\*----------------------------------------------------------------------

\* Programmer: Arthur Flores

\* Class Account: masc0200

\* Assignment or Title: Assignment 3

\* Filename: prog3.s

\* Date completed: 11/25/14

\*----------------------------------------------------------------------

\* Problem statement: Print all prime numbers up to n

\* Input: value for n

\* Output: all prime numbers up to n

\* Error conditions tested: value of n greater than 1000 and valid YyNn \* to terminate

\* Included files: prog3.s

\* Method and/or pseudocode: java to pseudocode

\* References:

\*----------------------------------------------------------------------

\*

ORG $0

DC.L $3000 \* Stack pointer value after a reset

DC.L start \* Program counter value after a reset

ORG $3000 \* Start at location 3000 Hex

\*

\*----------------------------------------------------------------------

\*

#minclude /home/ma/cs237/bsvc/iomacs.s

#minclude /home/ma/cs237/bsvc/evtmacs.s

\*

\*----------------------------------------------------------------------

\*

\* Register use

\* D1=i

\* D2=j

\* D3=

\* D4=

\* D5=input

\* D6=input/2

\* D7=index

\* A1=array

\* A2=primes

\*

\*----------------------------------------------------------------------

\*

start: initIO \* Initialize (required for I/O)

setEVT \* Error handling routines

\* initF \* For floating point macros only

start2: lineout title \*prints titles

bra promp \*branch to prompt

newint: lineout invalid \*prints when invalid inputs

promp: lineout prompt \*prints prompt

linein buffer \*reads user input

stripp buffer,D0 \*stripp input length

cmp.l #4,D0 \*compare input length to 4

bgt newint \*if input length > for then branch

lea buffer,A4 \*load address of buffer

move.l D0,D5 \*move length of input into D5

subq.w #1,D5 \*subtract 1 from length

check: cmpi.b #'0',(A4) \*compare content in arry to ascii 0

blo newint \*branch if less than

cmpi.b #'9',(A4)+ \*cmp content to 9 then move point 1

bhi newint \*branch if higher then

dbra D5,check \*branch always to check D5 times

cvta2 buffer,D0 \*convert input into 2's complement

move.l D0,D5 \*move convert into D1

move.l #1000,D2 \*move 1000 into D2

cmp.l D2,D5 \*compare input to 1000

bgt newint \*input > 1000 then branch

move.l #2,D3 \*move 2 into D3

cmp.l D3,D5 \*compare input to 2

blt newint \*input < 2 then branch

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*(Flag primes)

move.l #2,D1 \*move 2 into i

move.l D5,D6 \*move input into D6

divu.w #2,D6 \*divide input by 2

ext.l D6 \*extend input/2

fori: cmp.w D6,D1 \*compare input to 2

bgt print \*branch greater them

lea array,A1 \*load address of array into A1

adda.l D1,A1 \*add i to address to point to chrt

tst.b (A1) \*test if content is 0

beq bottomi \*if 0 then branch to bottomi

move.w D1,D2 \*move i to j

asl.w #1,D2 \*j=i\*2

forj: cmp.w D5,D2 \*compare input to j

bgt bottomi \*if j is greater than input branch

lea array,A1 \*reset A1 to beggining of array

adda.l D2,A1 \*add j to address of array

clr.b (A1) \*clear byte at address A1

bottomj: add.w D1,D2 \*add i to j

bra forj \*branch to forj

bottomi: addq.w #1,D1 \*add 1 to i

bra fori \*branch to fori

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*(Print Primes)

print: move.l #2,D7 \*make index=2

lea array+2,A1 \*point A1 to begginning of array

lea primes,A2 \*load primes into A2

move.b #$0A,(A2)+ \*go to next line

move.b #$0D,(A2)+ \*carriage return

cont\_print:

cmp.w D5,D7 \*compare index to input

bgt done2 \*if greater then go to done2

tst.b (A1) \*test contents of A1

beq bottom \*if 0 then go to bottom

move.l D7,D0 \*move index into D0

ext.l D0 \*extend D0

cvt2a (A2),#3 \*convert D0 to ascii in A2

stripp (A2),#3 \*stripp ascii charcter

adda.l D0,A2 \*add length of string to address

move.b #' ',(A2)+ \*add a space in memory then move

bottom: addq.l #1,D7 \*add 1 to index

addq.l #1,A1 \*add 1 to A1 address

bra cont\_print \*branch to cont\_print

done2: clr.b (A2) \*null terminate

lineout answer \*print answer

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*(Ask for new primes)

bra valid \*branch to valid

ivalid: lineout invalid \*print to ask for new input

valid: lineout prompt2 \*ask for inpuit

linein buffer \*read output

cmp.w #1,D0 \*compare length of input to 1

bgt ivalid \*branch if length > 1

ori.b #$20,buffer \*change upperc letter to lowerc

cmpi.b #'y',buffer \*compare input to y

beq promp \*if y then branch to promp

cmpi.b #'n',buffer \*compare input to N

beq done3 \*if n then branch to done3

bne ivalid \*if input != Y,y,N,n branch

done3: lineout end \*print end

break \* Terminate execution

\*

\*----------------------------------------------------------------------

\* Storage declarations

title: dc.b 'Program #3, Arthur Flores, masc0200',0

prompt: dc.b 'Enter a value for n',0

buffer: ds.b 81

array: dcb.b 1001,1

answer: dc.b 'The prime numbers in this range are: '

primes: ds.b 800

prompt2: dc.b 'Do you want to find more primes? (Y/N)',0

end: dc.b 'The program has terminated.',0

invalid: dc.b 'Please enter a valid entry.',0

end