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         double[] d = new double[10.5];
6         System.out.println(d.length);
7     }
8 }
```

```
6             int[] x = new int[10.0f];
7
8 }           ^

1 error
D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: possible lossy conversion from double to
int
    double[] d = new double[10.5];
                           ^
1 error
```

Max allowed array size- max int value

```
56
57
58
59 byte
60 short
61 char
62 int
63 2147483647
```

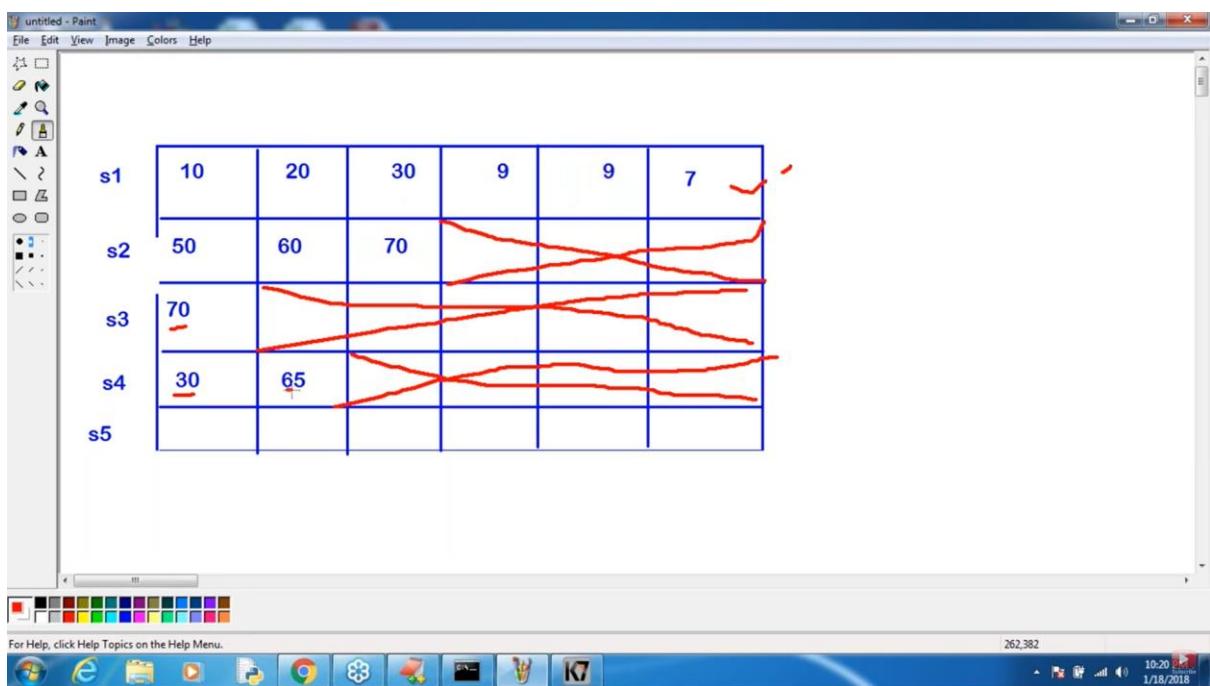
valid

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 18 - 01 - 2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         double[] d = new double[2147483647];
6     }
7 }
```

