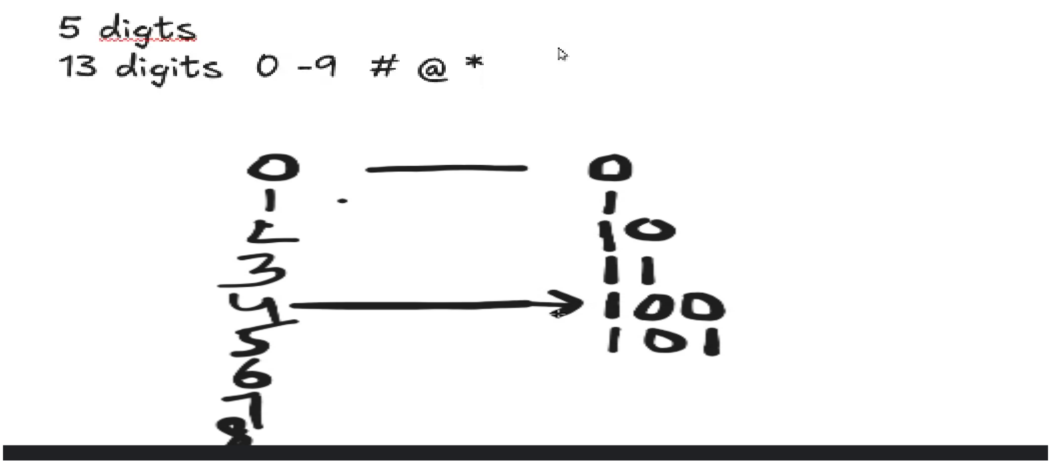
DAY5
654- accepted octal value
876- not an octal number

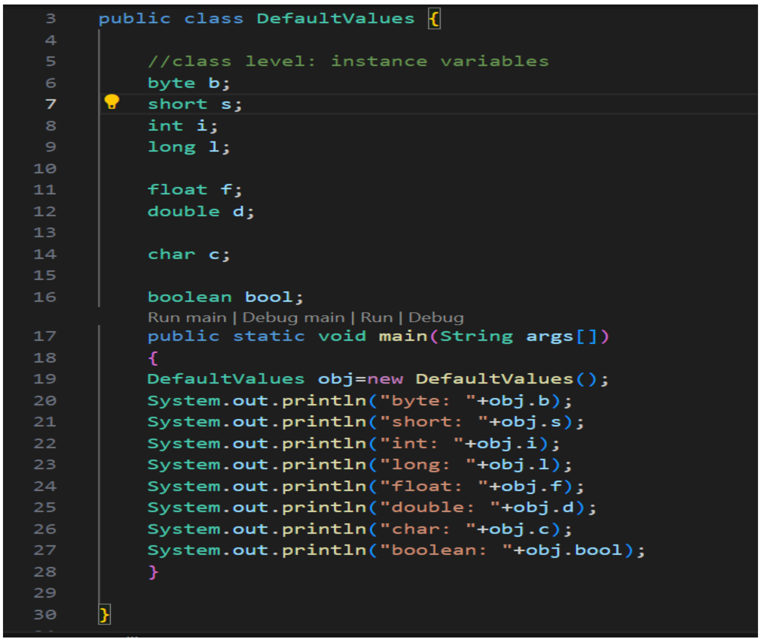


How constants are declared in java?
We use final keyword to declare constant. Constants are fixed values, once assigned we can't change its value. If we try to change it then error will arise.

