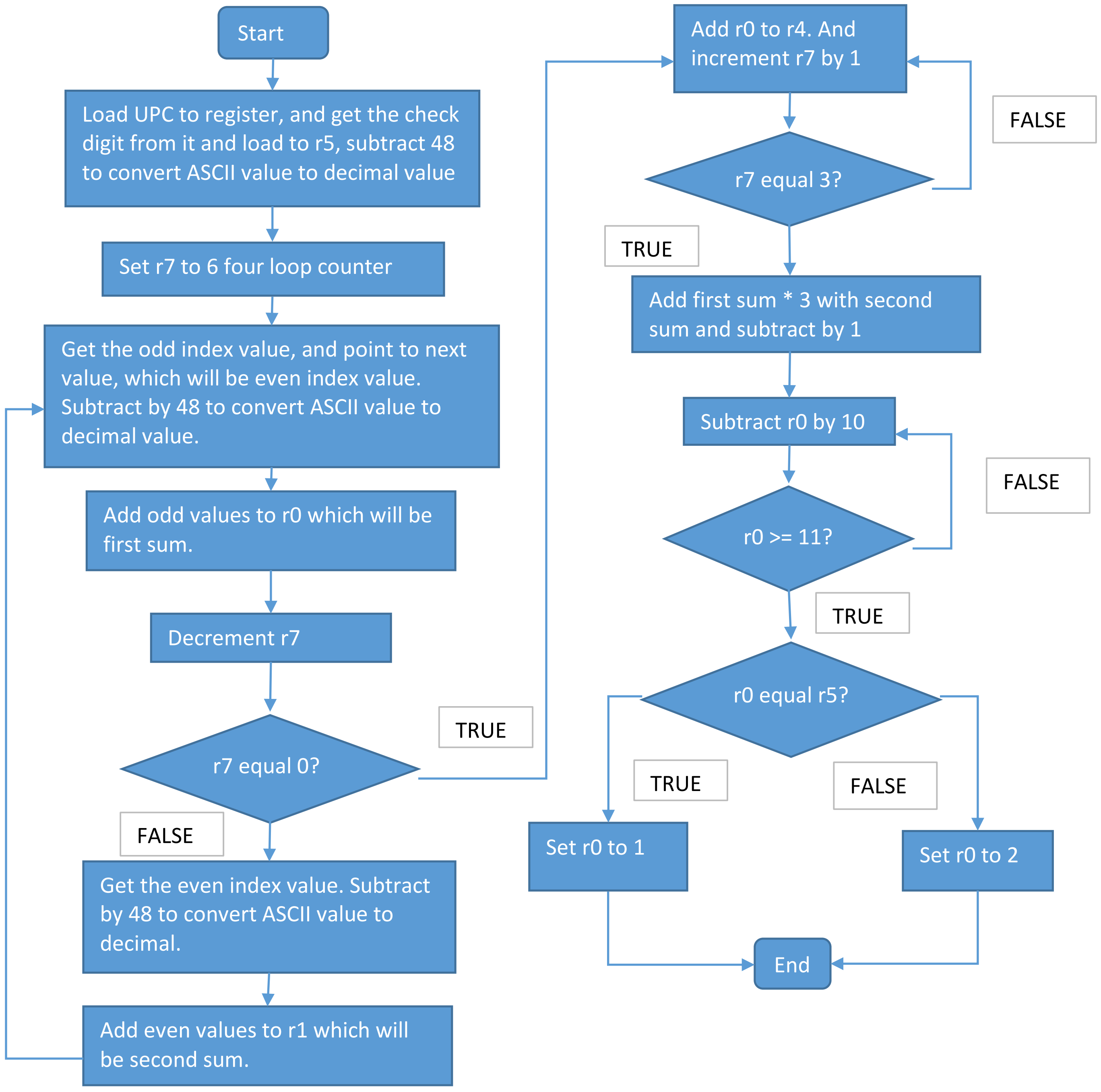
Assignment 3 Question 1 Flow Chart

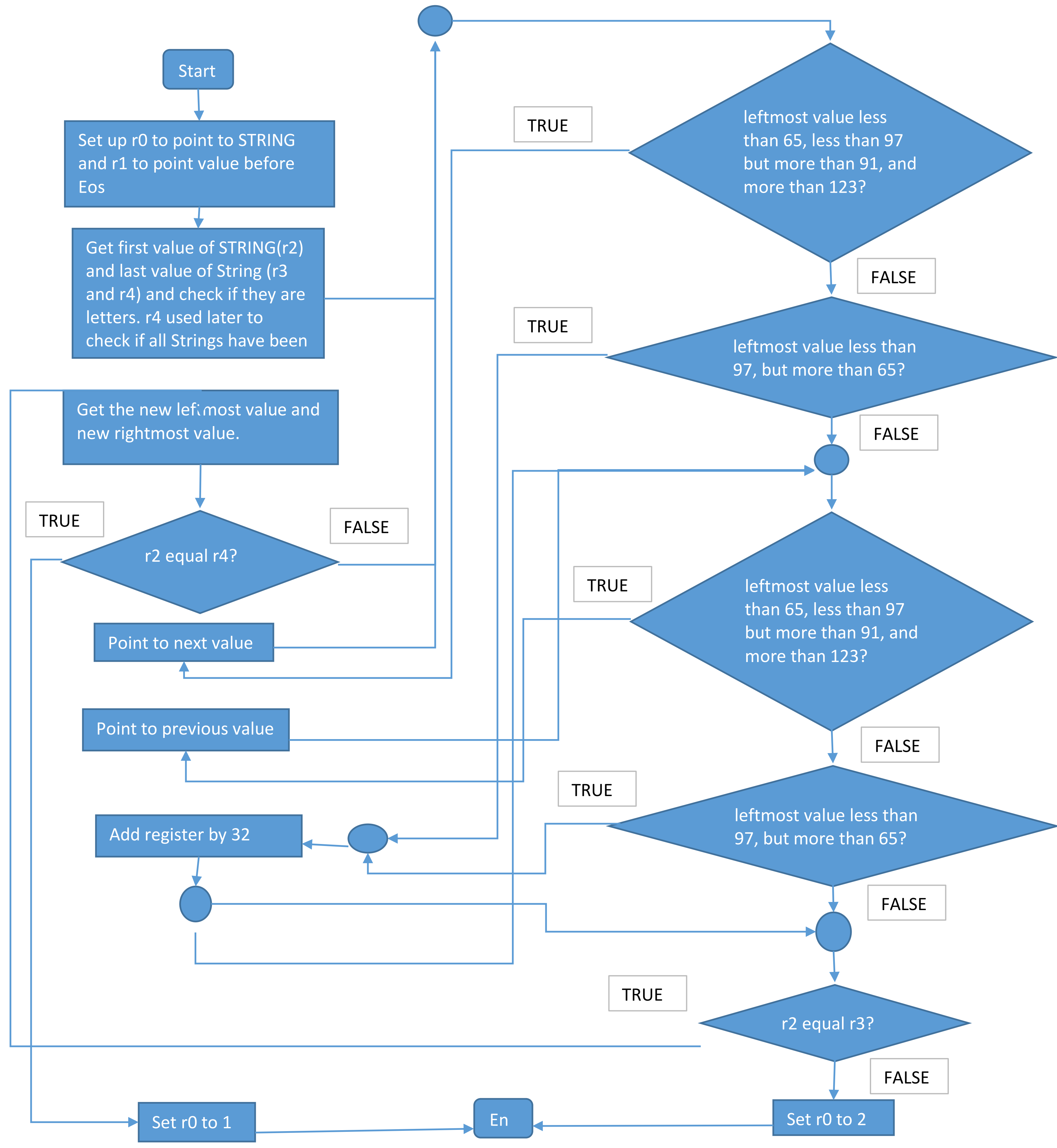

Start
Load UPC to register, and get the check digit from it and load to r5, subtract 48 to convert ASCII value to decimal value
Set r7 to 6 four loop counter
Get the odd index value, and point to next value, which will be even index value. Subtract by 48 to convert ASCII value to decimal value.
Add odd values to r0 which will be first sum.
Decrement r7
r7 equal 0?
TRUE
FALSE
Get the even index value. Subtract by 48 to convert ASCII value to decimal.
Add even values to r1 which will be second sum.
Add r0 to r4. And increment r7 by 1
r7 equal 3?
FALSE
TRUE
Add first sum * 3 with second sum and subtract by 1
Subtract r0 by 10
r0 >= 11?
FALSE
TRUE
r0 equal r5?
TRUE
FALSE
Set r0 to 1
Set r0 to 2
End