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 18-01-2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         double[] d = new double[2147483647];
6         System.out.println("Hello Listen Dont sleep");
7     }
8 }
9
10 // 2147483647*8 bytes: OutOfMemoryError
```

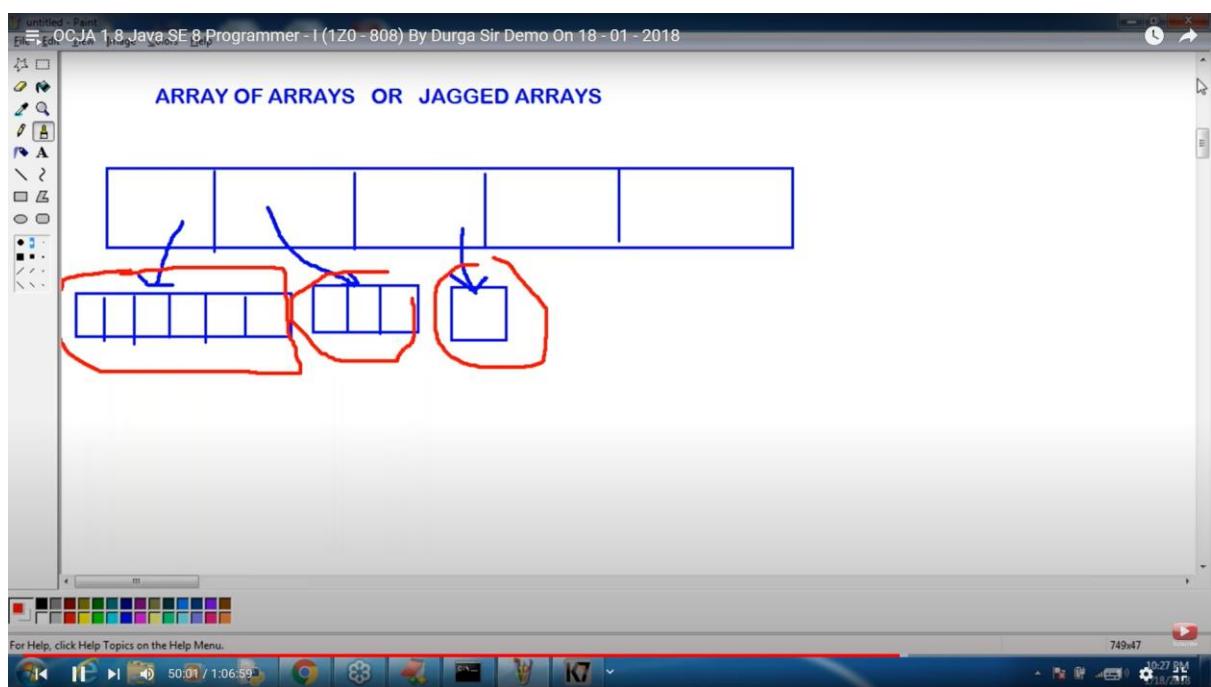
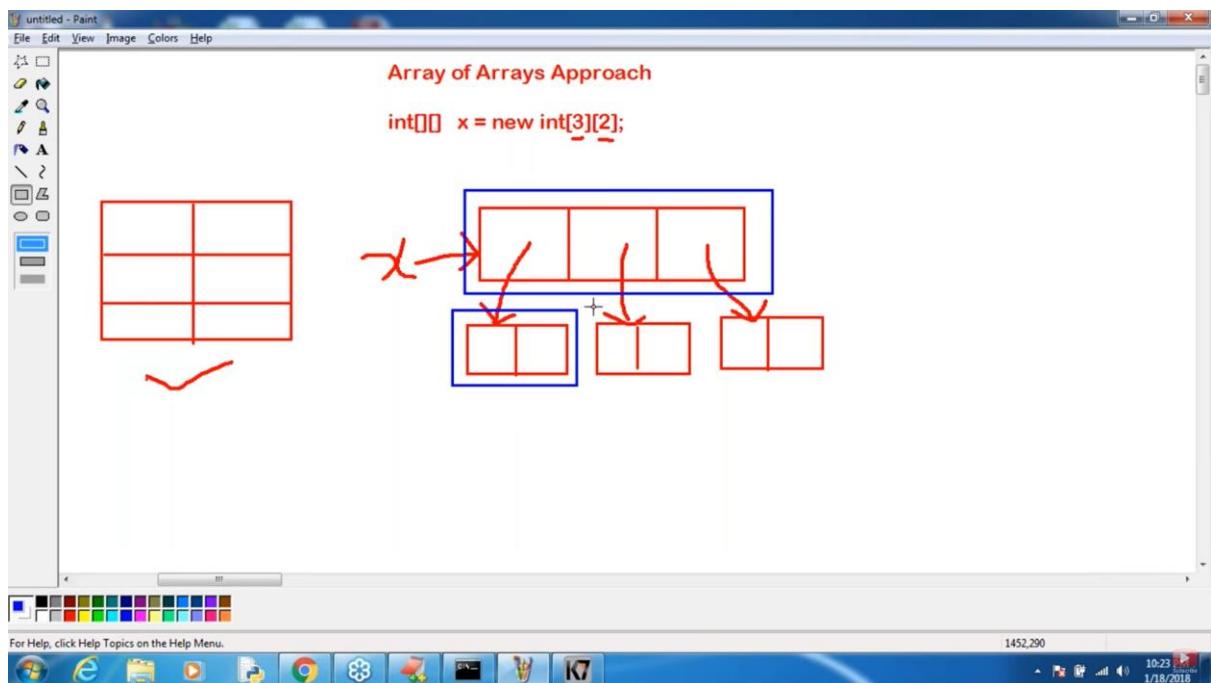
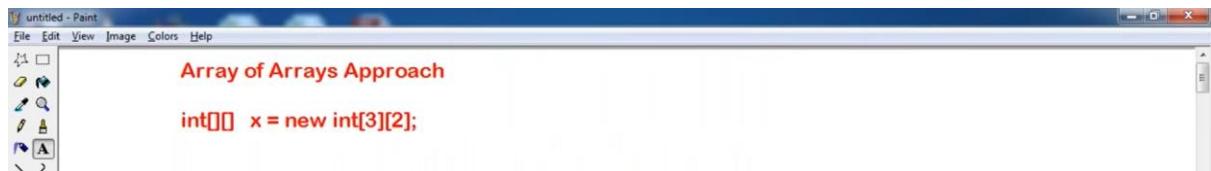
```
1 error
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
Exception in thread "main" java.lang.OutOfMemoryError: Requested array size exceeds VM limit
        at Test.main(Test.java:5)
```

In Java 2D are not implemented as Matrix Style due to memory wastage

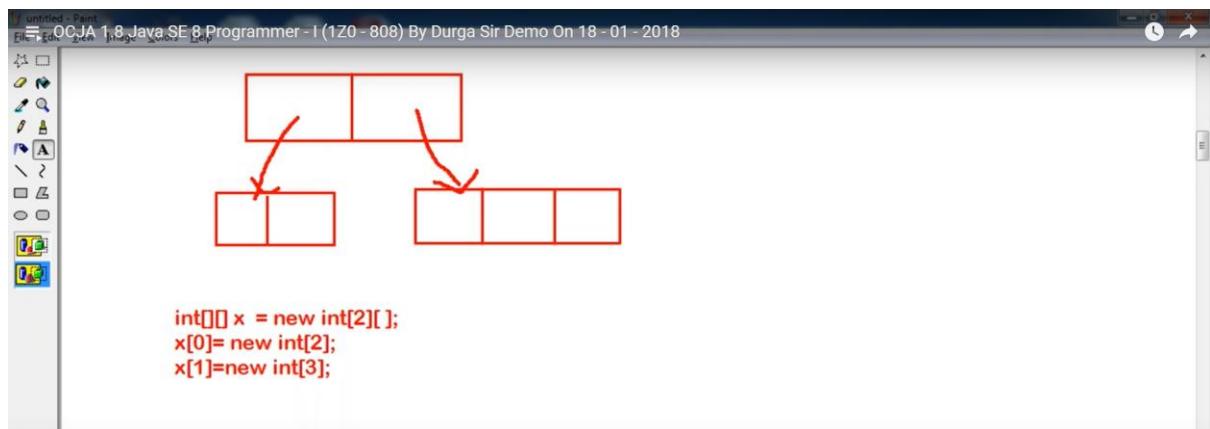
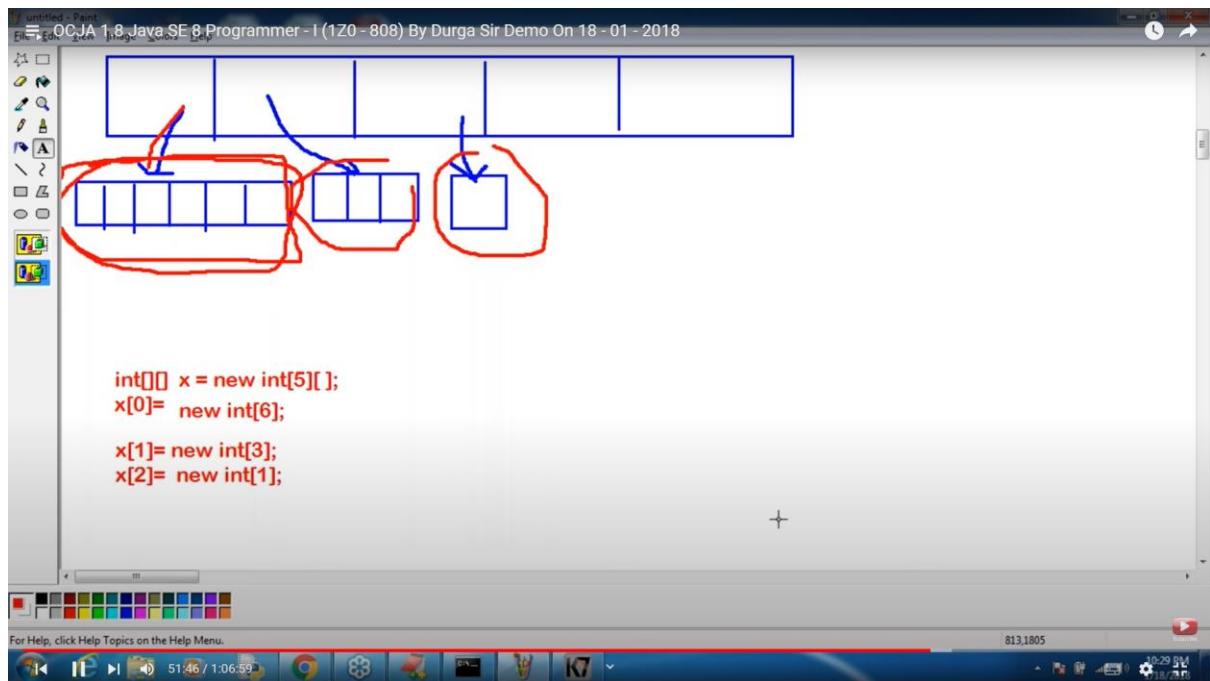


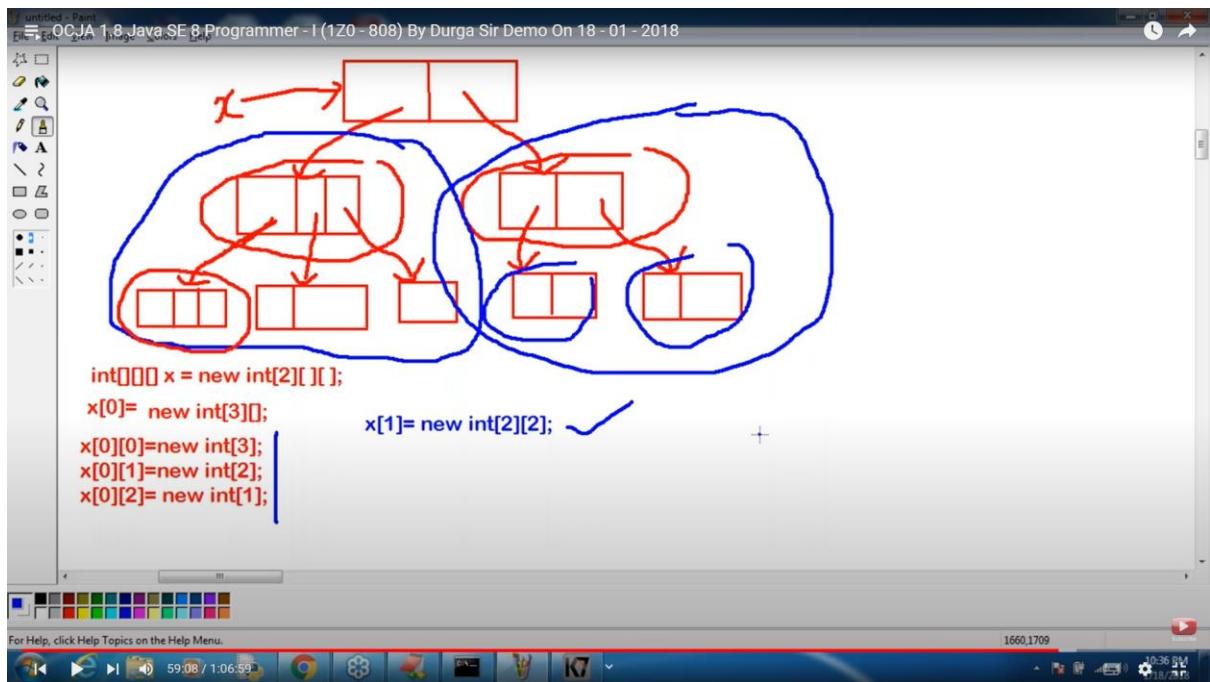
Arrays of Arrays Approach is followed

1



In java its ok if we only specify the base size not all dimension





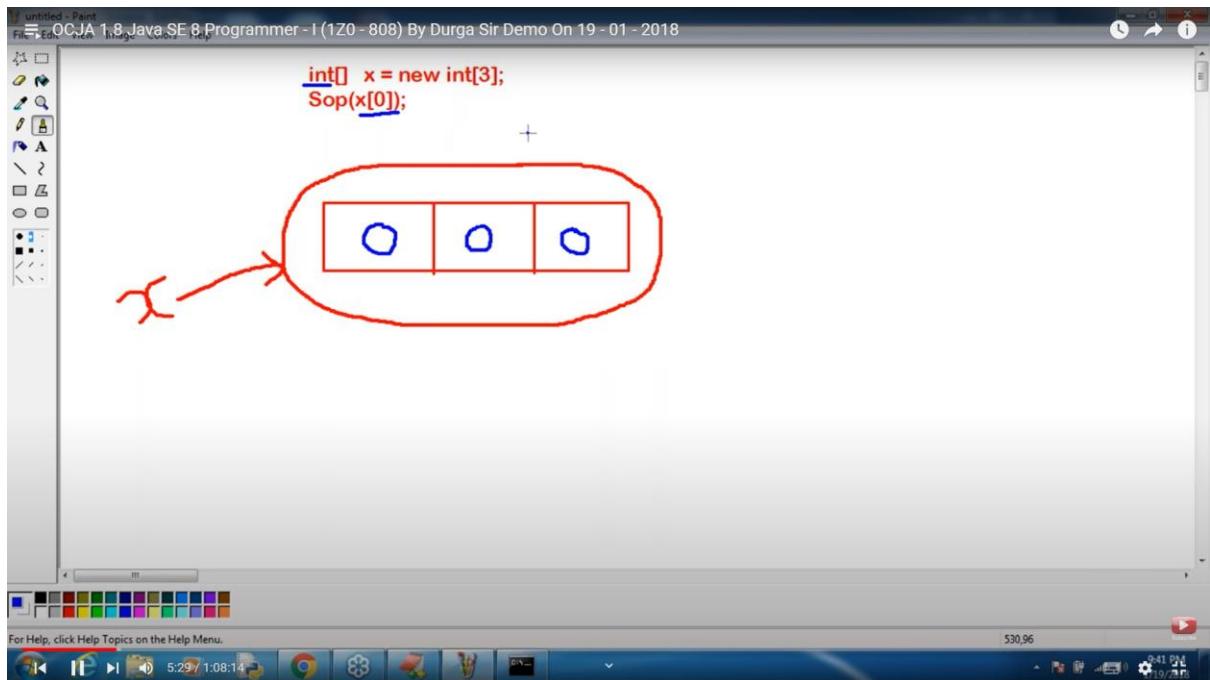
Line 8 error

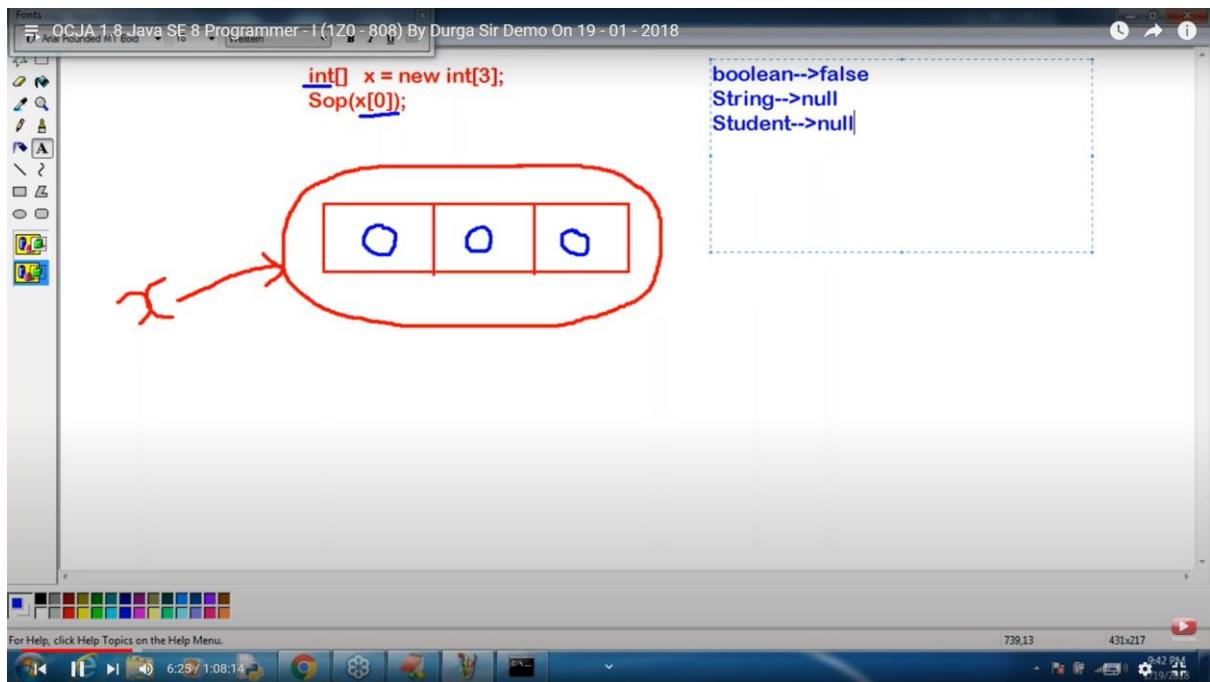
```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x = new int[];
6         int[][] x = new int[4][3];
7         int[][] x = new int[4][];
8         int[][] x = new int[3];
9
10    }
11 }
```

Without knowing 3rd D how would you specify 4th and without knowing 6th how would you specify 7

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 18 - 01 - 2018  
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         int[][][] x = new int[2][4][6];  
6     }  
7 }
```