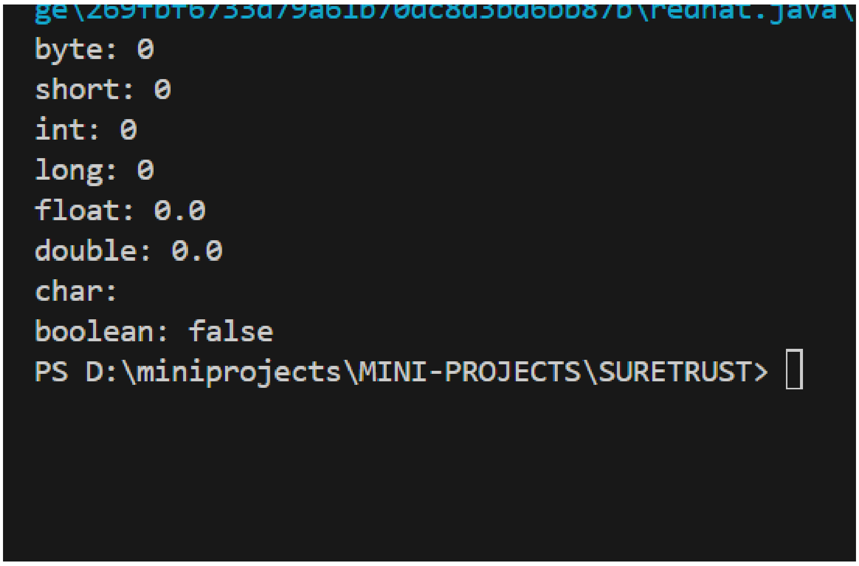
- Float will accept 8 digits after decimal, it will loose its precision after 8 digits after decimal.
- We have to explicitly mention the suffix to float. i.e **3.14f**
- Double will have 17 digits precision after decimal point.
- For double mentioning the suffix is optional.
- For float we need to explicitly mention the suffix f or f because by default the decimal values in java are considered as double. So if we mention 3.14, it is considered as double. So in order make that as float, use the suffix f or F.
- if we just mention , float f=3.14, we will get possible lossy conversion, because conversion from larger datatype to smaller datatype is not done implicitly.

We should not leave primitives uninitialized, we should initialize them atleast any where in the program. Otherwise, error: the variable may not have intialized.

```
//We can't check default values for primitives in the case of local variable(variables that are declared inside the method).
//If we want to check, then primitives should be declared class level, i.e instance variable (variables that are declared inside the class but outside the method).
//Check the values through creating an object for the class and printing them with the object.
```

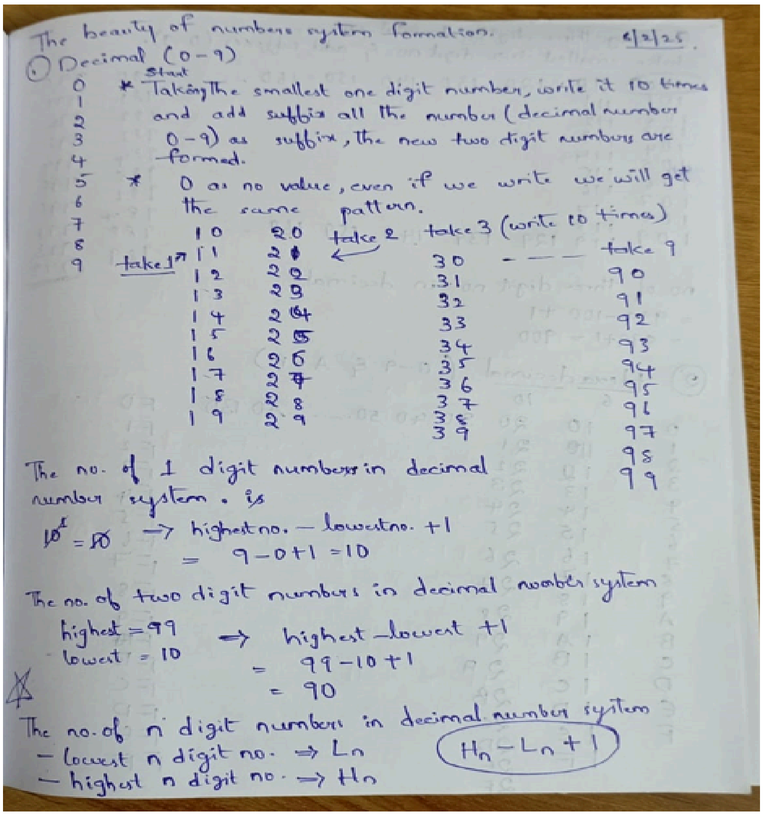


```java
public class DefaultValues {

    //class level: instance variables
    byte b;
    short s;
    int i;
    long l;

    float f;
    double d;

    char c;

    boolean bool;
    public static void main(String args[])
    {
    DefaultValues obj=new DefaultValues();
    System.out.println("byte: "+obj.b);
    System.out.println("short: "+obj.s);
    System.out.println("int: "+obj.i);
    System.out.println("long: "+obj.l);
    System.out.println("float: "+obj.f);
    System.out.println("double: "+obj.d);
    System.out.println("char: "+obj.c);
    System.out.println("boolean: "+obj.bool);
    }

}
```



```
ge\269fbf6733d79a61b70dc8d3bd6bb87b\redhat.java\
byte: 0
short: 0
int: 0
long: 0
float: 0.0
double: 0.0
char:
boolean: false
PS D:\miniprojects\MINI-PROJECTS\SURETRUST>
```

