Set1

1. Calculate n!

n = 10, 100, 1000, ..

Print first 10 digits to verify the result

**SOLUTION:**

import sys

sys.set\_int\_max\_str\_digits(0)

n = [10,100,1000,10000,100000,1000000,10000000,100000000,1000000000,10000000000]

for x in n:

fact = 1

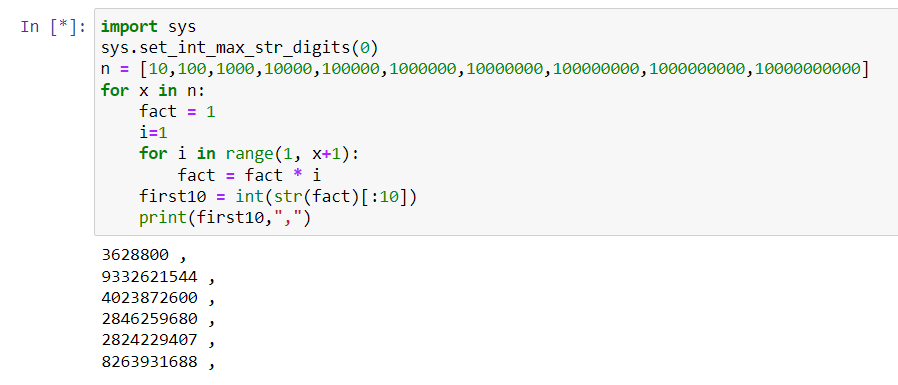
i=1

for i in range(1, x+1):

fact = fact \* i

first10 = int(str(fact)[:10])

print(first10)



1. Detect potential functional and execution issues:

void transfer(Account from, Account to, int amount) {

synchronized(from){

   synchronized(to) {

          from.debit(amount);

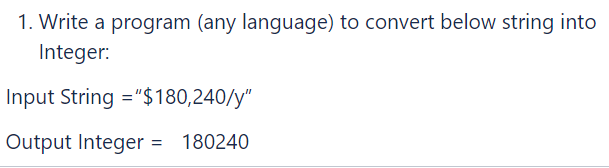
          to.credit(amount);

   }

}

**SOLUTION:**

1. There is no check if the “from” account has the balance greater than or equal to “amount”. This will give us error if there is no enough balance to debit.
2. What if amount being passed contains decimal values – It’s taking “amount” as integer. Hence decimal values may get lost and there should be a rounding off step before calling this “transfer” function.
3. There is one closing brace } missing.
4. “from” and “to” accounts should be validated and only if VALID, “transfer” function should be called otherwise ABORT.
5. As per my understanding, the synchronized keyword is a modifier that locks a method so that only one thread can use it at a time. Since ‘from’ and ‘to’ objects are synchronized separately, there could be a possibility of deadlock situation. Hence instead of defining ‘from’ and ‘to’ objects as synchronized separately, we can define the function transfer as synchronized - public static synchronized void transfer(Account from, Account to, int amount).



**SOLUTION:**

import re

input="$180,240/y"

input1=re.sub('[^0-9]', '', input)

output=int(input1)

print("Output Integer= ", output)

