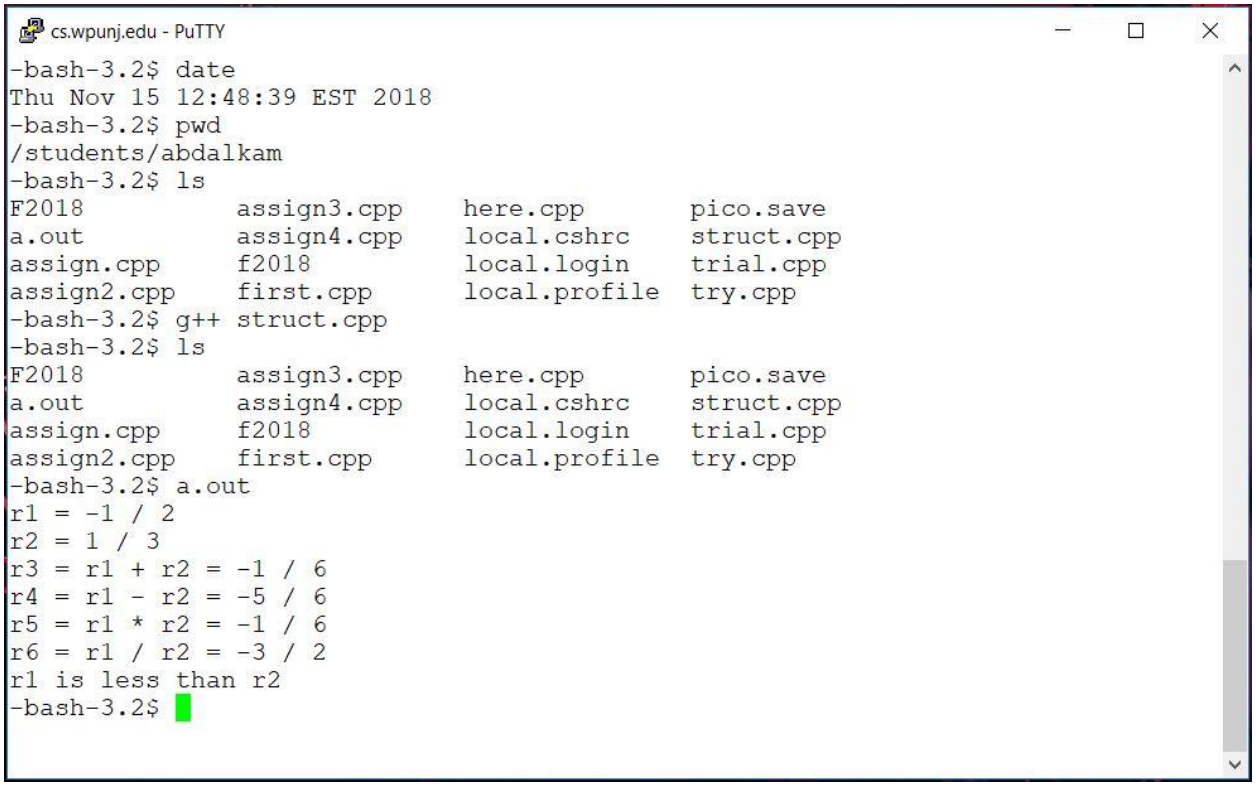
```
//             Course:  CS2400-60 Computer Science 2
//               Name:  Abdalkarim, Marina
//      Assignment:  Programming Assignment P6.1
//    Date assigned: 10/28/18
//           Date due: 11/27/17
//  Date handed in: 11/27/17
//             Remark: The program tests all functions.
#include <iostream>
using namespace std;
class rational
{
public:
        int helper();
        // Another accessor function; gaining access to the gcd function
        // Postcondition: returns the greatest common divisor
        void set(int aa, int bb);
        // Postcondition: rational number r is set to aa / bb
        void display(rational &r);
        // Postcondition: displays a rational number  r in the following format: a / b, e.g., 1 / 2,  -5
        // /9 (not 5 / -9), 1 / 4 (not 2 / 8, etc.)
        rational add(const rational &r1, const rational &r2);
        // Postcondition: (r1 + r2) -- a rational number -- is returned (notice the return type is
        // rational!)
        rational subtract(const rational &r1, const rational &r2);
        // Postcondition: (r1 - r2) -- a rational number -- is returned
        rational multiply(const rational &r1, const rational &r2);
        // Postcondition: (r1 * r2) -- a rational number -- is returned
        rational divide(const rational &r1, const rational &r2);
        // Postcondition: (r1 / r2) -- a rational number -- is returned
        int compare(const rational &r1, const rational&r2);
        // Postcondition: returns 1 if r1 is greater than r2; 0 if r1 is equal to r2; -1 is r1 is less than
        // r2
private:
        int GCD() const;
        // You must use the Euclidean algorithm.
