//Task #4 Identity Matrix.

**public** **class** identity {

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

*Id*(3);

*Id*(5);

}

**public** **static** **void** Id(**int** n) {

**int**[][]A=**new** **int** [n][n];

**for**(**int** i=0;i<n;i++) {

**for**(**int** j=0;j<n;j++) {

A[i][j]=0;

}

}//initialize

**for**(**int** i=0;i<n;i++) {

A[i][i]=1;

}//add identity

**for**(**int** i=0;i<n;i++) {

**for**(**int** j=0;j<n;j++) {

System.***out***.print(A[i][j]+" ");

}

System.***out***.print("\n");

}//print

}//Id

}

//Output

1 0 0

0 1 0

0 0 1

1 0 0 0 0

0 1 0 0 0

0 0 1 0 0

0 0 0 1 0

0 0 0 0 1

//Task#5 Limits of simple types

//output for java

Script started on 2018-02-15 12:52:45-0500

#]0;jsheng@errai:~/cs330/lab4#[jsheng@errai lab4]$ ./size######gcc -o size size.c###################[12P./size

2147483637

2147483638

2147483639

2147483640

2147483641

2147483642

2147483643

2147483644

2147483645

2147483646

2147483647

-2147483648

-2147483647

-2147483646

-2147483645

-2147483644

-2147483643

-2147483642

-2147483641

-2147483640

-2147483639

-2147483638

-2147483637

-2147483636

-2147483635

-2147483634

-2147483633

-2147483632

//output for java

Script started on 2018-02-15 12:55:00-0500

#]0;jsheng@errai:~/cs330/lab4#[jsheng@errai lab4]$ java##[K##[K##[K##[K./size#######[Ksiz##[K##[K##[Kjava size

2147483637

2147483638

2147483639

2147483640

2147483641

2147483642

2147483643

2147483644

2147483645

2147483646

2147483647

-2147483648

-2147483647

-2147483646

-2147483645

-2147483644

-2147483643

-2147483642

-2147483641

-2147483640

-2147483639

-2147483638

-2147483637

-2147483636

-2147483635

-2147483634

-2147483633

-2147483632

-2147483631

-2147483630

-2147483629

-2147483628

-2147483627

-2147483626

-2147483625

-2147483624

-2147483623

-2147483622

-2147483621

-2147483620

-2147483619

-2147483618

I find that when i is maximum value (i=2147483647),and add 1 to i, error does not pop up in both C and Java. Instead, we will have i+1=-2147483648, which is the minimum value of int. And since -2147483648<M, we will continue the loop until next time approaching the maximum value of int and then become -2147483648 again when added 1. This whole process will keep repeating and the loop will never end.

//Task#6 Password Generator

**import** java.util.Random;

**public** **class** password {

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

/\* *passGen*(10);

*passGen*(8);

*passGen*(16);

*passGen*(13);

*passGen*(12);\*/

for(int i=0;i<1000000;i++){

passGen(8);

}

}

**public** **static** **void** passGen(**int** n) {

Random rand = **new** Random();

**char**[]A=**new** **char**[n];

**for**(**int** i=0;i<n;i++) {

A[i] = (**char**) (rand.nextInt(127-33)+33);

System.***out***.print(A[i]);

}

System.***out***.print("\n");

}

}

//Output

vJrRFTvk&5

/}\_^K|I\*

#swRz:|`z+!(NMuA

MNS0#IFxtkH|U

]MQ7lH@?Jx,s

I use the run passGen(8)for a million times and use java password | sort | uniq -c in linux command to see the occurrence of each password, in which I found all of them occurs only once, thus it is not very likely for the same password to appear twice.