Assignment 3 Question 1 Instructions

```
          AREA UPC_Validation, CODE, READONLY
          ENTRY

          LDR r6, =UPC          ;    Load r6 with content at location
                                ;    UPC to access the
          LDRB r5, [r6, #11]    ;    Load r5 with 11th value from UPC
                                ;    String for check digit
          SUB r5, #48           ;    To convert ASCII code to decimal
                                ;    value, subtract by 48
          MOV r7, #6            ;    Set up for loop counter



SUMS      LDRB r2, [r6], #1     ;    Load r2 with odd value of UPC
                                ;    String, then go to next value
                                ;    which will be even value
          SUB r2, #48           ;    To convert ASCII code to decimal
                                ;    value, subtract by 48
          ADD r0, r2            ;    Add r2 to r0, which is going to
                                ;    be sum of all odd index of UPC
                                ;    String
          SUB r7, #1            ;    Decrement loop counter
          CMP r7, #0            ;    Performs test to end loop
          BEQ MULT3             ;    When r7 do equal zero, multiply
                                ;    first sum by 3
          LDRB r3, [r6], #1     ;    Load r3 with even value of UPC
                                ;    String, then ggo to next value
                                ;    which will be odd value
          SUB r3, #48           ;    To convert ASCII code to decimal
                                ;    value, subtract by 48
          ADD r1, r3            ;    Add r3 to r1, which is going to
```

```
                                ;     be sum of all even index of UPC
                                ;     String
           B SUMS               ;     Loops again, continue until r7
                                ;     equals zero



MULT3      ADD r4, r0           ;     Add r0 to r4, which is going to
                                ;     be first sum times 3.
           ADD r7, #1           ;     Increment loop counter
           CMP r7, #3           ;     Performs test to end loop
           BNE MULT3            ;     Continue until count equals 3



           ADD r0, r4, r1       ;     Add two numbers (first sum * 3 +
                                ;     second sum)
           SUB r0, #1           ;     Subtract 1 from total



REMAINDER  SUB r0, #10          ;     Subtract 10 until r0 results in
                                ;     remainder
           CMP r0, #11          ;     Performm test at end of loop
           BPL REMAINDER        ;     Continue until r0 will have
                                ;     value less than 10 which will be
                                ;     the remainder



           RSB r0, r0, #9       ;     Subtract 9 from r0 to get the
                                ;     check digit
           CMP r0, r5           ;     Perform test to see if check
                                ;     digit matches the calculation
TRUE       MOVEQ r0, #1         ;     TRUE IF values equal, which
                                ;     stores 1 in r0
FALSE      MOVNE r0, #2         ;     ELSE FALSE, which stores 2 in r0



UPC        DCB "013800150738"        ;     UPC string

           END
```

Assignment 3 Question 2 Flow Chart


Start
Set up r0 to point to STRING and r1 to point value before Eos
Get first value of STRING(r2) and last value of String (r3 and r4) and check if they are letters. r4 used later to check if all Strings have been
Get the new leftmost value and new rightmost value.
TRUE
r2 equal r4?
FALSE
Point to next value
Point to previous value
Add register by 32
Set r0 to 1
En
Set r0 to 2
TRUE
leftmost value less than 65, less than 97 but more than 91, and more than 123?
FALSE
TRUE
leftmost value less than 97, but more than 65?
FALSE
TRUE
leftmost value less than 65, less than 97 but more than 91, and more than 123?
FALSE
TRUE
leftmost value less than 97, but more than 65?
FALSE
TRUE
r2 equal r3?
FALSE