Every array element is of default value.





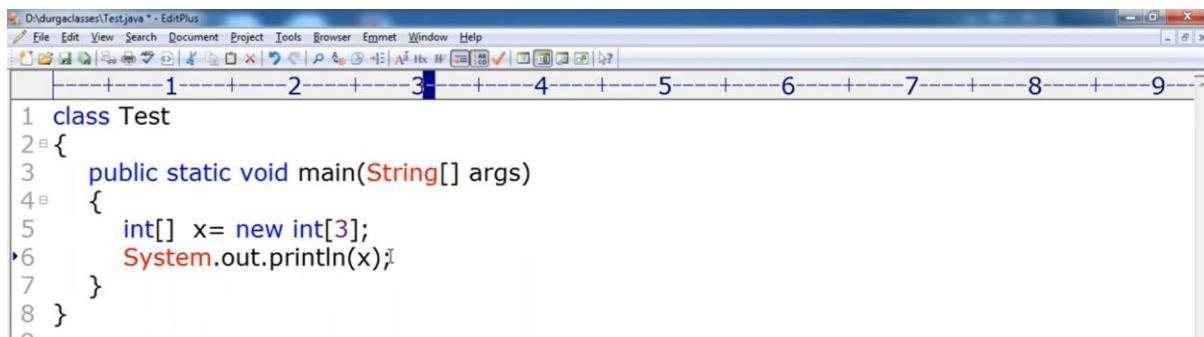
RE – Array Index OOB Exception

A screenshot of an IDE window titled "D:\durgaclasses\Test.java - EditPlus". The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x= new int[3];
6         System.out.println(x[10]);
7     }
8 }
```

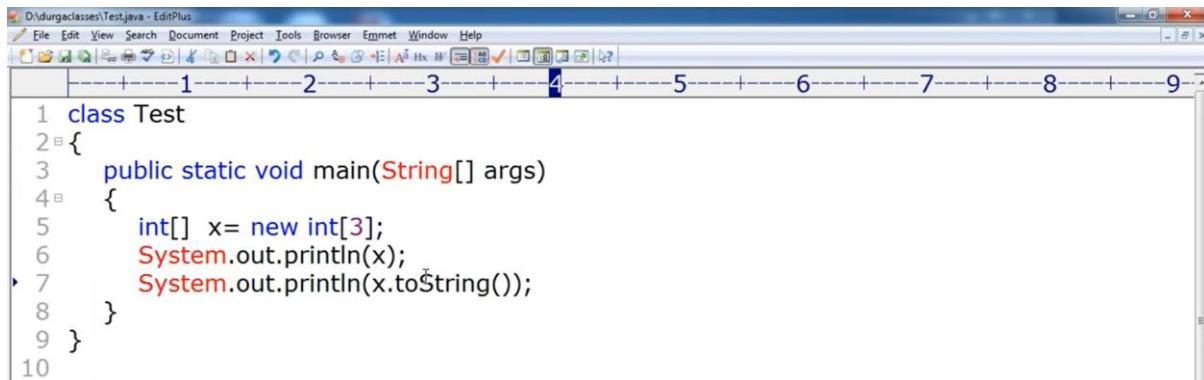
The line `System.out.println(x[10]);` is highlighted in red, indicating an error. The code editor has a ruler at the top with numbered tick marks from 1 to 9. The cursor is positioned at the end of the line `x[10]`.

When we print or assign ref variable the `toString` method will be called

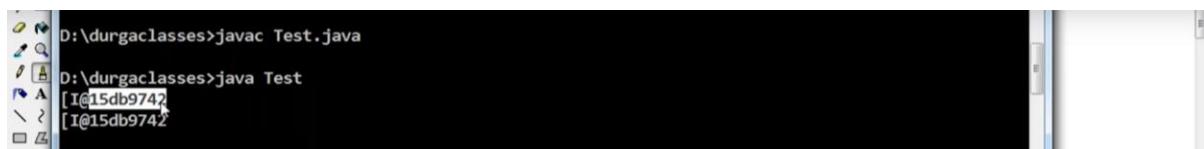
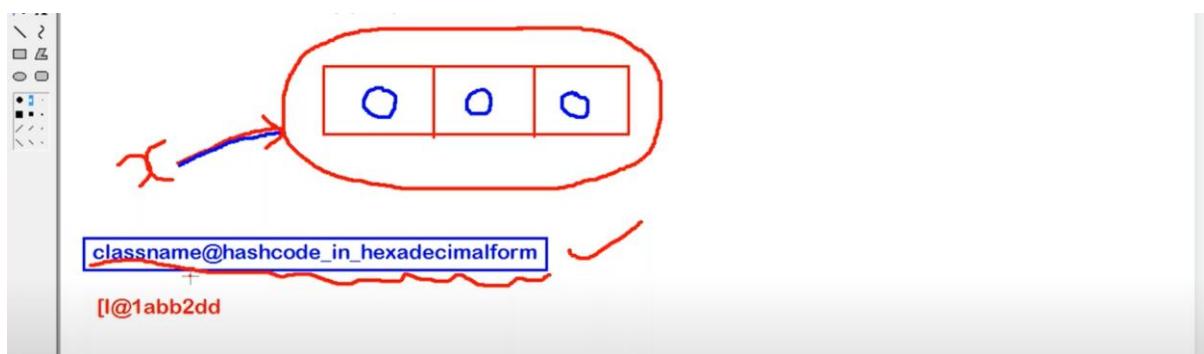


```
D:\durgaclasses>Test.java - EditPlus
File Edit View Search Document Project Tools Browser Emet Window Help
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x= new int[3];
6         System.out.println(x);
7     }
8 }
```

X will be converted to x.`toString()`. The default implementation of `toString` method is `classname@hascode_in_hexadecimal`. The hashcode will change from machine to machine

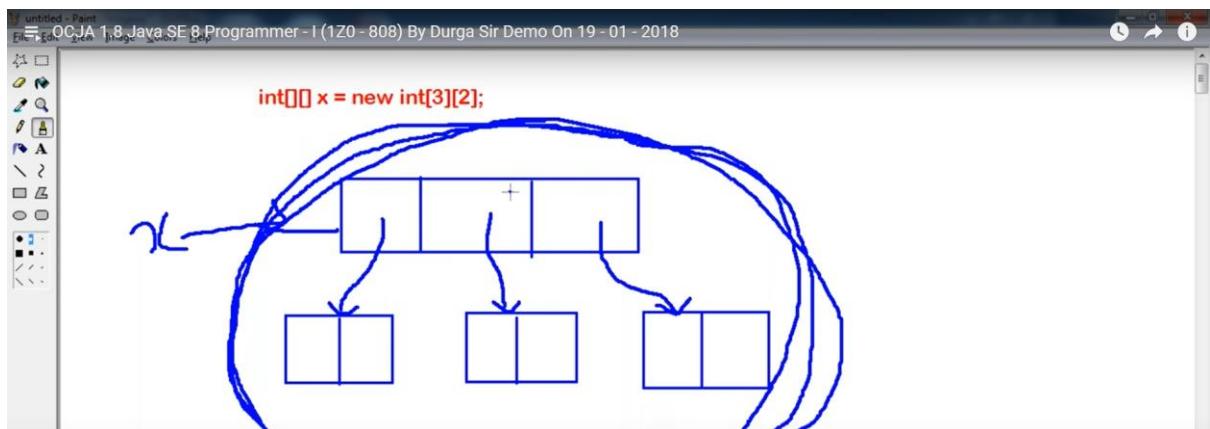


```
D:\durgaclasses>Test.java - EditPlus
File Edit View Search Document Project Tools Browser Emet Window Help
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x= new int[3];
6         System.out.println(x);
7         System.out.println(x.toString());
8     }
9 }
10
```



```
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
[I@15db9742
[I@15db9742
```

```
D:\durgaclasses\Text.java -> EditPlus  
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 19 - 01 - 2018  
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         int[][] x = new int[3][2];  
6         System.out.println(x); //[[I@hashcode  
7         System.out.println(x[0]);  
8         System.out.println(x[0][0]);  
9     }  
10 }  
11
```



```
D:\durgaclasses\Text.java -> EditPlus  
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 19 - 01 - 2018  
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         int[][] x = new int[3][2];  
6         System.out.println(x); //[[I@hashcode  
7         System.out.println(x[0]); //[[I@hashcode  
8         System.out.println(x[0][0]); //0  
9     }  
10 }  
11
```

```

1 Array Declaration
2 Array Creation
3 Array Initialization
4
5 java.lang package==>Object class methods hashCode()
6 int[] x = new int[3];
7 x[0]=100;
8 x[1]=20;
9
10
11
12
13
14
15
16
17
18
19
20
21 int[] x = {10,20,30};
22 char[] ch={'a','e','i','o','u'};
23 String[] s={"Katrina", "Kareena", "Sunny", "mallika"};
24
25

```

Below is compile time error if you want to use shortcut the initialization and declartion should be in single line.

