Lab 10 bonus

```
; Greg Violan, 011706641
; Andy Sien 012064856
: Lab 10 bonus
;variable and constant definitions
keyBytesRAMaddress EQU 0x40; symbolic constant for base address of
; encryption key in RAM
keyLength EQU 0x30; variable to track length of key
keyvalIndex EQU 0xe0; variable to index the keyval constant array
; keyvalIndex is also an alias for accumulator
chIndex EQU 0xe0 ;alias variable for Accumulator
;begin section from lab 9
imp main ;jump past interrupt vector table
org 0x0030 ;put main program at rom location 0x0030
main:
      mov tmod, #0x21
                           ; configure timer 0, mode 1
      mov scon, #0x50
                                  ; config serial 8-data, 1 start, 1 stop, no parity
      mov th1, #0xFD
                                  : 9600 baud
      setb tr1
                                  ; start timer 1 to enable serial communication
      mov dptr, #Prompt ; initialize ROM pointer
      Call displayChar
                          ; call displayChar
      mov r0, keyBytesRAMaddress
                                         ;initialize RAM pointer
main_loop2:
      inb ri, $
                                         ; wait to receive a char
                                 ; char received, get it!
      call getchar
      clr ri
                                        ; acknowledge char receive
      mov @r0, a
                                        ; move key into RAM
      inc r0
                                         ; increment RAM pointer
      cine a, #0x00, main loop2; check for null character in string
      mov dptr, #keyFile
                                 ; initialize ROM pointer
      Call displayChar
                                 ; call displayChar
;-----;
;keyval variable no longer used
; mov keyval, #0x23 ;load the keyval variable with encryption key
mov tmod, #0x20 ;config timer 1 mode 2
mov scon, #0x50 ;config serial 8-data, 1 start, 1 stop, no parity
```

```
mov th1, #0xFD ;9600 baud
setb tr1; start timer 1 to enable serial communication
;end section from lab 9
;In the following section load the key bytes from ROM into RAM
mov r0, #keyBytesRAMaddress; initialize RAM pointer
mov dptr, #keyvals2 ;initialize ROM pointer
mov keyvalIndex, #0x00 ;initialize keyvalIndex
LoadKey:
push keyvallndex ;preserve keyvallndex variable
movc a,@a+dptr;load byte of key from ROM
cine a, #0x00, notNull ;check for null terminating character
imp LoadDone; if null is found, enter main loop
notNull:
mov @r0, a ;put byte of key into ram
pop keyvallndex;restore keyvallndex variable
inc keyvalIndex;increment keyvalIndex
inc r0;increment RAM pointer
jmp LoadKey;continue the loop
LoadDone:
mov @r0, #0x00 ;append null char to string
mov r0, #keyBytesRAMaddress;re-initialize RAM pointer
;-----: END of Initialization/configuration -----;
;begin section from lab 9
;vvvvvvvvvvvvvvvvvvvvvvvvvvvv
mainloop:
inb ri, $; wait to receive a char
call getchar ;char received, get it!
; cjne a, #0x00, encrypt ;check for null character in string
cjne a, #0x00, checkKeyVal ;check for null character in string
imp terminate ;terminate program if null character is recieved
;end section from lab 9
checkKeyVal:
cine @r0, #0x00, Encrypt; go to Encrypt if keyVal is not null
mov r0, #keyBytesRAMaddress ;re-initialize RAM pointer
;begin section from lab 9
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
Encrypt:
xrl a, @r0 ;encrypt the character contained in the accumulator
call writechar; write the encrypted character
inc R0
jmp mainloop
```

```
terminate:
mov a, #0x00 ;load null character into accumulator
call writechar ;append the null character to text output
sjmp $ ;halt
;----- getchar -----;
;subroutine receives nothing before it is called
;writes the character to the serial console
;returns a byte in the accumulator
getchar:
mov a, sbuf ;get serial data (char)
clr ri ;acknowledge data received
ret ;return from subroutine call
;----- writechar -----;
;receives byte or character
;reads a character that has been received serially
:returns the c
writechar:
mov sbuf, a ;send data (char) serially
jnb ti, $; wait until data is sent
clr ti ;acknowledge data has been sent
ret ;return from subroutine call
;end section from lab 9
; BONUS
displayChar:
       mov chlndex, #0x00
                                   ; put char in serial buffer
       loopBack:
              push chindex
                                   ; preserve accumulator
              movc a, @a + DPTR; load byte of key from ROM
              inz not0
                                           ; jump to not0 if accumulator != 0
              pop chlndex
                                           ; restore accumulator
              ret
writeCharBonus:
       mov sbuf, a
                                           ; send data (char) serially
                                   ; wait until data is sent
      inb ti, $
       clr ti
                                           ; acknowledge data sent
       ret
not0:
       Call writeCharBonus
                                   : if accumulator is not nul call writeCharBonus
       pop chlndex
                                           ; restore accumulator
```

