CS350 Assignment 1
Handling Bitmaps and Pointers in C.
Due Monday September 30, 2019 at 10PM

Write a C program called tripleprime.c . The program takes one command line argument "N" and computes a bitmap of primes less than or equal to N by using the sieve method. After the bitmap is constructed, print out the number of primes found, and then read in pairs of even integers K and M from stdin until eof is reached. For each pair use the bitmap to calculate the number of "triple primes" in the bitmap. For a Prime P a triple prime would be 3 primes (P, P+K, P+M). For example if K=2 amd M=6, (5,7,11) and (11, 13, 17) are triple primes. For each pair of inputs you print the number of triple primes and the largest one found. your output should be in a display similar to the demo program /home/cs350001/share/tripleprime

(1) The bitmap in fact will be represented as a linked list of segments each containing 8192 bits. Define a segment using the structure below, as well as a coordinate structure;

```
typedef struct _seg {
int bits[256];
struct _seg *next;
} seg;

typedef struct {
  seg *segpt;
  int intnum;
  int bitnum;
} coordinate;
```

See sample.c for a longer sample of a seg. You should dynamically allocate enough segments to represent all odd numbers <= N.

(2) Initially all bits should be set to 0. Use the sieve method for marking ON the bits corresponding to composite numbers less than or equal to N. (Note: The sieve method marks on multiples of primes, and is done when it finishes marking multiples of primes not exceeding sqrt(N)).

(3) When you complete building the bitmap, take one pass through it counting the 0 bits, and print out:
The number of odd primes less than or equal to N is: XXXX

(4) You MUST write the 9 functions:

```
seg *  whichseg (int j)  //calc ptr to seg containing bit for j
int  whichint (int j) // calc index of array containing bit for h
int  whichbit (int j) // calc 0-31 bit postion of bit for j

coordinate getcoord (int j) // returns coord using which funcs
void markcoord(coordinate c) // sets bit to 1 at coordinate
int testcoord(coordinate c) // returns 0 or 1 at coordinate

void  marknonprime (int j) // uses getcoord and markcoord
int  testprime (int j) // uses getcoord and testcoord

coordinate incrcoord (c coordinate, int inc) //increments coord by inc
int  whichnum (coordinate c); //compute the integer coresp to coord
```

(5) Whichseg is often called with increasing "j". It should save the last segment pointer as a starting point in case the next j is larger than the last.

(6) The sieve portion of the program should be written using testprime, and marknonprime. The triple prime portion should only use testcoord, and incrcoord.

(7) Note, N is NOT the number of segments required.

(8) Do not provide any prompts or require any user interaction beyond entering the numbers. Do not ask if the user is done, you must detect EOF. The input could be coming from a redirected file or the keyboard. The program needs not know which. We will discuss how EOF is indicated from the keyboard in class.