```

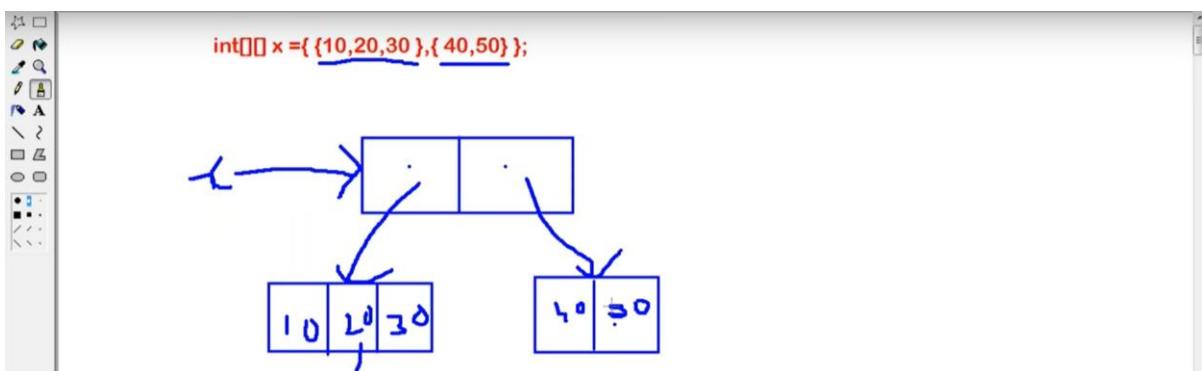
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x;
6         x={10,20,30};
7         System.out.println(x[0]);
8     }
9 }

27
28 Array of arrays concept
29
30 int[][] x ={ {10,20,30 },{ 40,50 } };
31 sop(x[0][1]);//20
32 sop(x[1][1]);//50
33 sop(x[2][0]);//AIOOBE
34

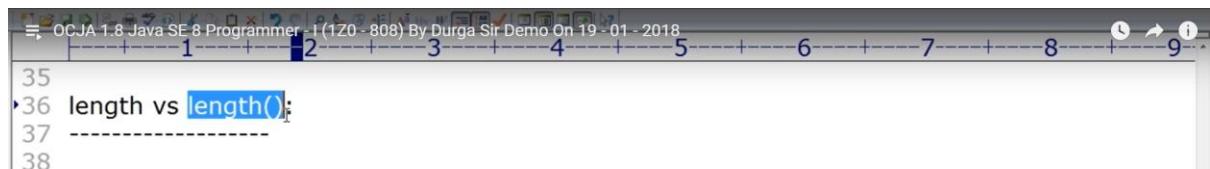
```

int x[];
x= new int[] is valid

int x[];
x={10,25,1,5} is invalid



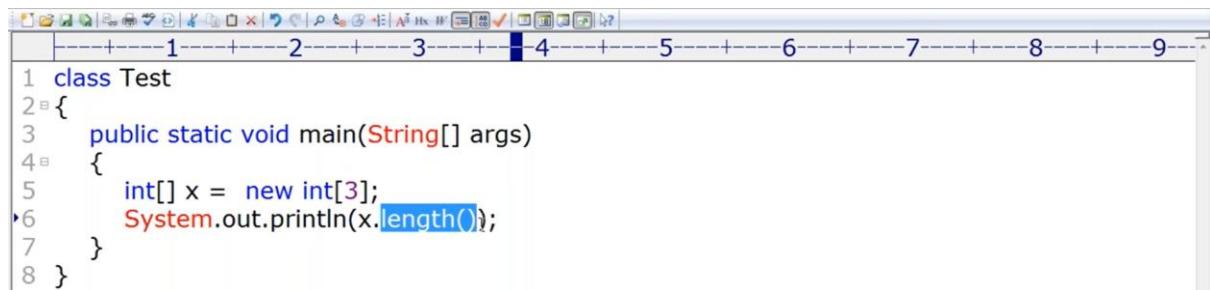
Length method is applicable for array and Length() method is for string length



OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 19 - 01 - 2018

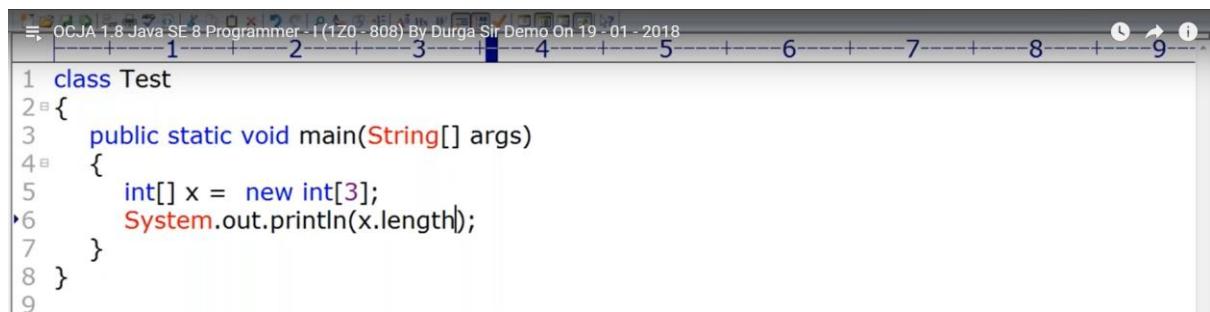
```
35
36 length vs length();
37 -----
38
```

length() cannot be applied to array – compile time error



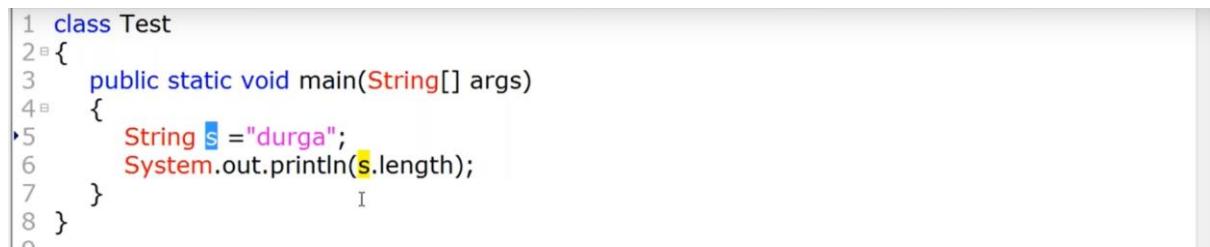
```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x = new int[3];
6         System.out.println(x.length());
7     }
8 }
```

Below is correct



```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x = new int[3];
6         System.out.println(x.length());
7     }
8 }
```

Below is wrong length variable is not applicable for string



```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         String s ="durga";
6         System.out.println(s.length());
7     }
8 }
```

```

1 class Test
2 {
3     public static void main(String[] args)
4     {
5         String[] s = {"A", "AA", "AAA"};
6         System.out.println(s.length);
7         //System.out.println(s.length());
8         //System.out.println(s[0].length);
9         System.out.println(s[0].length());
10    }
11 }

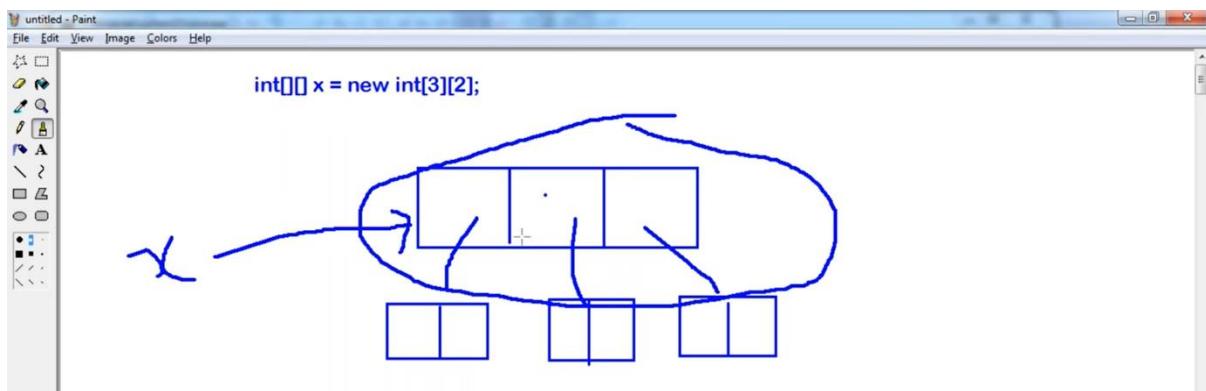
```

```

OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 19 - 01 - 2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[][] x = new int[3][2];
6         System.out.println(x.length);
7     }
8 }

```

Length - 3



`x[0].length = 2`

```

OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 19 - 01 - 2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[][] x = new int[3][2];
6         System.out.println(x.length);
7         System.out.println(x[0].length);
8     }
9 }

```

```
1 Anonymous Arrays:  
2 -----  
3 int[] x={10,20,30};  
4  
5 Array just for instant use  
6 new int[] {10,20,30}  
7  
8 new int[][] {{10,20},{30,40,50}}  
9
```

```
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         //10,20,30,40  
6         sum(new int[] {10,20,30,40});  
7     }  
8     public static void sum(int[] x)  
9     {  
10        int total=0;  
11        for (int x1:x)  
12        {  
13            total=total+x1;  
14        }  
15        System.out.println("The Sum:"+total);  
16    }  
17 }  
18 }  
19 }
```

For anonymous array you cant specify size

```
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         //10,20,30,40  
6         sum(new int[4] {10,20,30,40});  
7     }  
8 }
```

```
C:\Windows\system32\cmd.exe  
D:\durgaclasses>javac Test.java  
Test.java:6: error: ')' expected  
      sum(new int[4] {10,20,30,40});  
                           ^  
Test.java:6: error: not a statement  
      sum(new int[4] {10,20,30,40});  
                           ^  
Test.java:6: error: ')' expected  
      sum(new int[4] {10,20,30,40});  
                           ^
```

You can specify size or not .you should either specify size or elements if you specify both. U will get compile time error. Here you are creating an annoymous array and assigning it to x array.

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x=new int[]{10,20,30,40};
6         System.out.println(x[0]);
7     }
8
9 }
10
```

A screenshot of a Java code editor window. The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x=new int[]{10,20,30,40};
6         System.out.println(x[0]);
7     }
8
9 }
10
```

The line `int[] x=new int[]{10,20,30,40};` is highlighted in blue, indicating a syntax error. The editor has a ruler at the top with numbered tick marks from 1 to 9.

If you specify both size and elements it will throw error

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int[] x=new int[4]{10,20,30,40};
6         System.out.println(x[0]);
7     }
8
9 }
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 19 - 01 - 2018
Test.java:16: error: class, interface, or enum expected
        }
        ^
Test.java:18: error: class, interface, or enum expected
        ^
8 errors
D:\durgaclasses>javac Test.java
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
10
D:\durgaclasses>javac Test.java
Test.java:5: error: ';' expected
    int[] x= new int[4]{10,20,30,40};
               ^
5 seconds
```

The code in the editor is identical to the one above, but the declaration `int[] x=new int[4]{10,20,30,40};` is now highlighted in red, indicating a different type of error. The terminal window shows several compilation errors, with the first two being:

```
Test.java:16: error: class, interface, or enum expected
        }
        ^
Test.java:18: error: class, interface, or enum expected
        ^
8 errors
```

These errors are caused by specifying both the size and elements in the array declaration. The terminal also shows the command `javac Test.java` being run multiple times and failing.