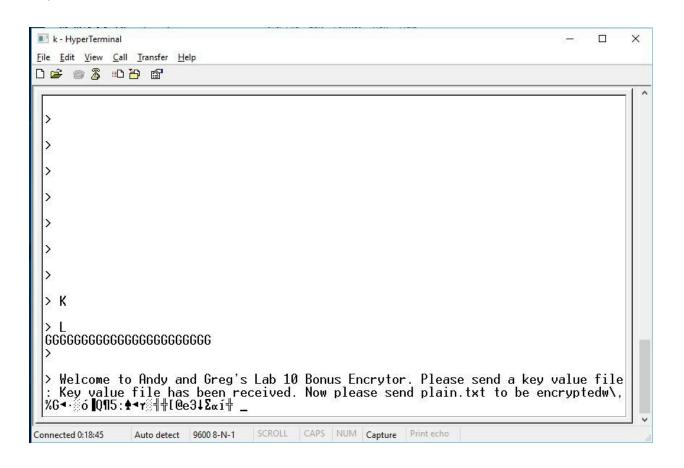
inc chIndex ; increment accumulator jmp loopBack ; jump back to loopBack

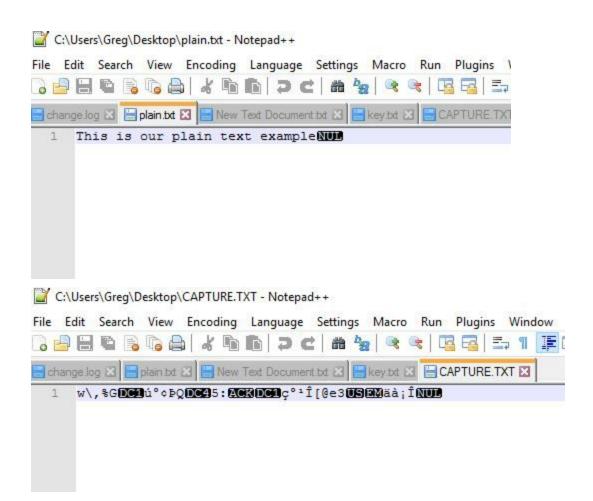
;multibyte keys are defined below, only one will be used at a time org 0x200

Prompt: db "Welcome to Andy and Greg's Lab 10 Bonus Encrytor. Please send a key value file: ",0

keyFile: db "Key value file has been received. Now please send plain.txt to be encrypted",0 keyvals: db '12345678',0

keyvals2: db 0x23, 0x34, 0x45, 0x56, 0x67, 0x78,0x89, 0x90, 0xCD, 0xAB, 0x00 DutyValues: DB 0x60, 0x70, 0x80, 0x90, 0xA0, 0xB0, 0xC0, 0xD0, 0xE0, 0xF0 End





aintx+) This is our plain text example Ascii Hex reg value

T > 0x54 0x23 NOR 0010 0011 0277 N 20x68 0x34 NOR 0110 1000 01011100 0x5C 1 > 0x69 0x45 XOR 0100 0107 0010 1100 8×2C 5-20x73 0x56 0111 0011 0107 0110 0010 0101 Upher Hox S Ascit 0×77 0×50 0×20 By 25 This materies our cipher