How data is stored in the memory?
The data is stored in the binary form inside the memory.



The beauty of numbers system formation. 6/2/25.

① Decimal (0 – 9)

0 1 2 3 4 5 6 7 8 9

* Start Taking The smallest one digit number, write it 10 times and add suffix all the number (decimal number 0 – 9) as suffix, the new two digit numbers are formed.

* 0 as no value, even if we write we will get the same pattern.

take 1 → 10 11 12 13 14 15 16 17 18 19

take 2 → 20 21 22 23 24 25 26 27 28 29

take 3 (write 10 times) 30 31 32 33 34 35 36 37 38 39

– – – take 9 90 91 92 93 94 95 96 97 98 99

The no. of 1 digit numbers in decimal number system. is

$\cancel{10}^1 = \cancel{10}$ → highest no. – lowest no. + 1
= 9 – 0 + 1 = 10

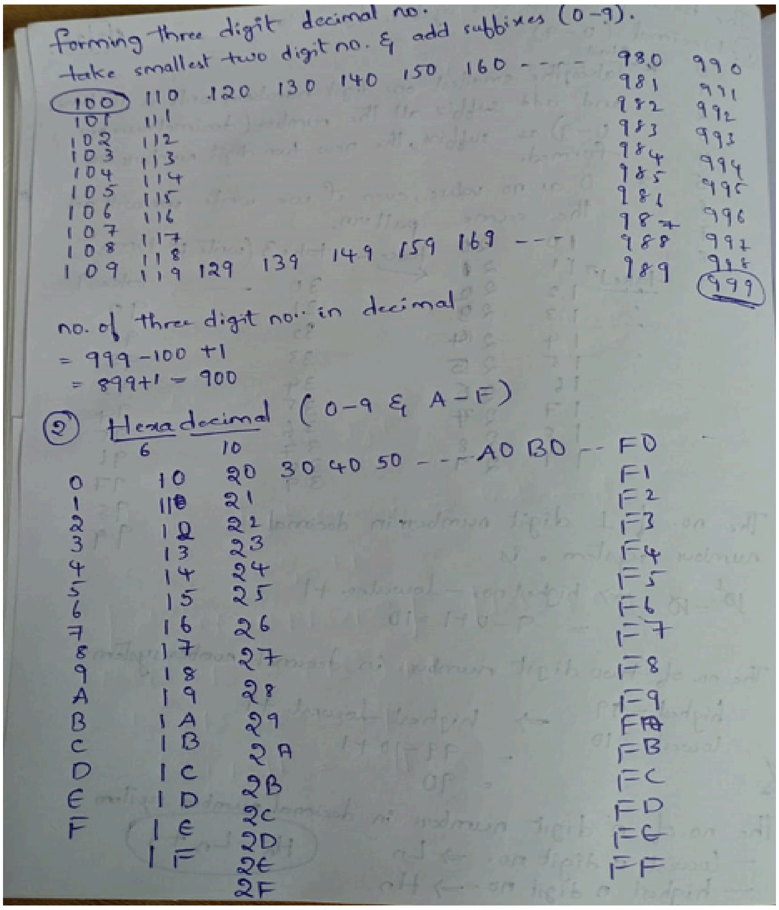
The no. of two digit numbers in decimal number system