Here `n[0].length = 3`

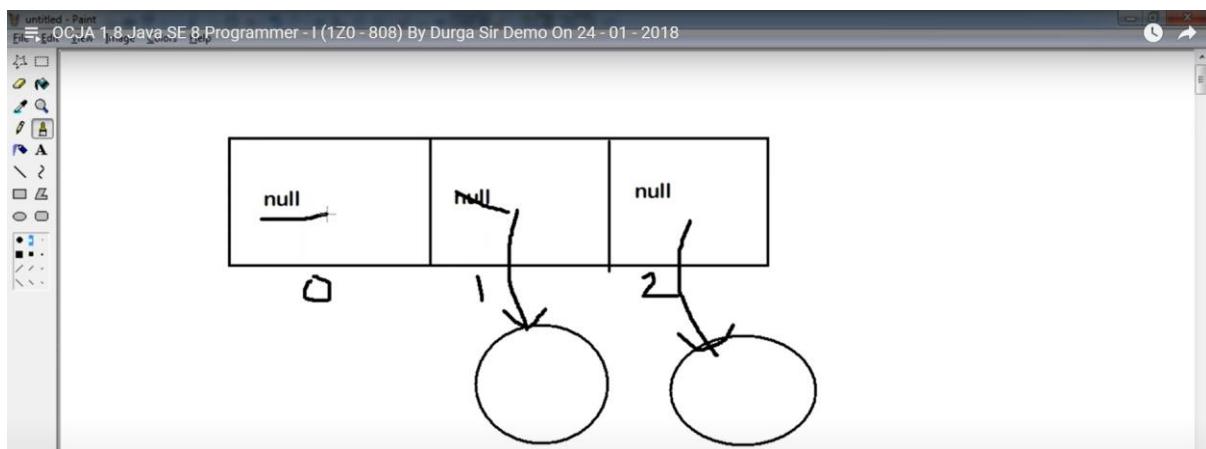
For `j=0 j>3` never going to happen. So default value will be saved. Inner loop not gonna get executed
.Default value would be displayed which is 0

The screenshot shows a Java code editor with the following code:

```
191 {  
192     public static void main(String[] args)  
193     {  
194         int[][] n = new int[1][3];  
195         for(int i =0; i< n.length; i++)  
196         {  
197             for (int j=0;j>n[i].length ;j++ )  
198             {  
199                 num[i][j]=10;  
200             }  
201         }  
202     }  
203 }  
204  
205 Which option represents the state of the array after successful completion of outer for loop?  
206  
207 A)  
208     n[0][0]=10;  
209     n[0][1]=10;  
210     n[0][2]=10;  
211 R)
```

The code editor interface includes a toolbar at the top, a status bar at the bottom, and a scroll bar on the right side of the code area.

```
1 class Student
2 {
3     String name;
4     public Student(String name)
5     {
6         this.name=name;
7     }
8 }
9 public class Test
10 {
11     public static void main(String[] args)
12     {
13         Student[] students = new Student[3];
14         students[1]= new Student("Durga");
15         students[2]= new Student("Ravi");
16         for(Student s : students)
17         {
18             System.out.println(s.name);
19         }
20     }
}
For Help, press F1
```



Null – calling name function on null. Null pointer exception. If we call any operation on null its gonna throw null pointer exception

```
11 D:\durgaclasses>java Test
12 Exception in thread "main" java.lang.NullPointerException
13     at Test.main(Test.java:18)
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018
1 -----
2
3
4 primitive variable
5 reference variables
6
7
8 local variables
9 static variables
10 instance variables
11 I
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018
1 -----
2
3
4 1. primitive variables:
5 -----
6 int x=10;
7 char ch ='a';
8 double d =10.5;
9
10 2. Reference variables:
11 -----
12 String s="Durga";
13 Student s1=new Student("Durga",101);
14
15
16
17
18
19
20
21
22
23
24
```

```
26 Division-2:
27 -----
28 Based on behaviour and position of declaration
29
30 1. instance variables
31 2. static variables
32 3. local variables
33
```

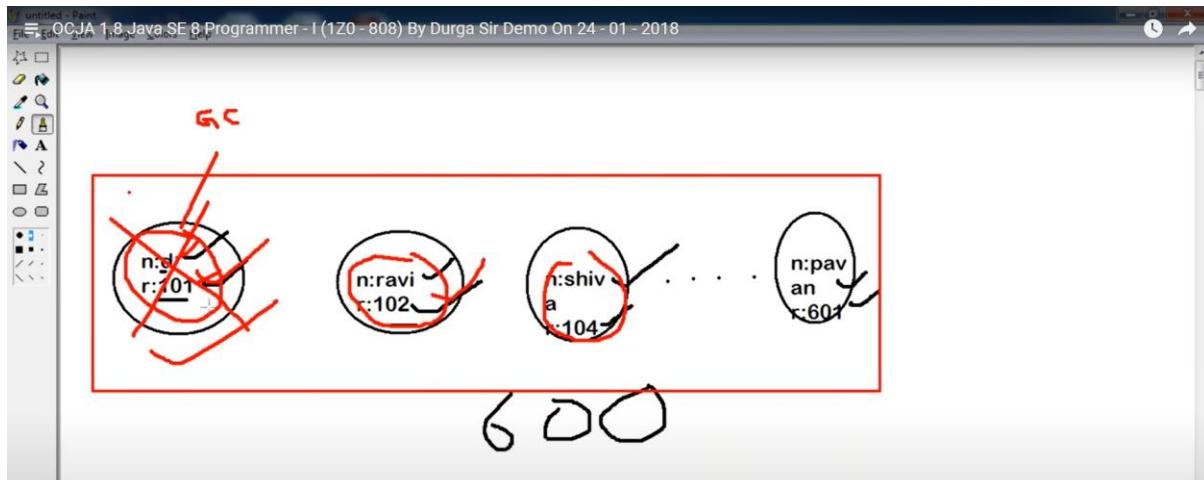
If the value of the variable is going to change for instance to instance its called instance variable.

Here name and roll no changes for every instance of Student. **Instance variables should be declared inside the class not inside methods or constructors or blocks**

Instance variables are created at the time of object creation. When the object is destroyed then instance variable will be destroyed

```
1 class Student
2 {
3     String name;
4     int rollno;
5 }
6
7 instance variables or object level variables...I
8
9
10
11 Student s1 = new Student("Durga",101);
12 Student s2 = new Student("Ravi",102);
13 Student s3 = new Student("Shiva",103);
14 ....600 students
```

For every object separate copy will be created



From static method we would not be able to access instance variable. The instance variable is part of object not class. Without object we cant access it from static method. Non static variable cannot be accessed from a static context.

```
1 public class Test
2 {
3     int x=10;
4     public static void main(String[] args)
5     {
6         System.out.println(x);
7     }
8 }
```

Instance variable can be accessed from instance area. This is right but main method will not provide any output

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018  
1 public class Test  
2 {  
3     int x=10;  
4     public static void main(String[] args)  
5     {  
6     }  
7     public void m1()  
8     {  
9         System.out.println(x);  
10    }  
11 }  
12 }
```

Static method cannot access instance method or variable directly.

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018  
1 public class Test  
2 {  
3     int x=10;           I  
4     public static void main(String[] args)  
5     {  
6         m1();|  
7     }  
8     public void m1()  
9     {  
10        System.out.println(x);  
11    }  
12 }
```

```
C:\Windows\system32\cmd.exe  
D:\durgaclasses>javac Test.java  
D:\durgaclasses>java Test  
10  
D:\durgaclasses>javac Test.java  
D:\durgaclasses>java Test  
D:\durgaclasses>javac Test.java  
Test.java:6: error: non-static method m1() cannot be referenced from a static context  
                ^  
1 error
```

To access it you need create an object and access it. We can access instance variable from static area using object ref

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 24 - 01 - 2018  
1 public class Test  
2 {  
3     int x=10;  
4     public static void main(String[] args)  
5     {  
6         Test t = new Test();  
7         t.m1();  
8     }  
9     public void m1()  
10    {  
11        System.out.println(x);  
12    }  
13 }
```

For instance variable we don't need to initialize. JVM will provide the default value explicitly. If we don't want default the we can specify default variable

```
1 public class Test  
2 {  
3     int x;  
4     public static void main(String[] args)  
5     {  
6         Test t = new Test();  
7         System.out.println(t.x);  
8     }  
9 }
```

Types of Variables.pdf - Adobe Reader
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018

Based on the behavior and position of declaration all variables are divided into the following 3 types.

- Instance variables
- Static variables
- Local variables

Instance variables:

- If the value of a variable is varied from object to object such type of variables are called instance variables.
- For every object a separate copy of instance variables will be created.
- **Instance variables will be created at the time of object creation and destroyed at the time of object destruction hence the scope of instance variables is exactly same as scope of objects.**
- Instance variables will be stored on the heap as the part of object.
- Instance variables should be declared with in the class directly but outside of any method or block or constructor.
- Instance variables can be accessed directly from Instance area. But cannot be accessed directly from static area.
- But by using object reference we can access instance variables from static area.

Types of Variables.pdf - Adobe Reader
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 24 - 01 - 2018

Example:

