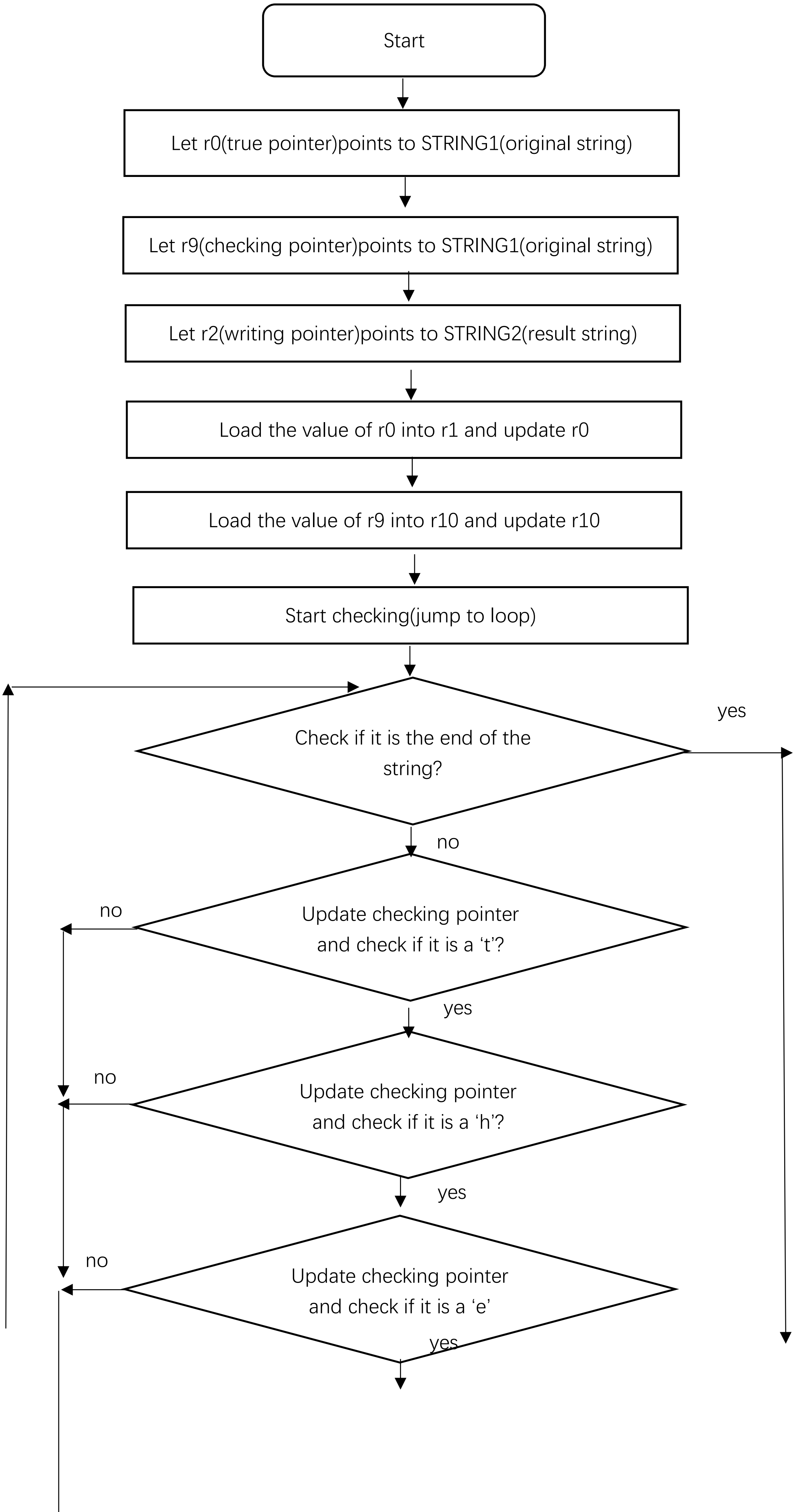
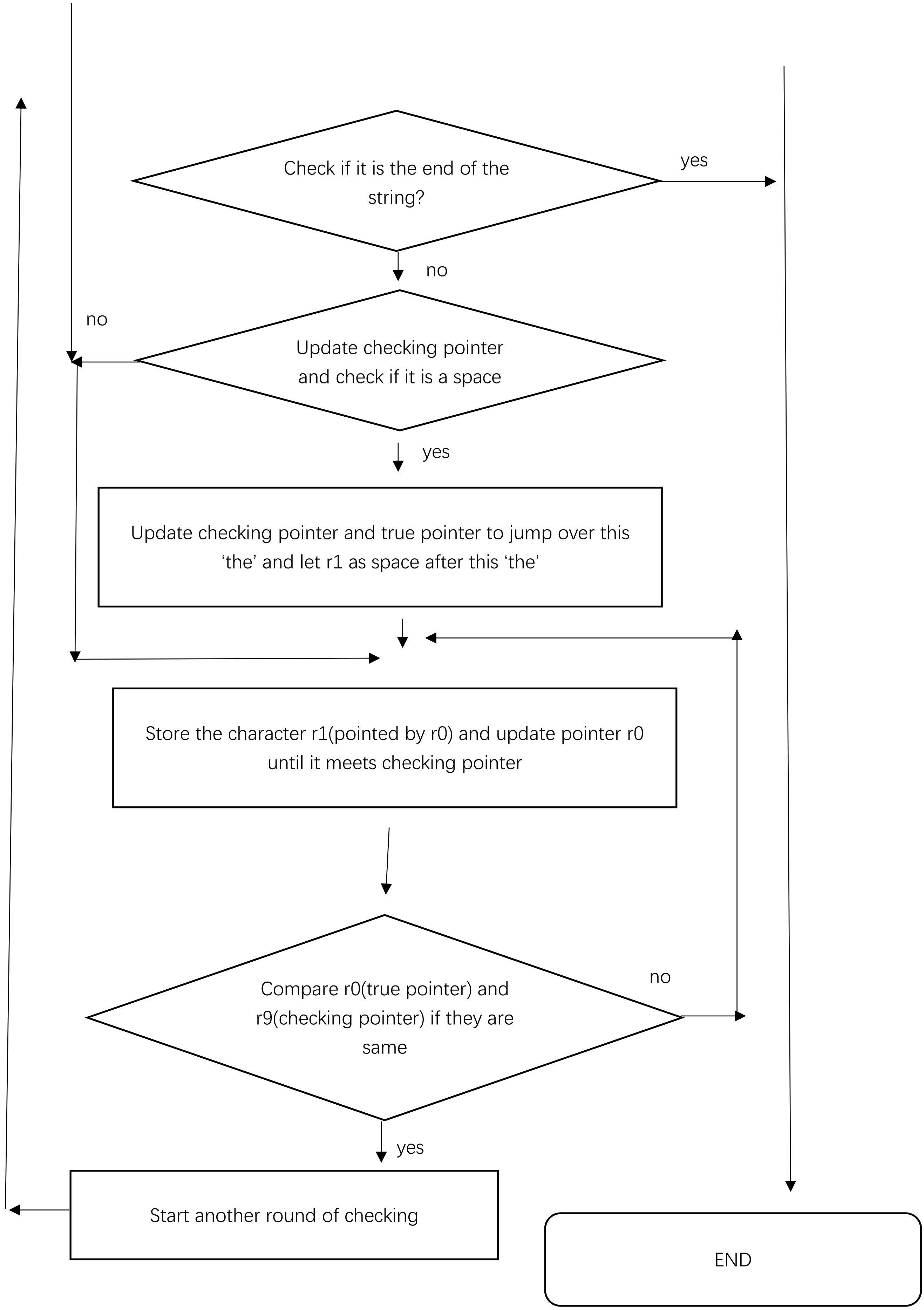

Start
Let r0(true pointer)points to STRING1(original string)
Let r9(checking pointer)points to STRING1(original string)
Let r2(writing pointer)points to STRING2(result string)
Load the value of r0 into r1 and update r0
Load the value of r9 into r10 and update r10
Start checking(jump to loop)
Check if it is the end of the string?
yes
no
Update checking pointer and check if it is a 't'?
no
yes
Update checking pointer and check if it is a 'h'?
no
yes
Update checking pointer and check if it is a 'e'
no
yes


Check if it is the end of the string?