Assignment 3 Question 2 Instructions

```
           AREA Palindrome, CODE, READONLY
           ENTRY

           ADR r0, STRING        ;     Set r0 up to point STRING
           ADR r1, EoS - 1       ;     Set up r1 to point value before
                                 ;     EoS




           LDRB r2,[r0]          ;     Load r2 with first value of
                                 ;     STRING
           LDRB r3, [r1]         ;     Load r3 with last value of
                                 ;     STRING
           LDRB r4, [r1]         ;     Load r4 with last value of
                                 ;     STRING for validation later



           B CHECKLM             ;     Go to CHECKLM, to check if
                                 ;     leftmost value is character



PASSED     LDRB r2, [r0, 1]!     ;     Load r2 with next value of
                                 ;     STRING from what was used before
           LDRB r3, [r1, -1]!    ;     Load r3 with previous value of
                                 ;     STRING from what was used before
           CMP r2, r4            ;     Performs test to end program
           BEQ TRUE              ;     IF values equal, then go to TRUE
           B CHECKLM             ;     Go to CHECKLM, to check if
                                 ;     leftmost value is letter
```

```
NEXTCHAR   LDRB r2, [r0, 1]!      ;     Load r2 with next value of
                                  ;     STRING from what was used before
                                  ;     which was not a letter
           B CHECKLM              ;     Go to CHECKLM, to check if
                                  ;     leftmost value is letter



PREVCHAR   LDRB r3, [r1, -1]!     ;     Load r3 with previous value of
                                  ;     STRING from what was used before
                                  ;     which was not a letter
           B CHECKRM              ;     Go to CHECKRM, to check if
                                  ;     rightmost value is letter



TOLOWER1   ADD r2, #32            ;     Convert Captial letter to lower
                                  ;     case letter by adding 32
           B CONT1                ;     Continue after CHECKLM



TOLOWER2   ADD r3, #32            ;     Convert Captial letter to lower
                                  ;     case letter by adding 32
           B CONT2                ;     Continue after CHECKRM



CHECKLM    CMP r2, #65            ;     Performs test to see if r2 is
```

```
                                ;     letter of not
          BCC NEXTCHAR          ;     Go to NEXTCHAR to get next value
                                ;     since current value is not a
                                ;     letter
          CMP r2, #91           ;     Performs test to see if r2 is
                                ;     capital letter or not
          BCC TOLOWER1          ;     Go to TOLOWER1 to get lower case
                                ;     letter of current, and since
                                ;     value is between 65 and 91, we
                                ;     know it is letter
          CMP r2, #97           ;     Performs test to see if r2 is
                                ;     letter of not
          BCC NEXTCHAR          ;     Go to NEXTCHAR to get next value
                                ;     since current value is not a
                                ;     letter
          CMP r2, #123          ;     Performs test to see if r2 is
                                ;     letter of not
          BPL NEXTCHAR          ;     Go to NEXTCHAR to get next value
                                ;     since current value is not a
                                ;     letter


CONT1
CHECKRM   CMP r3, #65           ;     Performs test to see if r3 is
                                ;     letter of not
          BCC PREVCHAR          ;     Go to PREVCHAR to get previous
                                ;     value since current value is not
                                ;     a letter
          CMP r3, #91           ;     Performs test to see if r3 is
                                ;     capital letter or not
          BCC TOLOWER2          ;     Go to TOLOWER2 to get lower case
                                ;     letter of current, and since
                                ;     value is between 65 and 91, we
                                ;     know it is letter
           CMP r3, #97          ;     Performs test to see if r3 is
                                ;     letter of not
           BCC PREVCHAR         ;     Go to PREVCHAR to get previous
                                ;     value since current value is not
                                ;     a letter
           CMP r3, #123         ;     Performs test to see if r3 is
                                ;     letter of not
           BPL PREVCHAR         ;     Go to PREVCHAR to get previous
                                ;     value since current value is not
                                ;     a letter


CONT2
CHECKPELIN CMP r2, r3           ;     Performs test to see if leftmost
                                ;     letter and rightmost letter is
                                ;     equal
           BEQ PASSED           ;     IF they are equal, then go to
                                ;     PASSED, and keep checking if
                                ;     letters are equal
FALSE      MOV r0, #2           ;     FALSE IF they aren't, and store
                                ;     r0 with 2
           B DONE               ;     Jump of TRUE
TRUE       MOV r0, #1           ;     IF pelindrome, then store r0
                                ;     with 1
DONE                            ;     End of program



STRING     DCB "He lived as a devil, eh?"        ;string
EoS        DCB 0x00                              ;end of string
```

END