```
1) class Test
2) {
3)     int i=10;
4)     public static void main(String[] args)
5)     {
6)         //System.out.println(i);
7)         //C:non-static variable i cannot be referenced from a static context(invalid)
8)         Test t=new Test();
9)         System.out.println(t.i); //10(valid)
10)        t.methodOne();
11)    }
12)    public void methodOne()
13)    {
14)        System.out.println(i); //10(valid)
15)    }
16) }
```

- For the instance variables it is not required to perform initialization JVM will always provide default values.

Example:

Tools | Fill & Sign | Comment | Sign In

Export PDF
Adobe ExportPDF
Convert PDF files to Word or Excel online.
Select PDF File:
Types of Variables.pdf
1 file / 977 KB
Convert To:
Microsoft Word (*.docx)
Recognize Text in English(U.S.)
Change
Convert

Create PDF
Edit PDF
Combine PDF
Send Files
Store Files

The screenshot shows a PDF document titled "Types of Variables.pdf" open in Adobe Reader. The main content area displays the following Java code:

```

1) class Test
2) {
3)     boolean b;
4)     public static void main(String[] args)
5)     {
6)         Test t=new Test();
7)         System.out.println(t.b); //false
8)     }
9)

```

Below the code, there is a bulleted list:

- For the instance variables it is not required to perform initialization JVM will always provide default values.

Example:

Another bulleted list follows:

- Instance variables also known as object level variables or attributes.

The right side of the interface features a sidebar with various tools and options, including "Export PDF" and "Selected PDF File".

If its static these are class level variable (information). For all 600 object only 1 copy of cname will be created at class level and can be accessed by all the objects. To save memory.. if the value does not change for object to object then that variable can be defined as static

The screenshot shows a code editor window with the following Java code:

```

1 class Student
2 {
3     String name;
4     int rollno;
5     static String cname;
6
7 }
8
9 instance variables or object level variables...
10
11
12
13 Student s1 = new Student("Durga",101);
14 Student s2 = new Student("Ravi",102);
15 Student s3 = new Student("Shiva",103);
16 ....600 students

```

Object level variables – Instance Variable

Class level variables – Static

For all the object if the variable value is unique – instance variable

Static variable are created when the class is loading. And destroyed at time of class unloading

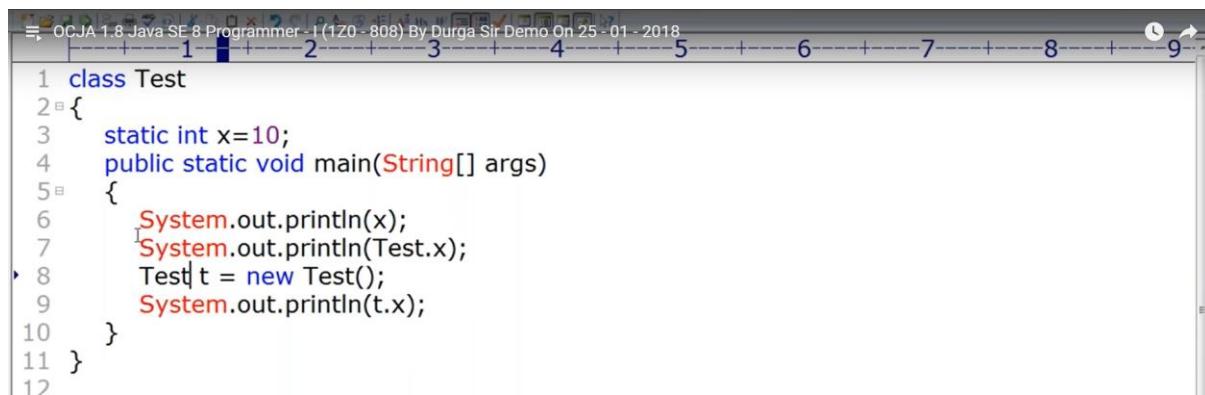


When ever we execute the class file.

Static variable will be created when the Test.class is loaded and destroyed when Test.class is unloaded

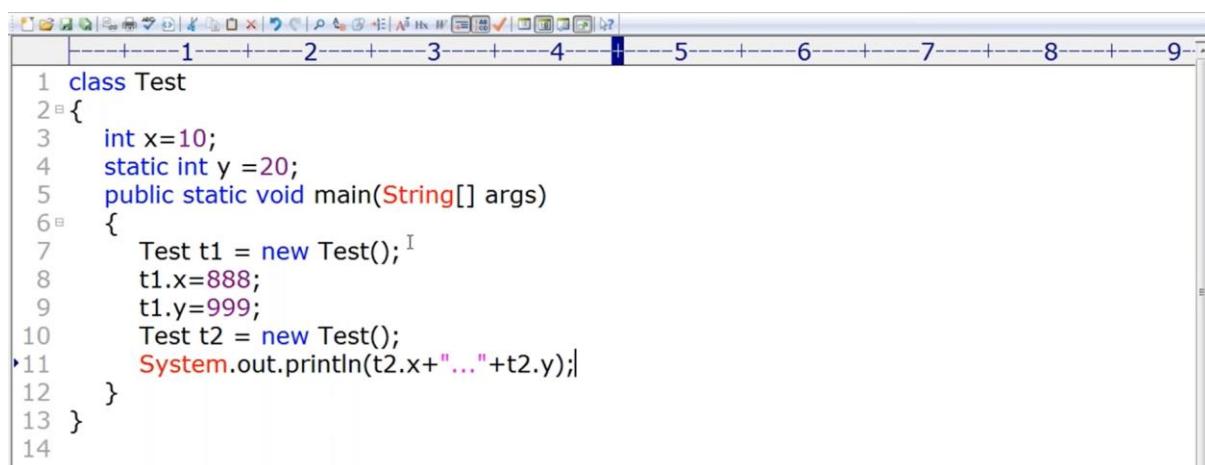
```
43  
44  
45 D:\durgaclasses>java Test I  
46  
47 1. JVM will be started  
48 2. Create and Start Main Thread by JVM  
49 3. Main Thread will search for Test.class==>(NoClassDefFoundError)  
50 4. Main Thread will load Test.class  
51 5. Execute main method  
52 6. unload Test.class file  
53 7. Terminate Main Thread  
54 8. JVM will be shutdown  
``
```

Static variable can be accessed via object ref but not advisable. Line 9

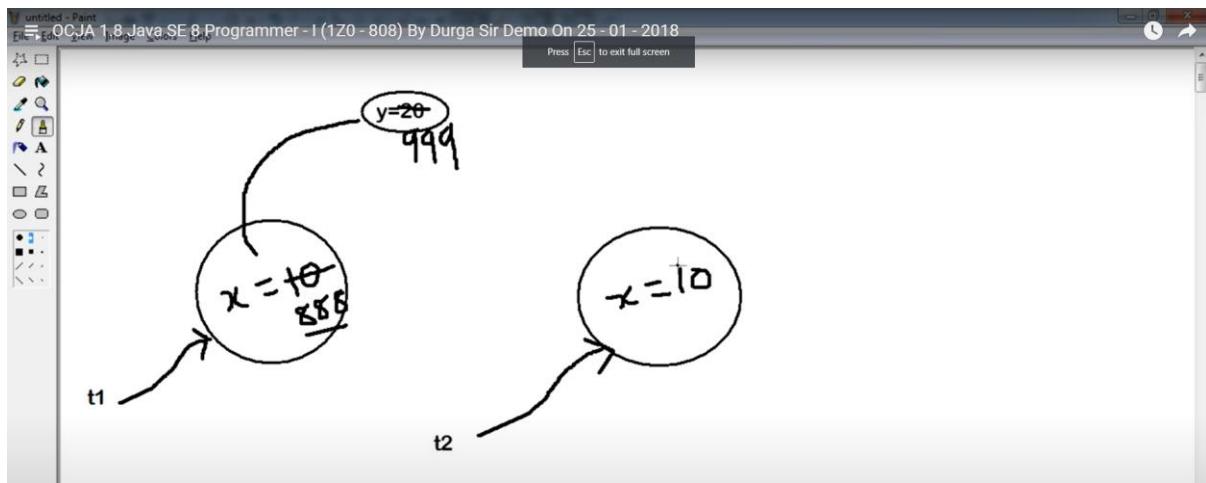


```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 25 - 01 - 2018  
1 class Test  
2 {  
3     static int x=10;  
4     public static void main(String[] args)  
5     {  
6         System.out.println(x);  
7         System.out.println(Test.x);  
8         Test t = new Test();  
9         System.out.println(t.x);  
10    }  
11 }  
12 
```

10 ...999. For all object ref a sep copy of instance variable will be there a common copy of static variable will be there. Any change to static variable will be ref across all objects ref



```
1 class Test  
2 {  
3     int x=10;  
4     static int y =20;  
5     public static void main(String[] args)  
6     {  
7         Test t1 = new Test();  
8         t1.x=888;  
9         t1.y=999;  
10        Test t2 = new Test();  
11        System.out.println(t2.x+"..."+t2.y);  
12    }  
13 }  
14 
```



20 and null pointer exception. Static needs no object creation

```

1 class Test
2 {
3     int x=10;
4     static int y =20;
5     public static void main(String[] args)
6     {
7         Test t = null;
8         System.out.println(t.y);
9         System.out.println(t.x);
10    }
11 }
12 
```

```

7 D:\durgaclasses>javac Test.java
8
9 D:\durgaclasses>java Test
10
11 Exception in thread "main" java.lang.NullPointerException
12     at Test.main(Test.java:9)
13 
```

```

19 -----
20 3. local variables:
21 -----
22 a variable inside method|block|constructor local variables
23 
```

Local variable will be created when the method block or const is executed and will be destroyed upon completion.

```

1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x=10;
6     }
7 } 
```

x-Scope within main and y- scope within try block

The screenshot shows a Java code editor with the following code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x =10;
6         try
7         {
8             int y=20;
9         }
10        catch ()
11        {
12        }
13    }
14 }
```

Variables 'x' and 'y' are highlighted in blue, indicating their scope. 'x' is accessible throughout the entire main method, while 'y' is only accessible within the try block.

Compile time error- J is accessed outside for loop

The screenshot shows a Java code editor with the following code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int i =0;
6         for(int j=0;j<3;j++)
7         {
8             i=i+j;
9         }
10        System.out.println(i+".."+j);
11    }
12 }
```

A red squiggly underline is under the variable 'j' in the println statement, indicating a compilation error.

The screenshot shows a command-line window with the following output:

```
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
20
Exception in thread "main" java.lang.NullPointerException
at Test.main(Test.java:9)

D:\durgaclasses>javac Test.java
Test.java:10: error: cannot find symbol
          System.out.println(i+".."+j);
                           ^
symbol:   variable j
location: class Test
1 error
```

2 compilation error x- is local to try and cant be accessed from catch and main

A screenshot of an IDE showing a Java file named 'Test.java'. The code contains a try-catch block where the variable 'x' is defined in the try block and used in the catch block. This results in a compilation error. The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             int x = Integer.parseInt("ten");
8         }
9         catch (NumberFormatException e)
10        {
11             x=10;
12         }
13         System.out.println(x);
14     }
15 }
```

For local variable JVM will not provide temp value , This is very local and temp.