        // https://en.wikipedia.org/wiki/Euclidean_algorithm
        // Postcondition: returns the "greatest common divisor" between r.a and r.b
        int num;        // numerator
        int den;        // denominator; b ≠ 0
```

```
};
int main()
{
	rational x, y, sum, diff, mul, div, comp;
	int com;
	x.set(-1, 2);
	y.set(1, 3);
	com = comp.compare(x, y);
	sum.add(x, y);
	diff.subtract(x, y);
	mul.multiply(x, y);
	div.divide(x, y);
	cout << "r1 = ";
	x.display(x);
	cout << "r2 = ";
	y.display(y);
	cout << "r3 = r1 + r2 = ";
	sum.display(sum);
	cout << "r4 = r1 - r2 = ";
	diff.display(diff);
	cout << "r5 = r1 * r2 = ";
	mul.display(mul);
	cout << "r6 = r1 / r2 = ";
	div.display(div);
	if (com == 1)
		cout << "r1 is greater than r2" << endl;
	else if (com == -1)
		cout << "r1 is less than r2" << endl;
	if (com == 0)
		cout << "r1 is equal to r2" << endl;
	return 0;
}
rational rational::add(const rational &r1, const rational &r2)
{
	rational sum;
	num = (r1.num * r2.den) + (r2.num * r1.den);
	den = r1.den * r2.den;
	sum.num = num;
	sum.den = den;
```

```
	return sum;
}
void rational::set(int a, int b)
{
	num = a;
	den = b;
}
void rational::display(rational &r)
{
	int gcd = GCD();
	int a, b;
	a = r.num / gcd;
	b = r.den / gcd;
	if (b < 0)
	{
		a = a * (-1);
		b = b * (-1);
	}
	cout << a << " / " << b << endl;
}
rational rational::subtract(const rational &r1, const rational &r2)
{
	rational diff;
	num = (r1.num * r2.den) - (r2.num * r1.den);
	den = r1.den * r2.den;
	diff.num = num;
	diff.den = den;
	return diff;
}
rational rational::multiply(const rational &r1, const rational &r2)
{
	rational mul;
	num = r1.num * r2.num;
	den = r1.den * r2.den;
	mul.num = num;
	mul.den = den;
	return mul;
}
rational rational::divide(const rational &r1, const rational &r2)
```

```
{
	rational div, temp;
	temp.num = r2.den;
	temp.den = r2.num;
	num = r1.num * temp.num;
	den = r1.den * temp.den;
	div.num = num;
	div.den = den;
	return div;
}
int rational::compare(const rational &r1, const rational&r2)
{
	rational diff;
	diff = subtract(r1, r2);
	if (diff.num == 0)
		return 0;
	else if (diff.num < 0)
		return -1;
	else if (diff.num > 0)
		return 1;
	return 0;
}
int rational::GCD() const
{
	int a = 0, b = 0;
	int remainder = num % den;
	while (remainder != 0)
	{
		a = den;
		b = remainder;
		remainder = a % b;
	}
	return b;
}
```




```
//             Course:  CS2400-60 Computer Science 2
//               Name:  Abdalkarim, Marina
//      Assignment:  Programming Assignment P6.2
//    Date assigned: 10/28/18
//          Date due: 11/27/17
//  Date handed in: 11/27/17
//            Remark: The program tests all functions.
#include <iostream>
#include <string>
using namespace std;
class rational
{
private:
        int num;        // num: numerator
        int den;        // denom: denominator
        int GCD() const;
        // You must use the Euclidean algorithm.
