Key to Practical 4
Calculator (Part 1)

Step 1

```
RemoveSpace     ; Save registers on the stack.
                movem.l d0/a0/a1,-(a7)

                ; A1 points to the destination string.
                ; (Destination string = Source string)
                movea.l a0,a1

\loop           ; Load a character of the string into D0 and increment A0.
                move.b  (a0)+,d0

                ; If the character is a space, branch to \loop.
                cmpi.b  #' ',d0
                beq     \loop

                ; Otherwise, the character is copied into the destination string
                ; and the destination pointer is incremented.
                ; If the character that has just been copied is not null,
                ; branch to \loop.
                move.b  d0,(a1)+
                bne     \loop

\quit           ; Restore registers from the stack and return from subroutine.
                movem.l (a7)+,d0/a0/a1
                rts
```

Step 2

```
IsCharError     ; Save registers on the stack.
                movem.l  d0/a0,-(a7)

\loop           ; Load a character of the string into D0 and increment A0.
                ; If the character is null, return false (no error).
                move.b   (a0)+,d0
                beq      \false

                ; Compare the character to the '0' character.
                ; If it is lower, return true (it is not a digit).
                cmpi.b   #'0',d0
                blo      \true

                ; Compare the character to the '9' character.
                ; If it is lower or equal, branch to \loop (it is a digit).
                ; If it is higher, return true (it is not a digit).
                cmpi.b   #'9',d0
                bls      \loop

\true           ; Return Z = 1 (error).
                ; (The BRA instruction does not modify Z.)
                ori.b    #%00000100,ccr
                bra      \quit

\false          ; Return Z = 0 (no error).
                andi.b   #%11111011,ccr

\quit           ; Restore registers from the stack and return from subroutine.
                ; (The MOVEM and RTS instructions do not modify Z.)
                movem.l (a7)+,d0/a0
                rts
```

Step 3

```
IsMaxError      ; Save registers on the stack.
                movem.l d0/a0,-(a7)

                ; Get the length of the string (in D0).
                jsr     StrLen

                ; If the length is longer than 5 characters, return true (error).
                ; If the length is shorter than 5 characters, return false (no error).
                cmpi.l  #5,d0
                bhi     \true
                blo     \false

                ; If the length is equal to 5 characters:
                ; Successive comparisons with '3', '2', '7', '6' and '7'.
                ; If longer, return true (error).
                ; If shorter, return false (no error).
                ; If equal, compare to the next character.
                cmpi.b  #'3',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'2',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'7',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'6',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'7',(a0)
                bhi     \true

\false          ; Return Z = 0 (no error).
                ; (The BRA instruction does not modify Z.)
                andi.b  #%11111011,ccr
                bra     \quit

\true           ; Return Z = 1 (error).
                ori.b   #%00000100,ccr

\quit           ; Restore registers from the stack and return from subroutine.
                ; (The MOVEM and RTS instructions do not modify Z.)
                movem.l (a7)+,d0/a0
                rts
```

Step 4

```
Convert         ; If the string is empty,
                ; return false (error).
                tst.b   (a0)
                beq     \false

                ; (At this stage, the string is not empty.)
                ; If a character error occurs,
                ; return false (error).
                jsr     IsCharError
                beq     \false

                ; (At this stage, the string is not empty
                ;  and contains only digits.)
                ; If the integer value of the string is higher than 32,767,
                ; return false (error).
                jsr     IsMaxError
                beq     \false

                ; The string is valid. We can convert it
                ; and return true (no error).
                jsr     Atoui

\true           ; Return Z = 1 (no error).
                ori.b   #%00000100,ccr
                rts

\false          ; Return Z = 0 (error).
                andi.b  #%11111011,ccr
                rts
```

Step 5

```
Print           ; Save registers on the stack.
                movem.l d0/d1/a0,-(a7)

\loop           ; Load a character of the string into D0.
                ; If the character is null, it is the end of the string.
                ; We can exit the subroutine.
                move.b  (a0)+,d0
                beq     \quit

                ; Display the character.
                jsr     PrintChar

                ; Increment the column where the next character will be displayed
                ; and branch to \loop.
                addq.b  #1,d1
                bra     \loop

\quit           ; Restore registers from the stack and return from subroutine.
                movem.l (a7)+,d0/d1/a0
                rts
```