Today's topic of discussion
==========
 1. OOPS(basic introduction)
 2. Identifier/variables
            a. rules to write an identifier
 3. Reserved words
 4. Data types and its chart.

yesterday topic of discussion
---
 1. JShell
 2. main method(public static void main(String[] args))
 3. command line arguments execution using IDE and command prompt.

OOPS

   It stands for Object Orientation Principles.
  Object -> real time instance or an entity.
                  eg: Car,Student,Employee
  Every object in realtime will have 2 parts
            what it has
            what it does

  eg: Car
                  brandName
                  noOfWheels
                  model
                  speed

                  move
                  accelerate
                  brake

Java code

```
class Car
{
            //HAS part of an Object is represented as a "variable".
            String brandName;
            int  noOfWheels;

            //DOES part of an object is represented through "methods"
              public  void move()
              {
                              //logic of moving a vehicle
              }

             public void accelerate()
             {
                        //logic of acclerating a vehicle
             }
}
```

#2.
  class Student
  {
            //HAS-part (variables)
              String name;
              int id;

```
              float height;

            //DOES-part(methods)
            public void study(){
                  //logic of studying
            }
            public void play(){
                  //logic of playing
            }
  }
```

Identifier
========
  It is a name in java program.
  It can be a classname,methodname,variable name and label name.

```
class Test{
      public static void main(String[] args){
                  int x= 10;
      }
}
```
Totally 5 identifers

```
eg#2.
class Test{
      public static void main(String[] args){
                  System.out.println("sachin");
      }
}
```
Totally 7 identifiers.

Rules(syntax for compiler + jvm) for writing an identifier

Rule1: The only allowed characters in java identifiers are
a to z, A to Z,0 to 9, _(underscore),$

Rule2: If we use any other characters it would result in compile time error
```
                  int ^* = 10;(invalid)
                  int total = 10;(valid)
                  int total#= 35;(invalid)
```

Rule3: Identifiers are not allowed to start with digits
```
                  int telusko1 =100;(valid)
                  int 1telusko = 100;(invalid)
```

Rule4: Java identifiers are case sensitive,meaning number and Number is different.
```
      class   Demo{
                        int number=10;
                        int Number=20;
                        int nUmber= 30;
                        int NUMBER = 40;
            }
```

Rule5: There is no lenght limit on java identifiers, but still it is a good practise to keep the
length of the identifier not more than 15characters.
```
            int priorityOfThreadWithMinValue = 1;
```

Rule6: We can't use reserve words as a identifers.
```
        eg:  int if = 10; //CE
```

Rule7: Predefined class names can be used as identifiers like String,Runnable

eg#1
```
String Runnable = "sachin";
System.out.println(Runnable);//sachin
```

eg#2
```
int String = 10;
System.out.println(String);//10
```

Note: Even though predefined class names can be used as an indetifiers,it is not a good practise to keep.

Interview Question
```
-------------------------
int If =10;//if and If is different
System.out.println(If);//10

int Integer = 10;
System.out.println(Integer);//10

int int =10;//CE
System.out.println(int);
```

ReservedWords
```
--------------------
```
It is a built in words/keywords which has already a predefined meaning to it.
refer: Reservewords.png

Note:
Literal Any constant value which can be assigned to a variable is called literal
```
int data =10;
literal -> 10
data  -> variableName/identifier
int     -> datatype/reserveword
```

Note: for boolean datatypes the only values allowed for a variable is "true,false", other than this if we try
to keep any values it would reslut in "CompileTimeError".
=> All reserved words names would start with "lower case ".
=> In java all Classnames/interfacenames would start with "upper case".

Which of the following list contain only reservewords/keywords/builtinwords?

1. final,finally,finalize
ans. finalize is not a reserveword, it is a method in Object class.

2. break,continue,return,exit
ans. exit is not a reserve word, it is a method in System class

3. byte,short,Integer,long
ans. Integer is not a reserve word,it is a predefined class

4. throw,throws,thrown

ans. thrown is not a reserve word,it is a userdefined variable.

Datatypes

Every varaible has a type,every expression has a type and all types are strictly typed/define in java
becoz java is strictlytype /statically typed language.
Compiler role -> Compiler will check the value stored can be handled by datatype or not
This checking which is done by compiler is called "TypeChecking/Strictlytype checking".

Primtive datatypes

meaning -> data which is commonly used and supported by any langauge to store directly.

a.Numeric values
=> to store number
a. whole number
b. realnumber
b.character values
=> to store character type of data
c.boolean values
=> to store logical values

Number data

To store whole numbers we have 4 datatypes
a. byte
b. short
c. int
d. long

datatype information like
a. size of dataype(how much memory is allocated on the ram for that datatype by JVM)
b. minvalue what it can keep
c. maxvalue what it can keep

note:

```
System.out.println("Size of byte is :: "+Byte.SIZE);
System.out.println("MINVALUE  of byte is :: "+Byte.MIN_VALUE);
System.out.println("MAXVALUE of byte is :: "+Byte.MAX_VALUE);
```

byte:
size -> 8 bits
minvalue -> -128
maxvalue -> 127

eg:

```
byte marks=35   //valid
byte  marks = 135; //CE: possible loss of precission
byte  marks = -1;//valid

byte a = true;//CE: incompatible types
byte b = "nitin";//CE: incompatible types
```

When to use byte datatype?

it is commonly used when we handle the data which is coming from stream,network.
stream -> java.io package
" " -> means String data
' ' -> char data

short
System.out.println("Size of short is :: "+Short.SIZE);
System.out.println("MINVALUE of short is :: "+Short.MIN_VALUE);
System.out.println("MAXVALUE of short is :: "+Short.MAX_VALUE);

size : 16bits(2 byte)
minvalue: -32768
maxvalue: +32767

eg:
short data=137;//valid
short data = true; //CE: incompatible types
short data = "sachin";//CE:incompatible types

Note: This data is not at all used in java and this data type is best sutied only if u have old processors like 8086.

int:
System.out.println("Size of int is :: "+Integer.SIZE);
System.out.println("MINVALUE of int is :: "+Integer.MIN_VALUE);
System.out.println("MAXVALUE of intt is :: "+Integer.MAX_VALUE);

size: 32bits(4 bytes)
minvalue:-2147483648
maxvalue: 2147483647

eg: int data = 323445;
int result = true;//ce:incompatible types
int result ="pass";//ce:incompatible types

note: The most commonly used datatype for storing whole number is "int" only and by default if we specify any literal of
number type compiler will try to keep it as "int" only,but we can keep either in short or byte also.

long
System.out.println("Size of long is :: "+Long.SIZE);
System.out.println("MINVALUE of long is :: "+Long.MIN_VALUE);
System.out.println("MAXVALUE of long is :: "+Long.MAX_VALUE);

size: 64bits(8bytes)
minvalue:-9223372036854775808
maxvalue:9223372036854775807

eg: long data = 10;
long data = 9223372036854775807 ;

If the data goes beyond the range of int,then to keep the data inside long data type we need to explicitly suffix the
data with 'L' or 'l' otherwise it would result in "CompileTimeError".
eg: long firstData=9223372036854775807;//CE
long secodData=9223372036854775807;//CE

long firstData=9223372036854775807L;

```
                     long secodData=92233720368547758O7l;

                    long data = 10L;//now also no problem becoz 10 is int and we
have added 'L'
                    long number = 5l;
```

Note:

When int is not enough to hold the big values, then we use long data type.

When we work with large files, data would come to java program in terms of GB's.

```
            long size = file.length();
```