highest = 99
lowest = 10 → highest – lowest + 1
= 99 – 10 + 1
= 90

The no. of n digit numbers in decimal number system

– lowest n digit no. ⇒ L_n
– highest n digit no. ⇒ H_n

$\boxed{H_n - L_n + 1}$



forming three digit decimal no.
take smallest two digit no. & add suffixes (0 – 9).

100 101 102 103 104 105 106 107 108 109

110 111 112 113 114 115 116 117 118 119

120 – – 129

130 – – 139

140 – – 149

150 – – 159

160 – – 169

– – –

980 981 982 983 984 985 986 987 988 989

990 991 992 993 994 995 996 997 998 999

no. of three digit no. in decimal
= 999 – 100 + 1
= 899 + 1 = 900

② Hexadecimal (0 – 9 & A – F)
6 10

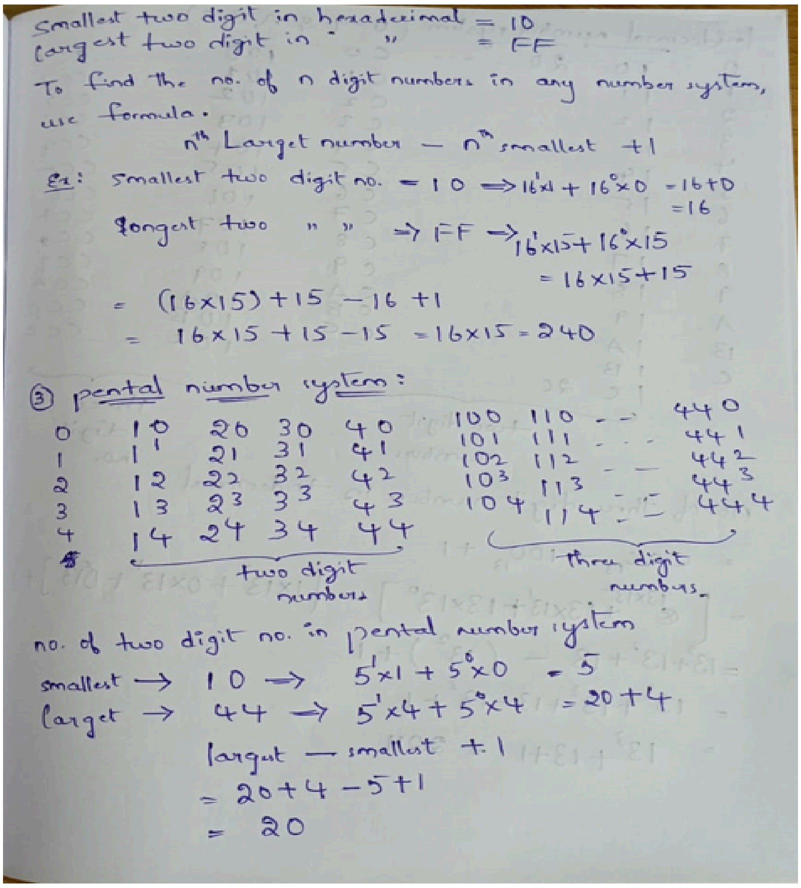
0 1 2 3 4 5 6 7 8 9 A B C D E F

10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F

30 40 50 – – – A0 B0 – –

F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF



Smallest two digit in hexadecimal = 10
Largest two digit in " " = FF

To find the no. of n digit numbers in any number system, use formula.

n^{th} Larget number − n^{th} smallest +1

Ex: Smallest two digit no. = 10 $\Rightarrow 16^1\times1 + 16^0\times0 = 16+0$
$= 16$