        // https://en.wikipedia.org/wiki/Euclidean_algorithm
        // Postcondition: returns the "greatest common divisor" between r.a and r.b
public:
        int helper();
```

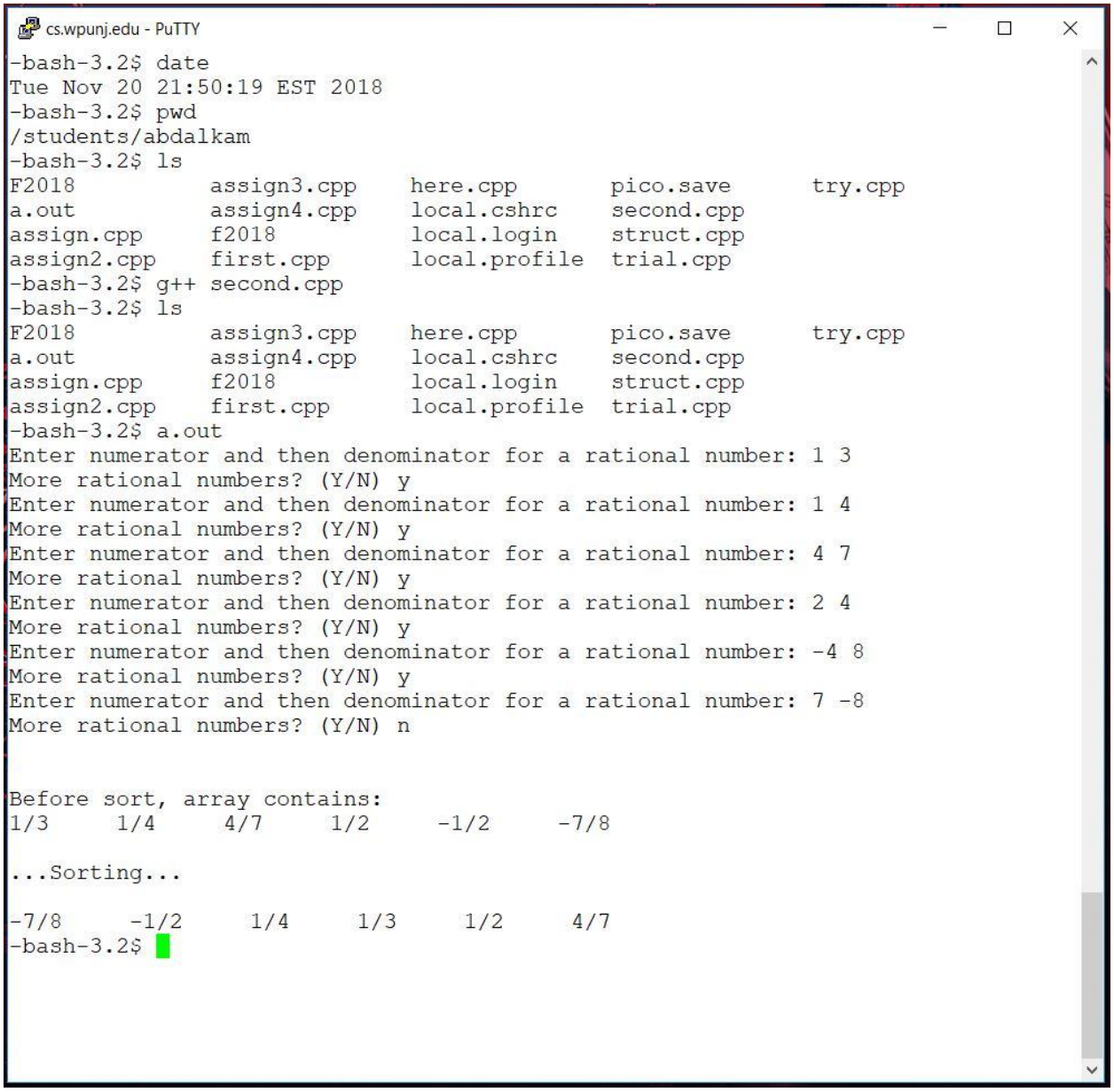
```
        // Another accessor function; gaining access to the gcd function
        // Postcondition: returns the greatest common divisor
        int getNum();
        // Another accessor function; gaining access to the value of the "num" data member of the
        // calling rational object
        // Postcondition: returns the name of the calling object
        int getDen();
        // Another accessor function; gaining access to the value of the "den" data member of the
        // calling rational object
        // Postcondition: returns the name of the calling object
        void setNum(int a);
        // Another accessor function; gaining access to the value of the "num" data member of the
        // calling rational object
        // Postcondition: initializes the calling objects
        void setDen(int a);
        // Another accessor function; gaining access to the value of the "den" data member of the
        // calling rational object
        // Postcondition: initializes the calling object
};
int fillArray(rational r[], int size);
// Precondition: address & physical size of the rational array declared in the calling function
// must be passed to the function
// Postcondition: returns the actual # of rational numbers entered by user which must be less than
// or equal to size
void displayArray(rational r[], int n);
// Postcondition: display n rational numbers
void selectionSort(rational r[], int n);
// Precondition:   rational array declared in the calling function and the # of elements to be sorted
// must be passed to the function
// Postcondition: n rational number is the array are sorted in ascending order
int findSmallestRationalNumber(rational r[], int first, int n);
// Precondition: accepts address, subscript value of the first element of the unsorted sub-list, and
// the # of array elements n
// Postcondition: returns subscript value of the smallest rational number in the unsorted sub-list
// of the array
void swap(rational &r1, rational &r2);
// Postcondition: contents of memory locations referenced by r1 and r2 are swapped
int main()
{
```

```
	const int SIZE = 6;
	rational s[SIZE];
	int fill, small = 0;
	fill = fillArray(s, SIZE);
	cout << endl << endl;
	cout << "Before sort, array contains: " << endl;
	displayArray(s, SIZE);
	cout << endl;
	selectionSort(s, SIZE);
	cout << endl;
	cout << "...Sorting..." << endl << endl;
	displayArray(s, SIZE);
	cout << endl;
	return 0;
}
int fillArray(rational r[], int size)
{
	char ask = 'y';
	int more = 1;
	int i = 0;
	do
	{
		int a, b;
		cout << "Enter numerator and then denominator for a rational number: ";
		cin >> a >> b;
		r[i].setNum(a);
		r[i].setDen(b);
		cout << "More rational numbers? (Y/N) ";
		cin >> ask;
		i++;
		more++;
	} while (ask == 'y' || ask == 'Y');
	return more;
}
void displayArray(rational r[], int n)
{
	for (int i = 0; i < n; i++)
	{
		int gcd = r[i].helper();
```

```
                int a = r[i].getNum() / gcd;
                int b = r[i].getDen() / gcd;
                if (b < 0)
                {
                        a = a * (-1);
                        b = b * (-1);
                }
                cout << a << "/" << b << "     ";
        }
