② Keywords, Variables, Identifiers & Data tables.

ⓐ ⇒ All known.

Long & float datatypes must have suffix.
→ L is necessary at last in nums.

ⓑ ⇒ camel Casing followed.
For classes, pascal casing.

// & /* */ } for comments.
/** */ → documentation.

③ Operators.

ⓐ ⇒ Diff operators given:

- Unary → ++, --, !, ~
- Arithmetic → +, -, *, /, %
- Relational → ==, >=, <=, >, <, !=
- Logical → AND, OR, NOT (&&, ||)
- Ternary → ? :
- Assignment → =, +=, *=, etc.
- Bitwise → &, |, <<, >>.

④ Type Conversion.

int + float = float
int + long + double = double
} whichever is largest of all, everything gets converted to that.

If you want double to int or such, typecast should be used.

Java. Generics is Templetes in C++.

f) Java
- → OOPS
- → Multi threeded
- → Architecture neutral.
- → Secure
- → Robus (Strong type checking & Auto memory management).

g) JIT compiler for optimizing reusable byte code to machine cod

' ' → char

" " → string.

Decimal Formatter.

5 Control structures

a) if else , switch , while , do-while
break , continue.

decimal → hexa
octa.
use radix..
(or)
Int.toHexString.

b) Switch syntax →

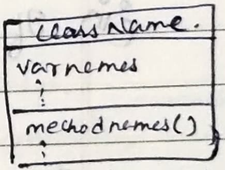
```
switch (variable){
    case value1: <statement>;
                 break;
    default : <statements>;
}
```

6 OOPS

a) → we write code for one customer & order type but there can be thousands & thousands of both.

b) → Focus is not only on functionality (methods) but also on data of real life entities.

c) → Classes → methods. (explain behaviour of parts.)
→ variables/attributes. (explain names of parts.)
↓
obj (instance of the class)



+ publicmemb
- privete
protected.

d) →

```
Class Customer {
        public String varname1 ;
          "     type      "    2 ;
          "     type      "    3 ;
}
```

} Access modifier datatypes & names/Attributes.

Then add methods
then type main class & use obj to use OOPS.

Customer myCustomer = new Customer();