```
01                      ;                                       RCSL: 43-GL2367
02                      ;                                       AUTHOR: DHA
03                      ;                                       EDITED: 76.02.15
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23                      ;                       MURMS
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42                      ; KEYWORD: MUS,RECORD I/O, MOVE, PROCEDURES,
43                      ;               LISTING.
44                      ; ABSTRACT: MUS RECORD I/O PROCEDURES.
45                      ; ASCII PAPER TAPE RCSL 43-GL2368
46                      ; REL. BINARY PAPER TAPE RCSL 43-GL2369
47
48
```

```
1 0002  MUK03
01                ;    ****** RECORD I/O PROCEDURES ******
02
03                .TITL MUK03
00                .NREL
05        000001  .TXTM 1
06        000012  .RDX 10
07
08
```

```
1 0003  NUCBOS
01                ; PROCEDURE GETREC(ZONE,BYTES,ADDR);
02                ; THE CALL PARAMETER BYTES IS ONLY RELEVANT FOR UNFORMATTED BLOCK
03                ; RECORDS. GETREC GETS THE NEXT RECORD FROM THE ZONE ACCORDING
04                ; TO THE FOLLOWING ALGORITHM:
05                ;
06                ; LOCAL PROCEDURE GETBLOCK;
07                ;    REPEAT INBLOCK(ZONE) UNTIL REM.ZONE<>0;
08                ;
09                ; LOCAL PROCEDURE GETHEAD;
10                ;    BEGIN
11                ;    LENGTH.ZONE:=4; IF REM<4 THEN GOTO BLOCKERROR;
12                ;    BYTES:= BYTE(TOP.ZONE) CON BYTE(TOP.ZONE+1)-4;
13                ;    LENGTH.ZONE:= BYTES;
14                ;    UPDATE(4);
15                ;    IF BYTES>REM THEN GOTO BLOCKERROR;
16                ;    END;
17                ;
18                ; LOCAL PROCEDURE UPDATE;
19                ;    BEGIN
20                ;    FIRST.ZONE:=ADDR:=TOP.ZONE;
21                ;    TOP.ZONE:=TOP.ZONE+COUNT;
22                ;    REM.ZONE:=REM.ZONE-COUNT;
23                ;    END;
24                ;
25                ; GETREC:
26                ;   CASE FORMAT.ZONE OF
27                ;    BEGIN
28                ; U:   GETBLOCK; BYTES:=REM.ZONE;
29                ;
30                ; UB:  WHILE BYTES>REM.ZONE DO GETBLOCK;
31                ;
32                ; F:   GETBLOCK; BYTES:=LENGTH.ZONE;
33                ;      IF BYTES<>REM.ZONE THEN GOTO BLOCKERROR;
34                ;
35                ; FB:  BYTES:=LENGTH.ZONE;
36                ;      IF BYTES>REM.ZONE THEN
37                ;        IF REM.ZONE<>0 THEN GOTO BLOCKERROR
38                ;      ELSE BEGIN GETBLOCK; GOTO FB END;
39                ;
40                ; V:   GETBLOCK; GETHEAD;
41                ;      IF BYTES>REM.ZONE THEN GOTO BLOCKERROR;
42                ;      IF BYTES=REM.ZONE THEN GOTO VB;
43                ;      IF ZONE.KIND(11)<>1 THEN GOTO BLOCKERROR
44                ;                          ELSE REM.ZONE:=BYTES;
45                ;      GOTO VB
46                ;
47                ; VB:  IF REM.ZONE=0 THEN GOTO V;
48                ;      GETHEAD;
49                ;    END !CASE!;
50                ;
51                ;  LENGTH.ZONE:=BYTES;
52                ;  UPDATE(BYTES);
53                ; RETURN;
54                ;         CALL:              RETURN:
55                ; AC0     BYTES              BYTES
56                ; AC1                        ADDR
57                ; AC2     ZONE               ZONE
58                ; AC3     LINK
```

```
                                                                                
                                                                                
                A50:    STA     3   Z5,2           ; GETREC:
                        JSR         A503           ;  CASE FORMAT.ZONE OF
                        A50U9                      ;  U-ADDR:
                        A5010+1                    ;  UB-ADDR;
                        A5011                      ;  F-ADDR:
                        A5011+3                    ;  FB-ADDR;
                        A5012                      ;  V-ADDR:
                        A5014                      ;  VB-ADDR;
                A5009:  JSR         A505           ; U:  GETBLOCK(ZONE,REM,BYTES);
                        STA     1   ZLENGTH,2      ;     LENGTH.ZONE:=REM;
                        JMP         A502           ;
                A5010:  JSR         A505           ; R1: GETBLOCK(ZONE,REM,BYTES);
                        STA     0   ZLENGTH,2      ; UB: LENGTH.ZONE:=BYTES;
                        ADCZ#   1,0 SZC            ;     IF BYTES>REM THEN
                        JMP          A5010         ;      GOTO R1;
                        JMP         A502           ;
                A5011:  MOV     1,1 SZR            ; F:  IF REM.ZONE<>0 THEN
                        JMP         A508           ;      GOTO ERROR;
                        JSR         A505           ;     GETBLOCK(ZONE,REM,BYTES);
                        LDA     0   ZLENGTH,2      ; FB: BYTES:=LENGTH.ZONE;
                        ADCZ#   1,0 SZC            ;     IF BYTES>REM THEN
