Key to Practical 3
Stack and Subroutines

Step 4

```
LowerCount  movem.l d1/a0,-(a7)

            clr.l   d0

\loop       move.b  (a0)+,d1
            beq     \quit

            cmp.b   #'a',d1
            blo     \loop

            cmp.b   #'z',d1
            bhi     \loop

            addq.l  #1,d0
            bra     \loop

\quit       movem.l   (a7)+,d1/a0
            rts
```

Step 5

```
UpperCount  movem.l d1/a0,-(a7)

            clr.l   d0

\loop       move.b  (a0)+,d1
            beq     \quit

            cmp.b   #'A',d1
            blo     \loop

            cmp.b   #'Z',d1
            bhi     \loop

            addq.l  #1,d0
            bra     \loop

\quit       movem.l   (a7)+,d1/a0
            rts
```

```
DigitCount  movem.l d1/a0,-(a7)

            clr.l   d0

\loop       move.b  (a0)+,d1
            beq     \quit

            cmp.b   #'0',d1
            blo     \loop

            cmp.b   #'9',d1
            bhi     \loop

            addq.l  #1,d0
            bra     \loop

\quit       movem.l   (a7)+,d1/a0
            rts
```

```
AlphaCount  ; Count the number of small letters
            ; and push it onto the stack.
            jsr     LowerCount
            move.l  d0,-(a7)

            ; Count the number of capital letters and add it
            ; to the top of stack (without popping off).
            ; Top of stack = Small letters + Capital letters
            jsr     UpperCount
            add.l   d0,(a7)

            ; Count the number of digits.
            ; The top of stack (Small letters + Capital letters)
            ; is added to the number of digits (D0).
            ; The sum is loaded into D0.
            ; D0 = Small letters + Capital letters + Digits
            ; The top of stack is poppep off (postincrement mode).
            jsr     DigitCount
            add.l   (a7)+,d0

            ; Return from subroutine.
            rts
```

Step 6

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

                ; Initialize the output variable to 0.
                clr.l   d0

                ; Initialize the conversion variable to 0.
                clr.l   d1

\loop           ; Copy the current character into D1.
                ; Then A0 points to the next character (postincrement mode).
                move.b  (a0)+,d1

                ; If the copied character is null,
                ; branch to \quit (end of string).
                beq     \quit

                ; Otherwise, the character is converted into an integer.
                subi.b  #'0',d1

                ; Shift the output variable to the left (x10),
                ; and add the integer value.
                mulu.w  #10,d0
                add.l   d1,d0

                ; Next character.
                bra     \loop

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