JAI SHREE RAM

write SIC or SIC/XE pgm to implement linear search...

----------SIC Program for Linear Search----------

```
START  1000          ; Program start address
      LDX    #0      ; Initialize index register to 0
      LDA    LENGTH  ; Load the length of the array
LOOP   COMP   X,LENGTH ; Compare index with the length of the array
      JEQ    NOTFOUND ; If index equals length, key not found
      LDA    ARRAY,X  ; Load array element at index X
      COMP   KEY      ; Compare array element with the key
      JEQ    FOUND    ; If equal, go to FOUND
      TIX             ; Increment index (X = X + 1)
      J    LOOP       ; Repeat the loop
FOUND  STA    RESULT   ; Store the found position in RESULT
      J     END       ; End the program
NOTFOUND  LDA  #-1     ; Load -1 to indicate "not found"
        STA  RESULT  ; Store the "not found" flag in RESULT
END    HLT             ; Halt the program
LENGTH WORD  5         ; Length of the array
KEY    WORD  30        ; Key to search for
ARRAY  WORD  10, 20, 30, 40, 50 ; Array elements
RESULT RESW  1         ; To store the result
      END   START     ; End of the program
```

----------SIC/XE Program for Linear Search----------

```
START  1000          ; Program start address
      LDX    #0      ; Initialize index register to 0
```

```
    LDA    LENGTH  ; Load the length of the array
LOOP   COMP   X,LENGTH ; Compare index with the length of the array
    JEQ    NOTFOUND ; If index equals length, key not found
    LDB    #3      ; Use base-relative addressing
    LDA    ARRAY,X  ; Load array element at index X using displacement
    COMP   KEY     ; Compare array element with the key
    JEQ    FOUND   ; If equal, go to FOUND
    TIXR   X       ; Increment index register (X = X + 1)
    J     LOOP     ; Repeat the loop
FOUND  STA    RESULT   ; Store the found position in RESULT
    J     END      ; End the program
NOTFOUND  LDA  #-1    ; Load -1 to indicate "not found"
       STA  RESULT  ; Store the "not found" flag in RESULT
END    HLT           ; Halt the program
LENGTH WORD  5        ; Length of the array
KEY    WORD  30       ; Key to search for
ARRAY  WORD  10, 20, 30, 40, 50 ; Array elements
RESULT RESW  1        ; To store the result
    END   START    ; End of the program
```

---

pgm to devide BETA by GAMMA setting integer portion of a quotient in ALPHA and DELTA to remainder use register

to register instr to make the calculation is efficient

```
LDA     BETA
LDS     GAMMA
DIVR    S,A
STA     ALPHA
MULR    S,A
```

```
LDS     BETA
SUBR    A,S
STS     DELTA
ALPHA   RESW 1
```

SIC/XE MACHINE TO CLEAR 20 BYTE string to all blanks use immediate addressing mode reg to reg instr

```
LDT #20
LDX #0
LDCH #0
STCH STR1,X
TIXR T
JLT LOOP
STR1 RESB 20
```

Problem: Implement a Binary Search to find a key in a sorted array of 50 elements.

```
START 1000
    LDA #0          ; Low = 0
    STA LOW
    LDA LENGTH
    SUB #1          ; High = LENGTH - 1
    STA HIGH
LOOP   LDA LOW
```

```
    ADD HIGH
    DIV #2          ; Mid = (Low + High) / 2
    STA MID
    LDA ARRAY,MID    ; Load array[MID]
    COMP KEY
    JEQ FOUND       ; If key found, go to FOUND
    JLT LOWER       ; If key < array[MID], search lower half
    JGT UPPER       ; If key > array[MID], search upper half
LOWER  LDA MID
    SUB #1          ; High = Mid - 1
    STA HIGH
    J LOOP
UPPER  LDA MID
    ADD #1          ; Low = Mid + 1
    STA LOW
    J LOOP
FOUND  STA RESULT      ; Store the found index in RESULT
    J END
NOTFOUND LDA #-1       ; If not found, store -1 in RESULT
       STA RESULT
END    HLT
ARRAY  WORD 10, 20, 30, 40, 50
LENGTH WORD 5
KEY    WORD 30
LOW    RESW 1
HIGH   RESW 1
MID    RESW 1
RESULT RESW 1
    END START
```