                        JMP         A5011          ;      GOTO F:
                        JMP         A502           ;
                A5012:  JSR         A505           ; V:  GETBLOCK(ZONE,REM,BYTES);
                        JSR         A506           ;     GETHEAD(ZONE,REM,BYTES);
                        SUBZ#   0,1 SNC            ;     IF BYTES>REM THEN
                        JMP         A508           ;      GOTO ERROR;
                        SUB#    0,1 SNR            ;     IF BYTES=REM THEN
                        JMP         A5014          ;       GOTO VB;
                        LDA     1   .1B11          ;
                        LDA     3   ZKIND,2        ;
                        AND     1,3 SNR            ;     IF ZONE.KIND(11)<>1 THEN
                        JMP         A508           ;       GOTO ERROR         ELSE
                        MOV     0,1                ;       BEGIN
                        STA     1   ZREM,2         ;         REM:=BYTES; REM.ZONE:=REM;
                A5014:                             ;       END;
                        MOV     1,1 SNR            ; VB: IF REM=0 THEN
                        JMP         A5012          ;      GOTO V;
                        JSR         A506           ;     GETHEAD(ZONE,REM,BYTES);
                A502:                              ;  ESAC:
                        LDA     0   ZLENGTH,2      ;  BYTES:=LENGTH.ZONE:
                A5021:                             ; UPDATE:
                        JSR         A504           ;  UPDATE(BYTES);
                        JSR@        Z5,2           ; RETURN;

                A503:   LDA     1   ZFORMAT,2      ;  GET RECORDFORMAT;
                        ADD     1,3                ;
                        LDA     1   ZREM,2         ;  REM:=REM.ZONE;
                        JMP@        +0,3           ;


```

```
1  0205  PERMS
01
02
03                ; PROCEDURE UPDATE(BYTES);
04                ;       CALL:             RETURN:
05                ; AC0   BYTES             BYTES
06                ; AC1                     FIRST.ZONE
07                ; AC2   ZONE              ZONE
08                ; AC3   LINK
09                ;
10  0065'025023 A504: LDA     1   ZREM,2      ; UPDATE:
11  0056'100400       SUB     0,1            ;
12  0057'045023       STA     1   ZREM,2      ;  REM.ZONE:=REM.ZONE-BYTES;
13  0060'025220       LDA     1   ZTOP,2      ;
14  0061'100000       ADD     0,1            ;
15  0062'045020       STA     1   ZTOP,2      ;  TOP.ZONE:=TOP.ZONE+BYTES;
16  0063'100400       SUB     0,1            ;
17  0064'045017       STA     1   ZFIRST,2    ;   FIRST.ZONE:=TOP.ZONE-BYTES;
18  0065'000000       JMP         0,3        ; RETURN;
19
```

```
                        ; PROCEDURE GETBLOCK(ZONE,REM,BYTES);
                        ;
                        ;       CALL:           RETURN:
                        ; AC0                    BYTES
                        ; AC1                    REM
                        ; AC2  ZONE             ZONE
                        ; AC3  LINK
                        ;
A505:   STA     3   Z4,2        ; GETBLOCK:
        INBLOCK                 ; REPEAT
        LDA     1   ZREM,2      ;   INBLOCK(ZONE);
        MOV     1,1 SNR         ;   REM:=REM.ZONE
        JMP         .-3         ;  UNTIL REM<>0;
        LDA     0   ZLENGTH,2   ;  BYTES:=LENGTH.ZONE;
        JMP@        Z4,2        ; RETURN;

                        ; PROCEDURE GETHEAD(ZONE,REM,BYTES);
                        ;       CALL:           RETURN:
                        ; AC0                    BYTES
                        ; AC1  REM              REM
                        ; AC2  ZONE             ZONE
                        ; AC3  LINK
                        ;
A506:   STA     3   Z3,2        ; GETHEAD:
        LDA     0   .4          ;
        STA     0   ZLENGTH,2   ;
        ADCZ#   1,0 SZC         ;  IF REM<4 THEN
        JMP         A508        ;   GOTO ERROR;
        JSR         A504        ;  UPDATE(4);
        MOVZR   1,3             ;  ADDRESS:=FIRST/2;
        LDA     0   0,3         ;  BYTES:=WORD(ADDRESS);
        LDA     1   1,3         ;  TAIL:=WORD(ADDRESS+1);
        MOVS    1,1 SNC         ;  IF ODD(FIRST) THEN
        JMP         A5064       ;   BEGIN
        LDA     3   .255        ;
        AND     3,1             ;   B2:=TAIL(0:7);
        ANDCS   3,0             ;   B1:=BYTES(8:15) SHIFT 8;
        ADD     1,0             ;   BYTES:= B1+B2;
A5064:  LDA     1   .4          ;   END;
        SUB     1,0             ;  BYTES:=BYTES-4;
        LDA     1   ZREM,2      ;  REM:=REM.ZONE;
        STA     0   ZLENGTH,2   ;  LENGTH.ZONE:=BYTES;
        ADCZ#   1,0 SNC         ;   IF BYTES<=REM THEN
        JMP@        Z3,2        ;   RETURN;

A508:   LDA     1   SBLOCK      ; BLOCK ERROR:
        INC     1,1             ;  STATUS:=BLOCKERROR+SOFT;
        STA     1   ZU,2        ;  ZU.ZONE:=STATUS;
                        ; ENTRY TO GIVEUP WITH STATUS=188+1815, LENGTH.ZONE=ATTEMPTED
                        ; RECSIZE, AND REM.ZONE=AVAILABLE BYTES. IN CASE OF NORMAL RETURN
                        ; RECSIZE IS SET TO REM.ZONE.