~~L~~longest two " " $\rightarrow$ FF $\rightarrow 16^1\times15 + 16^0\times15$
$= 16\times15+15$

$= (16\times15)+15 - 16 + 1$
$= 16\times15 + 15 - 15 = 16\times15 = 240$

③ pental number system:

0	10	20	30	40	100	110	--	440
1	11	21	31	41	101	111	...	441
2	12	22	32	42	102	112	--	442
3	13	23	33	43	103	113	--	443
4	14	24	34	44	104	114	--	444
~~5~~								

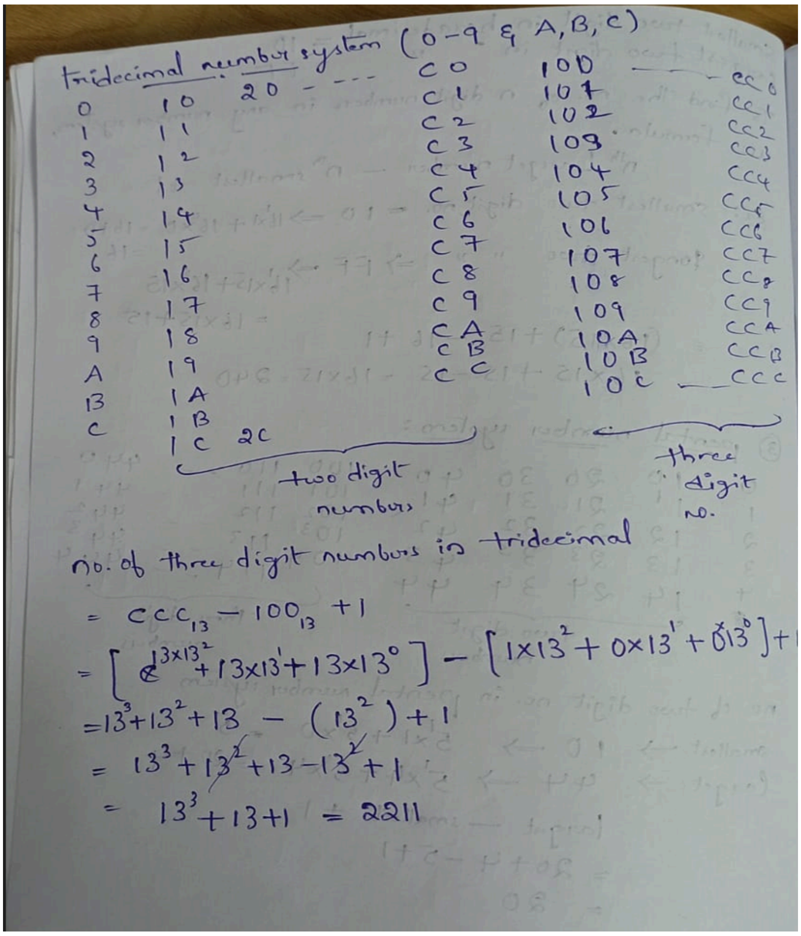
two digit numbers

three digit numbers.

no. of two digit no. in pental number system

smallest $\rightarrow$ 10 $\rightarrow 5^1\times1 + 5^0\times0 = 5$
larget $\rightarrow$ 44 $\rightarrow 5^1\times4 + 5^0\times4 = 20+4$

larget − smallest + 1
$= 20+4-5+1$
$= 20$



tridecimal number system (0-9 & A, B, C)

0
1
2
3
4
5
6
7
8
9
A
B
C

10
11
12
13
14
15
16
17
18
19
1A
1B
1C

20 - ---
2C

two digit numbers

C0
C1
C2
C3
C4
C5
C6
C7
C8
C9
CA
CB
CC

100 ————— CC0
101
102
103
104
105
106
107
108
109
10A
10B
10C ——— CCC

CC1
CC2
CC3
CC4
CC5
CC6
CC7
CC8
CC9
CCA
CCB

three digit no.

