**YouTube tutorial 17 – Generic method**

**class** apples {

**public** **static** **void** main(String args[]) {

Integer[] iray = {1,2,3,4};

Character[] cray = {'b','u','c','k','y'};

*printMe*(iray);

*printMe*(cray);

}

**public** **static** **void** printMe(Integer[] i){

**for**(Integer x :i)

System.*out*.printf("%s ",x);

System.*out*.println();

}

**public** **static** **void** printMe(Character[] i){

**for**(Character x :i)

System.*out*.printf("%s ",x);

System.*out*.println();

}

}

The result is:

1 2 3 4

b u c k y

But how can you make it a generic method instead of making all those overriding methods?

-It’s shown in the next page.

**YouTube tutorial 18 – Generic method’s implementation**

**class** apples {

**public** **static** **void** main(String args[]) {

Integer[] iray = {1,2,3,4};

Character[] cray = {'b','u','c','k','y'};

*printMe*(iray);

*printMe*(cray);

}

//Generic method

**public** **static** <T> **void** printMe(T[] i){

**for**(T x :i)

System.*out*.printf("%s ",x);

System.*out*.println();

}

}

The result is the same as above:

1 2 3 4

b u c k y

By making this generic method, it can take any type of object. Keep in mind that primitives (int) weren’t used. It’s because primitive types cannot be used with generics.