//PROBLEM3.cpp

//Andrew Ribeiro

//October 6, 2008

//Files used: PROBLEM3\_FUNCTIONS.cpp

/\*File discription:

This program calculates the first 100 EMIRPS. EMIRPS are numbers that are naturaly prime and

prime when they are reversed.

Note: from my research single digit numbers are not considered EMIRPS; however, from the

assignment it looks like they are considered EMIRPS. This is easy to change if this is not

the case.

\*/

#include <iostream>

#include <iomanip>

using namespace std;

//Prototypes

int reverseNumber(int num);

bool isPrime(int num);

int main()

{

/\*incriment: is the starting number, of which the loop starts checking if it is a emirp.

this should be changed to 11 if single digit numbers are not considered emirps.

counter: counts how many emirps have been found.\*/

int incriment = 2,counter =0;

//Print header of menu.

cout<<endl

<<"-------------------------FIRST 100 EMIRPS---------------------"<<endl

<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

//0 - 99 = 100 iterations including 0.

//Loop through and find the first hundered numbers that are emirp.

while(counter < 100)

{

//If incriment is prime reversed and as is, it is emirp.

if(isPrime(reverseNumber(incriment)) && isPrime(incriment) )

{

if (counter % 10 == 0) //If 10 is in the row, make a new row.

{

cout<<"\n";

}

counter++;

cout<<setw(6)<<incriment;

}

incriment++;

}

//Print footer of menu.

cout<<endl<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n"<<endl;

return 0;

}