no. of three digit numbers in tridecimal

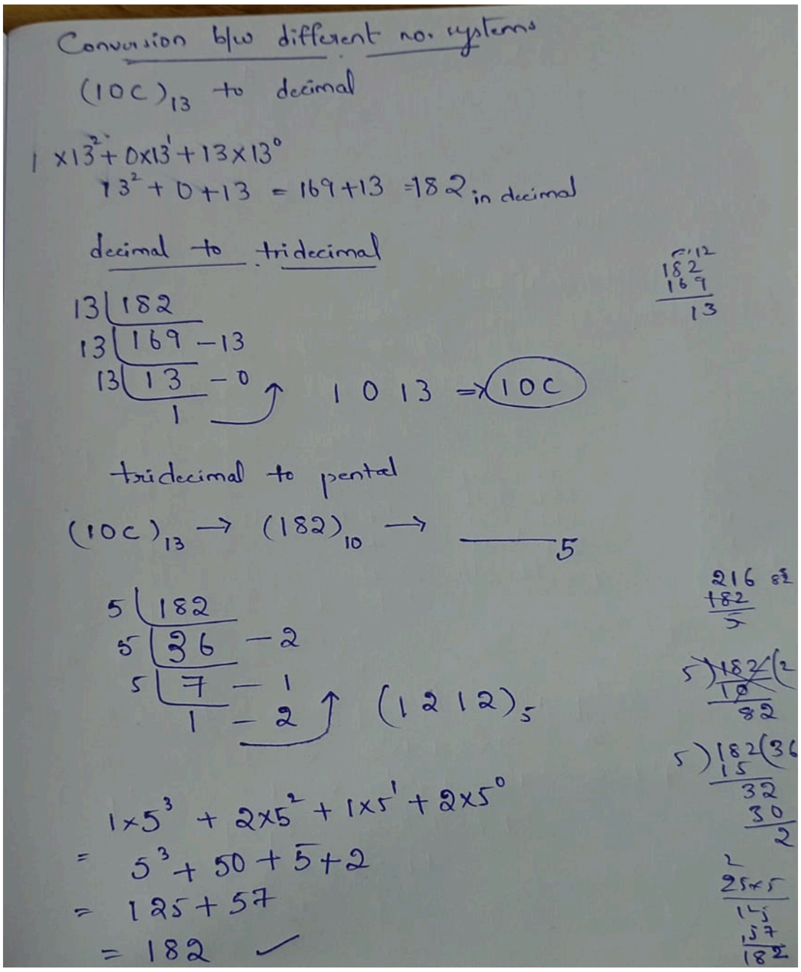
$= CCC_{13} - 100_{13} + 1$

$= [~~C~~^{13\times13^2} + 13\times13^1 + 13\times13^0] - [1\times13^2 + 0\times13^1 + 0\times13^0] + 1$

$= 13^3 + 13^2 + 13 - (13^2) + 1$

$= 13^3 + \cancel{13^2} + 13 - \cancel{13^2} + 1$

$= 13^3 + 13 + 1 = 2211$



Conversion b/w different no. systems

$(10C)_{13}$ to decimal

$1 \times 13^2 + 0 \times 13^1 + 13 \times 13^0$

$13^2 + 0 + 13 = 169 + 13 = 182$ in decimal

decimal to tridecimal

13 | 182
13 | 169 − 13
13 | 13 − 0
1

1 0 13 ⇒ 10C

182
169
13

tridecimal to pental

$(10C)_{13} \rightarrow (182)_{10} \rightarrow \_\_\_\_\__5$

5 | 182
5 | 36 − 2
5 | 7 − 1
1 − 2

$(1212)_5$

216
182
5

5) ~~182~~ (2
~~10~~
82

5) 182 (36
15
32
30
2

$1 \times 5^3 + 2 \times 5^2 + 1 \times 5^1 + 2 \times 5^0$

$= 5^3 + 50 + 5 + 2$

$= 125 + 57$

$= 182$ ✓

25×5
125
57
182