Instructions on running commands:
-s: Print out all primes and identify whether or not they are interesting (Lucas, Mersenne, Fibonacci).
-p: Print out palindromic primes in bases 2, 9, 10, and the first letter of your last name + 10. Your personal base
-n <value>: Specifies the largest value to consider, inclusively, for your prime sieve. By default your program ru
For each prime number check if its a palindromic prime for the following bases:
–Base 2
–Base 9
–Base 10
–Base of the first letter of my last name + 10. 'z' + 10 = 36

pre-lab Part1.1:

```
bv_m = create(max)
bv_l = create(max)
bv_f = create(max)

i = 2
while 2^i - 1 <= (max):
        bv_set_bit(bv_m, 2^i - 1)
        i++

i = 2, j = 1
bv_set_bit(bv_l, i)
while j <= max:
        bv_set_bit(bv_l, j)
        i, j = j, i + j

i = 1, j = 1
bv_set_bit(bv_f, i)
while j <= max:
        bv_set(bv_f, j)
        i, j = j, i + j

for i in range(2 : max):
        if bv_get_bit(bv_p, i):
                print i, ": prime"
                if bv_get_bit(bv_m, i):
                        print ", mersenne"
                if bv_get_bit(bv_l, i):
                        print ", lucas"
                if bv_get_bit(bv_f, i):
                        print ", fibonacci"
                print "\n"
```

Pre-lab Part1.1 another version

```
listPrime[] for {2, 3, 5, ..., <= max}
generate listMersenne[] for {3, 7, 15, ..., <= max}
generate listLucas[] for {2, 1, 3, ..., <= max}
generate listFibonacci for {1, 2, 3, 5, ..., <= max}
for i in range(len(listPrime[i]):
        print i, ": prime"
        if isIn(listMersenne, i):
                print ", mersenne"
        if isIn(listLucas, i):
                print ", lucas"
        if isIn(listFibonacci, i):
                print ", fibonacci"
        print "\n"

def isIn(list, n):
        for i in range(len(list)):
                if list[i] == n:
                        return true
        return false
```

Pre-lab1.2 the question should have some wrong, it should be "a number in base is a palindrome prime"

```
remains = listPrime[n]
base = 10
i = 0
while remains > 0:
        digit = remains % base
        if  0 <= digit <= 9:
                str[i] = digit + '0'
        else
                str[i] = digit + 'a' - 10
        i++;
        remains /= base
if(isPalindrome(str, i)):
        return true
else
        return false
```

Pre-lab Part2.1
implementation as the source code

Pre-lab Part2.2
first, release the v->vector pointer, free the allocated memory for the vector in struct BitVector;

then, release the v pointer, free the allocated memory for the struct BitVector.

Pre-lab Part2.3

In the inner for-loop, most of the calls for bv_clr_bit() can be avoided if we add a if-statement, for example, when i = 2, the 4, 6, 8,... are all cleared, then all evens will be unnecessarily checked.

```
void sieve(BitVector *v){
	bv_set_all_bits(v);
	bv_clr_bit(v, 0);
	bv_clr_bit(v, 1);
	bv_set_bit(v, 2);
	for (uint32_t k = 0; (k+2)*2 <= bv_get_len(v); k++){
		bv_clr_bit(v, (k+i)*2);
	}

	for(uint32_t i = 3; i < sqrtl(bv_get_len(v)); i++){
		if(i % 2 != 0){
			if(bv_get_bit(v, i)){
				for (uint32_t k = 0; (k+i)*i <= bv_get_len(v); k++){
					if((k+i)*i % 2 != 0){
						bv_clr_bit(v, (k+i)*i);
					}
				}
			}
		}
	}
}
```