A screenshot of an IDE showing a Java file named 'Test.java'. It declares a local variable 'x' without initializing it. The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x;
6         System.out.println("Hello dont sleep");
7     }
8 }
```

Before using local variable we nee to initialize , if we are not using we don't hv to initialize

A screenshot of a terminal window showing the execution of the 'Test.java' program. The output shows the string 'Hello dont sleep'.

```
5 D:\durgaclasses>javac Test.java
6 D:\durgaclasses>java Test
7 Hello dont sleep
```

Below compile time local variable should be initialized before using

A screenshot of an IDE showing a Java file named 'Test.java'. It declares a local variable 'x' without initializing it. This results in a compilation error. The code is as follows:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x;
6         System.out.println(x);
7     }
8 }
```

The if will be executed only when there are command line args . Compile error. It can't assume that run time args will be passed

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x;
6         if(args.length >0)
7         {
8             x=10;
9         }
10        System.out.println(x);
11    }
12 }
13
```

The screenshot shows a Java code editor window. The code is identical to the one above, but it contains a syntax error at line 10: "System.out.println(x);". The word "System" is colored red, indicating a compilation error. The status bar at the bottom of the editor window displays the message "Press Esc to exit full screen".

Not recommended to initialize local variable inside logical blocks. Good to initialize beginning with default value.

```
8 D:\durgaclasses>javac Test.java
9 Test.java:10: error: variable x might not have been initialized
10             System.out.println(x);
```

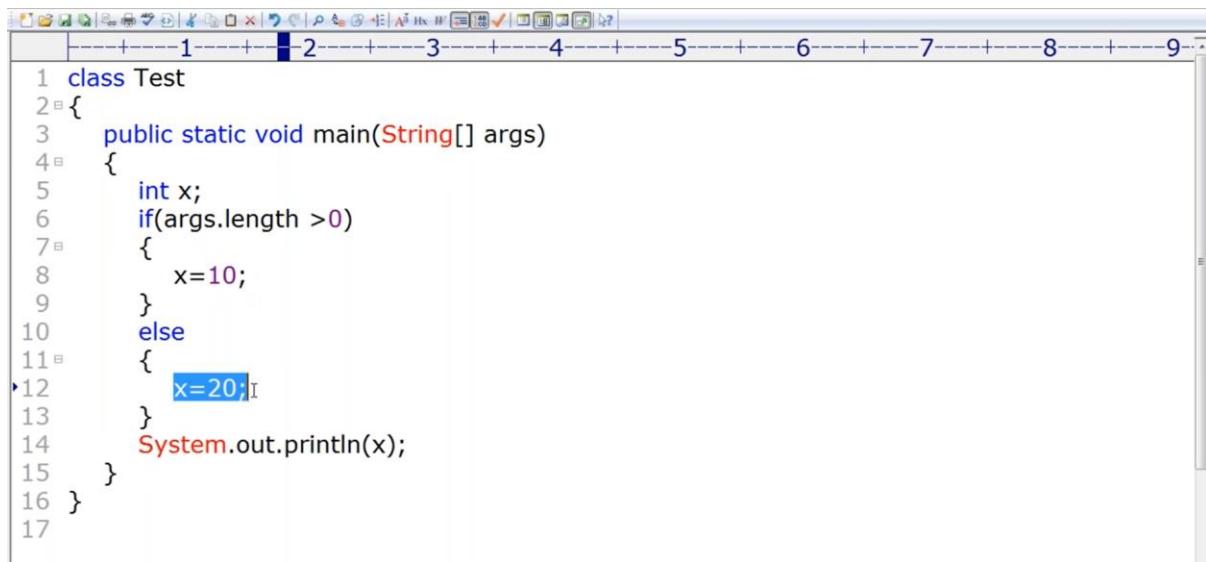
The screenshot shows a terminal window with the command "javac Test.java" entered. The output shows a single error message at line 10: "Test.java:10: error: variable x might not have been initialized". This is the same error as the one in the code editor, indicating that the variable "x" has not been initialized before it is used.

Below pass, default value is passed

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x=0; I
6         if(args.length >0)
7         {
8             x=10;
9         }
10        System.out.println(x);
11    }
12 }
```

The screenshot shows a Java code editor window with the same code as before, but it has been corrected. At line 5, the initialization "int x;" is followed by a semicolon and a comment "I" (Intentionally). This tells the compiler that the variable "x" is initialized to zero at the start of the block. The code now compiles successfully without any errors.

X value is always initialized . so compiler knows that .



```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         int x;
6         if(args.length >0)
7         {
8             x=10;
9         }
10        else
11        {
12            x=20;I
13        }
14        System.out.println(x);
15    }
16 }
17
```

Local variables before using should always be initialized

A top-level class cannot be defined as static but inner level class can be defined as static. Variables and methods can be declared as static.

The variable and method are class level no object instance is needed



```
1 class Test
2 {
3     static int x=10;
4     public static void m1()
5     {
6     }
7 }
```

Any person is added to access this class and its members

```
public class Test{  
final class Test{ - cant create child class  
static class Test{ - Static means only 1 copy is maintained – when we define class static it does not provide any value or meaning
```

the below is an inner class. If want to create an object of inner class we need to create an object of outer class.

```
\OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 26 - 01 - 2018
1 class Test
2 {
3     class Inner
4     {
5     }
6     static int x=10;
7     public static void m1()
8     {
9     }
10}
11 Test.Inner i=new Test().newInner();
```

If we define the inner class as static we can create an object of inner class without creating an object of outer class

```
\OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 26 - 01 - 2018
1 class Test
2 {
3     static class Inner
4     {
5     }
6     static int x=10;
7     public static void m1()
8     {
9     }
10}
11 Test.Inner i=new Test().new Inner();
12 Test.Inner i = new Test.Inner();
13
```

```
\OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 26 - 01 - 2018
1 class Test
2 {
3     int x =10;
4     public static void m1()
5     {
6     }
7     public void m2()
8     {
9     }
10}
```

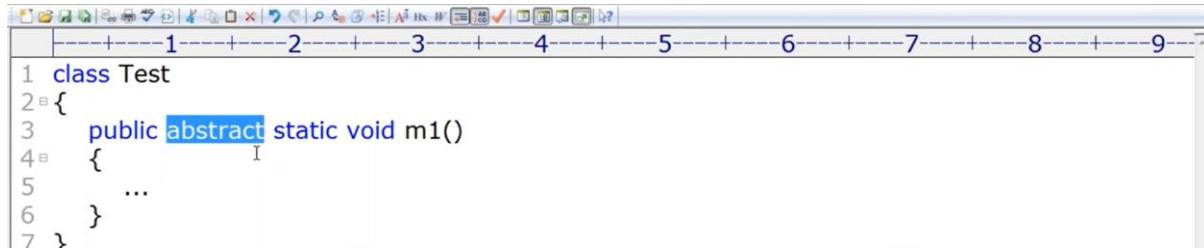
Questions

X	Question	Asker	R...
S	sir which are the top level class?	Pooja Chavan	
	the class which have main	Shubham Gupta	

Static method should always have an implementation cant be empty or without body

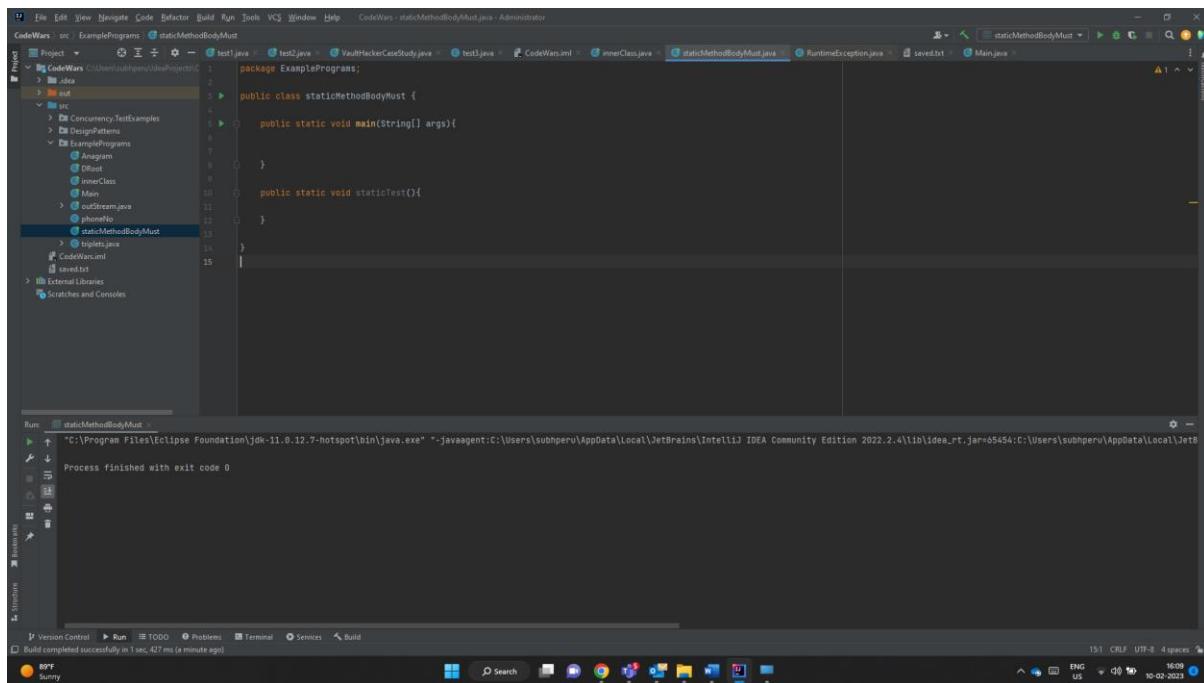
Abstract method cannot have an implementation, static method should have an implementation

So abstract method can be static. Illegal combination of modifiers.



```
1 class Test
2 {
3     public abstract static void m1()
4     {
5         ...
6     }
7 }
```

The below works static method the implementation is empty



D:\durgaclasses\Test.java - EditPlus

```
1 public class Test
2 {
3     String myStr="7007";
4     public void doStuff(String s)
5     {
6         int myNum=0;
7         try
8         {
9             String myStr=s;
10            myNum=Integer.parseInt(myStr);
11        }
12        catch(NumberFormatException e)
13        {
14            System.err.println("Error");
15        }
16        System.out.println("myStr: "+myStr+" ,myNum: "+myNum);
17    }
18    public static void main(String[] args)
19    {
20        Test t = new Test();
21    }
}
```