yes
no
no
Update checking pointer and check if it is a space
yes
Update checking pointer and true pointer to jump over this ‘the’ and let r1 as space after this ‘the’
Store the character r1(pointed by r0) and update pointer r0 until it meets checking pointer
Compare r0(true pointer) and r9(checking pointer) if they are same
no
yes
Start another round of checking
END

```
            AREA assignment2Q1, CODE,READONLY
            ENTRY
            ADR r0,STRING1          ;let r0(true pointer) points to STRING1
            ADR r9,STRING1          ;let r9(checking pointer) points to STRING1
            ADR r2,STRING2          ;let r2(store pointer) points to STRING1
            LDRB r1,[r0],#1         ;load the value of r0 into r1 and update automatically
            LDRB r10,[r9],#1    ;load the value of r0 into r1 and update automatically
            B LOOP                  ;jump to loop to start checking
    WRITE   CMP r0,r9               ;if r0 is not in the same position as r9, keep writing
            BEQ LOOP                ;if r0 is in the same position as r9, continue checking
            STRBNE r1,[r2],#1  ;store r1 in the address pointed by r2 and update r2
            LDRBNE r1,[r0],#1 ;keep loading and writing until r0 meets r9
            BNE WRITE               ;keep looping until r0 meets r9
    LOOP    CMP r10,#0x00           ;check if it is the end of string
            STRBEQ r10,[r2],#1      ;if it is the end of string,load 0x00 to show the program
ends
            BEQ EXIT            ;EXIT if we've end checking
            CMP r10,#0x74           ;check if it is a word starting as 't'
            LDRB r10,[r9],#1    ;update checking pointer and register(r10)
            BNE WRITE               ;if it is not a 't', write into memory
            CMP r10,#0x68           ;if it is a 't', then check if the second one is   a 'h'
            LDRB r10,[r9],#1    ;update checking pointer and register
            BNE WRITE               ;if not, write these two letters
            CMP r10,#0x65           ;if it starts as 'th', then check if the third one is a 'e'
            LDRB r10,[r9],#1    ;update checking pointer and register
            BNE WRITE               ;if not, write these three letters
            CMP r10,#0x00           ;if it starts as 'the',then check if there are any other
words follow
            STRBEQ r10,[r2],#1      ;if it is the end of string, write 0x00 to end STRING2
            BEQ EXIT            ;then EXIT
            CMP r10,#0x20           ;if it is not the end of string, check if it is the word
'the',(no other letters follow)
            LDRB r10,[r9],#1    ;update checking pointer
            LDRBEQ r1,[r0,#2] ;if it is 'the',let r1 equals to 0x20(space) because we only want
to delete 'the'
            ADDEQ r0,#3             ;if it is 'the',update r0 to jump over this 'the',so we not
gonna write into memory
            B     WRITE             ;if it is 'the',write "space",if it is not 'the'(then we don't
update true pointer r0),write the whole word
    EXIT B EXIT                 ;this is how we exit


    STRING1 DCB "and the man said they must go" ;String1
    EoS DCB 0x00 ;end of string1
```

```
STRING2 space 0xFF ;just allocating 255 bytes
        END
```

```
                AREA assignment2Q2, DATA,READWRITE
                ENTRY
                ADR r13,S           ;let r13 points to reserved space
                ADR r0,X        ;let r0 points to the address of x
                LDR r0,[r0]         ;load the value of x into r0 for further calculation
                BL FUNC             ;jump into subroutine(function) and save link address in lr
                MOV r1,r0,LSL #1;when come back,after implementing the function,double the value and store in r1
                B EXIT              ;exit this program
     FUNC       STMIA r13!,{r1-r10}      ;store the value from r1 to r10
                ADR r10,A                ;let r10 points to the address of parameter a
                LDR r9,[r10],#4          ;let r9 gets the parameter a
                LDR r8,[r10],#4          ;let r8 gets the parameter b
                LDR r7,[r10],#4          ;let r7 gets the parameter c
                LDR r6,[r10]        ;let r6 gets the parameter d
                MULr1,r0,r8         ;b*x first
                MUL r2,r0,r0        ;x*x
                MLA r1,r2,r9,r1          ;a*x*x+b*x
                ADD     r1,r1,r7         ;a*x*x+b*x+c
                MOV r0,r1                ;move result into r0
                CMP r6,r0                ;compare d and y
                MOVLT r0,r6         ;if y is greater than d,return d
                LDMIA r13!,{r1-r10}      ;load the original value of r1 to r10 back
                MOV pc,r14               ;return to main function
     EXIT B     EXIT


                AREA assignment2Q2, CODE,READWRITE
     A    DCD5
     B    DCD6
     C    DCD7
     D    DCD 90
     X    DCD 3
     S    SPACE 0xFF
          END
```