To find max element from 100 words array and store it variable called max

```
LDS #3
LDT #300
LDX #0
MOVE LDA ALPHA,X
COMP MAX
JLT NOCHANGE
STA MAX
NOCHANGE ADDR S, X
COMP X, T
JLT MOVE
ALPHA RESW 100
MAX WORD -32565
```

Problem: Perform Sequential Search where each element is checked one by one.

```
START 1000
    LDX #0          ; Start index
SEQLP  LDA ARRAY,X
    COMP KEY
    JEQ FOUND
    TIXR X
    LDA LENGTH
    COMP X
    JLT SEQLP       ; Continue if index < LENGTH
    J NOTFOUND
FOUND  STA RESULT
    J END
NOTFOUND LDA #-1
```

```
        STA RESULT
END     HLT
ARRAY   WORD 15, 25, 35, 45, 55
LENGTH WORD 5
KEY     WORD 25
RESULT RESW 1
      END START
```

------write a set of SIC/XE instructions a MAX element from 100 words array and store it in variable called MAX.

```
START  1000          ; Program start address
      LDX    #0      ; Initialize index register X to 0
      LDA    ARRAY   ; Load the first element of the array into A
      STA    MAX     ; Assume the first element is the maximum
      LDB    #100    ; Load the array size (100 words) into B (used for counting)
LOOP   TIXR   X       ; Increment index register X
      JEQ    DONE    ; If X equals B (array size), we're done
      LDA    ARRAY,X ; Load the current array element using displacement
      COMP   MAX     ; Compare it with the current MAX value
      JLT    NEXT    ; If current element is less than MAX, skip
      STA    MAX     ; Otherwise, store the new maximum in MAX
NEXT   J      LOOP     ; Repeat the loop
DONE   HLT            ; Halt the program
ARRAY  RESW   100      ; Reserve space for a 100-word array
MAX    RESW   1        ; Reserve space for the maximum value
      END    START    ; End of the program
```

To arrange n elements in ascending order

```
SORT START 0
OUTER LDA INDEX
LDX #0
LDS ARR1, X
LDX #1
INNER LDT ARR1, X
COMP S, T
JLT LOOP
JEQ LOOP
RMO S,A
RMO T, S
RMO A, T
STA ARRI, X
LOOP RMO X,A
ADD #3
COMP LENGTH
JLT INNER
RMO A,X
JLT INNER
LDA INDEX
ADD #3
JLT OUTER
ARR1 RESW 10
LENGTH WORD 30
INDEX WORD 0
END
```

---Problem: Find the maximum value from an array of 100 elements and store it in a variable.--------

```
START 1000
    LDX #0
    LDA ARRAY
    STA MAX
    LDB LENGTH
LOOP   TIXR X
    JEQ DONE
    LDA ARRAY,X
    COMP MAX
    JLT NEXT
    STA MAX
NEXT   J LOOP
DONE   HLT
ARRAY  RESW 100
LENGTH WORD 100
MAX    RESW 1
    END START
```

Problem: Write a SIC/XE program to sort an array of 50 elements in ascending order.

```
START 1000
    LDX #0
SORT   LDA ARRAY,X
    TIXR X
    JEQ END
    COMP ARRAY,X
    JGT SWAP
    J SORT
SWAP   LDA ARRAY,X
    STA TEMP
```

```
    LDA ARRAY
    STA ARRAY,X
    LDA TEMP
    STA ARRAY
    J SORT
END   HLT
ARRAY  RESW 50
TEMP   RESW 1
    END START
```

Problem: Compute the factorial of a number stored in memory.

```
START 1000
    LDA NUMBER
    STA FACT
    LDX #1
FACTOR TIXR X
    JEQ DONE
    MUL FACT
    STA FACT
    J FACTOR
DONE   HLT
NUMBER WORD 5
FACT   RESW 1
    END START
```

Problem: Write a SIC/XE program to reverse a string of length 20.

```
START 1000
```

```
    LDX #19
    LDCH STRING,X
    STCH REVERSE,19-X
    TIXR X
    JGT START
DONE   HLT
STRING BYTE C'HELLO'
REVERSE RESB 20
    END START
```

Problem: Implement a SIC/XE bootstrap loader to load a program from a specified address.

```
START 1000
    LDA #BUFFER
    LDX #0
LOOP   RD INPUT
    JEQ DONE
    STCH BUFFER,X
    TIXR X
    J LOOP
DONE   HLT
BUFFER RESB 100
    END START
```