For Help, press F1

In 21 col 27 23 00 PC ANSI 9:58 PM 1/26/2018

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 26 - 01 - 2018

```
Test.java:1: error: Test is not abstract and does not override abstract method m1() in Test
class Test
^
2 errors

D:\durgaclasses>javac Test.java

D:\durgaclasses>java Test
myStr: 7007 ,myNum: 9009
```

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir Demo On 26 - 01 - 2018

```
1 class Test
2 {
3     int x =10;
4     static int y = 20;
5     public static void main(String[] args)
6     {
7         Test t1 =new Test();
8         Test t2 =new Test();
9         t1.x=100;
10        t1.y=200;
11        t2.x=300;
12        t2.y=400;
13        System.out.println(t1.x+" .. "+t1.y+" .. "+t2.x+" .. "+t2.y);
14    }
15 }
```

```
1 ERROR  
D:\durgaclasses>javac Test.java  
D:\durgaclasses>java Test  
100..400..300..400
```

5 4 5 6

D:\durgaclasses\Test.java - EditPlus

```
1 public class Test
2 {
3     static int x;
4     int y;
5     public static void main(String[] args)
6     {
7         Test t1 = new Test();
8         Test t2 = new Test();
9         t1.x=3;
10        t1.y=4;
11        t2.x=5;
12        t2.y=6;
13        System.out.println(t1.x+":"+t1.y+":"+t2.x+":"+t2.y);
14    }
15 }
```

OCJA 1.8 Java SE 8 Programmer I (1Z0-808) By Durga Sir Demo On 26 - 01 - 2018

```
1 public class Test
2 {
3     static int count=0;
4     int i=0;
5     public void modify()
6     {
7         while(i<5)
8         {
9             i++;
10            count++;
11        }
12    }
13    public static void main(String[] args)
14    {
15        Test t1 = new Test();
16        Test t2 = new Test();
17        t1.modify();
18        t2.modify();
19        System.out.println(t1.count+" "+t2.count);
20    }
}
```

138
139 What is the result?
140 A) 10..10

Instance variable can't be accessed in static method. Count

```
145
146 Q5. Consider the code
147 class Test
148 {
149     int count;
150     public static void display()
151     {
152         count++;//Line-1
153         System.out.println("Welcome Visit Count:"+count);//Line-2
154     }
155     public static void main(String[] args)
156     {
157         Test.display();//Line-3
158         Test.display();//Line-4
159     }
160 }
161 What is the result?
162
163 A) Welcome Visit Count: 1
164     Welcome Visit Count: 2
165 A) Welcome Visit Count: 1
166     Welcome Visit Count: 1
167 C) Compilation Fails at Line-1 and Line-2
168 D) Compilation Fails at Line-3 and Line-4
```

161 What is the result?
162
163 A) Welcome Visit Count: 1
164 Welcome Visit Count: 2
165 A) Welcome Visit Count: 1
166 Welcome Visit Count: 1
167 C) Compilation Fails at Line-1 and Line-2
168 D) Compilation Fails at Line-3 and Line-4

A screenshot of a Java IDE window titled "OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 26 - 01 - 2018". The code editor displays the following Java code:

```
1 public class Test
2 {
3     public static int x=100;
4     public int y = 200;
5     public String toString()
6     {
7         return y+":"+x;
8     }
9     public static void main(String[] args)
10    {
11        Test t1 = new Test();
12        t1.y=300;
13        System.out.println(t1); //300:100
14        Test t2 = new Test();
15        t2.x=300;
16        System.out.println(t2); //200:300
17    }
18
19 }
```

The code demonstrates variable initialization and printing. Lines 12 and 15 show assignments to local variables. Line 13 prints the object state, and line 16 prints the object state.

Local variables should always be initialized before use. The if block can or cannot be executed so compiler throws error at line 2.

A screenshot of a Java IDE window titled "OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir Demo On 26 - 01 - 2018". The code editor displays the following Java code:

```
1 public class Triangle
2 {
3     static double area;
4     int b=30,h=40;
5     public static void main(String[] args)
6     {
7         double p,b,h;// Line-1
8         if(area == 0)
9         {
10            b=3;
11            h=4;
12            p=0.5;
13        }
14        area=p*b*h;// Line-2
15        System.out.println(area);
16    }
17 }
```

This code attempts to calculate the area of a triangle using the formula $\text{area} = \frac{1}{2} \times \text{base} \times \text{height}$. It declares a static variable `area` and initializes `b` and `h` directly. The `if` block is intended to initialize `p`, but since `area` is not yet initialized, the condition `area == 0` is false, and the code inside the block is not executed.

D:\durgaclasses>javac Test.java

```
1 error
1 class
2 {
3     static
4     int p
5     public
6     {
7         Test.java:14: error: variable p might not have been initialized
8             area=p*b*h;// Line-2
9                 ^
10            Test.java:14: error: variable b might not have been initialized
11             area=p*b*h;// Line-2
12                 ^
13            Test.java:14: error: variable h might not have been initialized
14             area=p*b*h;// Line-2
15                 ^
16
17 3 errors
```

D:\durgaclasses>javac Test.java

```
1 class Demo
2 {
3     int ns;
4     static int s;
5     Demo(int ns)
6     {
7         if(s<ns)
8         {
9             s=ns;
10            this.ns=ns;
11        }
12    }
13    void display()
14    {
15        System.out.println("ns = "+ns+" s = "+s);
16    }
17 }
```

D:\durgaclasses>javac Test.java

```
19 public class Test
20 {
21     static int x;
22     int y;
23     public static void main(String[] args)
24     {
25         Demo d1= new Demo(50);
26         Demo d2= new Demo(125);
27         Demo d3= new Demo(100);
28         d1.display();//50..125
29         d2.display();//125..125
30         d3.display()//0..125
31     }
32 }
```

Compile error in line1 and Line2

The screenshot shows the Java code for a class named Test. The code attempts to divide an integer by zero and prints the result. Two specific lines are highlighted in red, indicating they are causing compilation errors:

```
1 public class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             int n=10;
8             int d=0;
9             int ans=n/d;
10        }
11        catch (ArithmeticException e)
12        {
13            ans=0;//Line-1
14        }
15        catch(Exception e)
16        {
17            System.out.println("Invalid Calculation");
18        }
19        System.out.println("Answer="+ans);//Line-2
20    }
}
```

The IDE interface includes tabs for Untitled3, Test.java, and TypesOfVariables_44. The status bar at the bottom shows the current file is Test.java.

The screenshot shows the command-line output of the javac compiler. It lists two errors, both related to the variable 'ans' being used before it is declared:

```
8 D:\durgaclasses>javac Test.java
9 Test.java:13: error: cannot find symbol
10         ans=0;//Line-1
11             ^
12         symbol:   variable ans
13         location: class Test
14 Test.java:19: error: cannot find symbol
15         System.out.println("Answer="+ans);//Line-2
16             ^
17         symbol:   variable ans
18         location: class Test
19 2 errors
```

Space is the default value for char “ ”

The screenshot shows the EditPlus IDE interface with a Java file named 'Test.java' open. The code defines a class 'Test' with fields 'c' (char), 'b' (boolean), and 'f' (float). It contains a 'print()' method that prints the values of these variables. The 'main()' method creates an instance of 'Test' and calls its 'print()' method. The code is as follows:

```
1 public class Test
2 {
3     char c;
4     boolean b;
5     float f;
6     public void print()
7     {
8         System.out.println("c = "+c);
9         System.out.println("b = "+b);
10        System.out.println("f = "+f);
11    }
12    public static void main(String[] args)
13    {
14        Test t = new Test();
15        t.print();
16    }
17 }
```

The status bar at the bottom indicates the file is 'Test.java' and the encoding is 'ANSI'. The system tray shows the date and time as 1/26/2018.

default value of char is \u0000 its like Square like 0 with a diagonal slash not blank
check ide

The screenshot shows a terminal window with the following output:

```
5
6      pu D:\durgaclasses>javac Test.java
7
8      D:\durgaclasses>java Test
9      c =
10     b = false
11     f = 0.0
```

999 and 777

The screenshot shows the EditPlus IDE interface with a Java file named 'Test.java' open. The code defines a class 'Test' with a method 'm1(int x)' that prints the value of 'x'. The 'main()' method calls 'm1(999)' and then 'm1(777)'. Both calls result in the output "Inside Method:" followed by the respective value. The code is as follows:

```
1 public class Test
2 {
3     public static void m1(int x)
4     {
5         x=999;
6         System.out.println("Inside Method:"+x);
7     }
8     public static void main(String[] args)
9     {
10        int x=777;
11        m1(x);
12        System.out.println("Inside Main method:"+x);
13    }
14 }
```

999 999

If we are passing primitive variable reflection would not happen whereas if we pass object ref reflection would happen

The screenshot shows a Java code editor with the following code:

```
1 public class Test
2 {
3     int x=10;
4     public static void m1(Test t1)
5     {
6         t1.x=999;
7         System.out.println("Inside Method:"+t1.x);
8     }
9     public static void main(String[] args)
10    {
11        Test t= new Test();
12        t.x=777;
13        m1(t);
14        System.out.println("Inside Main Method:"+t.x);
15    }
16
17
18 }
19 }
```

A hand-drawn diagram is overlaid on the code editor. It features a large red circle containing the number "999". A wavy line connects this circle to a smaller red circle containing the number "777". Another wavy line connects the "777" circle to the line of code "t1.x=999;". A third wavy line connects the "777" circle to the line of code "System.out.println("Inside Method:"+t1.x);". A fourth wavy line connects the "777" circle to the line of code "t.x=777;". A fifth wavy line connects the "777" circle to the line of code "m1(t);". A sixth wavy line connects the "777" circle to the line of code "System.out.println("Inside Main Method:"+t.x);".

The screenshot shows a Java code editor with the following code:

```
1 public class Test
2 {
3     public static void main(String[] args)
4     {
5         int x = 200;
6         System.out.print(m1(x)); //400
7         System.out.print(" "+x); //200
8     }
9     public static int m1(int x)
10    {
11        x=x*2;
12        return x;
13    }
14 }
```

To the right of the code editor is a list of questions from a forum. The list includes:

X	Question	Asker
not requires	Kalyan KK	
instance	Kalyan KK	
instance	Rajat Gupta	
ins	Shubham Gupta	
999 999	RAAMESWARA REDDY	
999	SANDEEP REDDY	
999	SANDEEP REDDY	
999 999	Rich Cohen	
999 and 999	Rajat Gupta	
999 and 999	SANDEEP REDDY	