}
void selectionSort(rational r[], int n)
{
        for (int pass = 1; pass < n; pass++)
        {
                for (int c = 0; c < n - pass; c++)
                {
                        double a, b, d, e;
                        a = r[c].getNum();
                        b = r[c].getDen();
                        d = r[c + 1].getNum();
                        e = r[c + 1].getDen();
                        double trial1, trial2;
                        trial1 = a / b;
                        trial2 = d / e;
                        if (trial1 > trial2)
                                swap(r[c], r[c + 1]);
                        int small;
                        small = findSmallestRationalNumber(r, c, n);
                        swap(r[0], r[small]);
                }
        }
}
int findSmallestRationalNumber(rational r[], int first, int n)
{
        rational small = r[first];
        for (int i = first + 1; i < n; i++)
        {
                double a, b, c, d, e, f;
                a = r[i].getNum();
```

```
                b = r[i].getDen();
                c = a / b;
                d = r[first].getNum();
                e = r[first].getDen();
                f = d / e;
                if (c < f)
                {
                        small = r[i];
                        first = i;
                }
                else
                {
                        small = r[first];
                        first = first;
                }
        }
        return first;
}
void swap(rational &r1, rational &r2)
{
        double temp1, temp2, a, b;
        temp1 = r1.getNum();
        a = r2.getNum();
        r1.setNum(a);
        r2.setNum(temp1);
        temp2 = r1.getDen();
        b = r2.getDen();
        r1.setDen(b);
        r2.setDen(temp2);
}
int rational::getNum()
{
        return num;
}
int rational::getDen()
{
        return den;
}
void rational::setNum(int a)
```

```
{
	num = a;
}
void rational::setDen(int a)
{
	den = a;
}
int rational::helper()
{
	int gcd;
	gcd = GCD();
	return gcd;
}
int rational::GCD() const
{
	int a = 0, b = 0;
	int remainder = num % den;
	while (remainder != 0)
	{
		a = den;
		b = remainder;
		remainder = a % b;
	}
	return b;
}
```


cs.wpunj.edu - PuTTY
-bash-3.2$ date
Tue Nov 20 21:50:19 EST 2018
-bash-3.2$ pwd
/students/abdalkam
-bash-3.2$ ls
F2018 assign3.cpp here.cpp pico.save try.cpp
a.out assign4.cpp local.cshrc second.cpp
assign.cpp f2018 local.login struct.cpp
assign2.cpp first.cpp local.profile trial.cpp
-bash-3.2$ g++ second.cpp
-bash-3.2$ ls
F2018 assign3.cpp here.cpp pico.save try.cpp
a.out assign4.cpp local.cshrc second.cpp
assign.cpp f2018 local.login struct.cpp
assign2.cpp first.cpp local.profile trial.cpp
-bash-3.2$ a.out
Enter numerator and then denominator for a rational number: 1 3
More rational numbers? (Y/N) y
Enter numerator and then denominator for a rational number: 1 4
More rational numbers? (Y/N) y
Enter numerator and then denominator for a rational number: 4 7
More rational numbers? (Y/N) y
Enter numerator and then denominator for a rational number: 2 4
More rational numbers? (Y/N) y
Enter numerator and then denominator for a rational number: -4 8
More rational numbers? (Y/N) y
Enter numerator and then denominator for a rational number: 7 -8
More rational numbers? (Y/N) n
Before sort, array contains:
1/3 1/4 4/7 1/2 -1/2 -7/8
...Sorting...
-7/8 -1/2 1/4 1/3 1/2 4/7
-bash-3.2$