        LDA     3   ZGIVEUP,2   ;
        MOV     3,3 SZR         ;  IF GIVEUP.ZONE<>0 THEN
        JSR         0,3         ;   GIVEUP.ZONE;
        LDA     0   ZREM,2      ;  BYTES:=REM.ZONE;
        JMP         A521        ;  GOTO UPDATE;
```

```
                ; PROCEDURE PUTREC(ZONE,BYTES,ADDR);
                ; MAKES ROOM FOR A RECORD IN THE ZONE ACCORDING
                ; TO THE FOLLOWING DESCRIPTION:
                ;
                ; LOCAL PROCEDURE UPDATE(COUNT);
                ;   SEE DESCRIPTION IN GETREC;
                ;
                ; LOCAL PROCEDURE CHANGEBLOCK;
                ;   BEGIN
                ;   OUTBLOCK(ZONE);
                ;   IF REM.ZONE<BYTES THEN
                ;     GOTO BLOCKERROR;
                ;   END;
                ;
                ;   CASE FORMAT.ZONE OF
                ;
                ; U:  CHANGEBLOCK;
                ;
                ; UB; IF REM.ZONE<BYTES THEN CHANGEBLOCK;
                ;
                ; F:  BYTES:=LENGTH.ZONE;
                ;     CHANGEBLOCK;
                ;
                ; FB: BYTES:=LENGTH.ZONE;
                ;     IF REM.ZONE<BYTES THEN CHANGEBLOCK;
                ;
                ; V:  BYTES:=BYTES+8;
                ;     CHANGEBLOCK;
                ;     UPDATE(4);
                ;     BYTES:=BYTES-8;
                ;     GOTO VB;
                ;
                ; VB: IF TOP.ZONE=FIRST.SHARE THEN GOTO V;
                ;     IF REM.ZONE<BYTES+4 THEN GOTO V;
                ;     UPDATE(4);
                ;     FIRST.ZONE(0:1):=BYTES+4;
                ;     FIRST.ZONE(2:3):=0;
                ;     LENGTH:=TOP.ZONE-FIRST.SHARE+BYTES;
                ;     HEAD:=FIRST.USED.ZONE;
                ;     HEAD(0:1):=LENGTH;
                ;     HEAD(2:3):=1;
                ;
                ;   END !CASE! ;
                ;
                ;   LENGTH.ZONE:=BYTES;
                ;   UPDATE(BYTES);
                ;
                ;         CALL:               RETURN:
                ; AC0     BYTES               BYTES
                ; AC1                         ADDR
                ; AC2     ZONE                ZONE
                ; AC3     LINK                DESTROYED
```

```
1 0038  NUMBIS
01                A51:                                  ; PUTREC:
02  00132'055031        STA     3   Z5,2              ;
03  00133'025015        LDA     1   ZFORMAT,2         ;   FORMAT:=FORMAT.ZONE;
04  00134'034117        LDA     3  .2                 ;
05  00135'137414        AND#    1,3 SZR               ;   IF FORMAT IS F THEN
06  00136'021016        LDA     0   ZLENGTH,2         ;     BYTES:=LENGTH.ZONE;
07  00137'175120        MOVZL   3,3                    ;
08  00140'137404        AND     1,3 SZR               ;     IF FORMAT>=4 THEN
09  00141'163000        ADC     3,0                   ;       BYTES:=BYTES+4
10  00142'055030        STA     3   Z4,2              ;   COUNT:=FORMAT AND 4;
11  00143'041016        STA     0   ZLENGTH,2        ;   LENGTH.ZONE:=BYTES;
12  00144'035023        LDA     3   ZREM,2            ;   REM:=REM.ZONE;
13  00145'162035        ADCZ#   3,0 SNC               ;   IF REM<BYTES OR
14  00146'125235        MOVZR#  1,1 SNC               ;      EVEN(FORMAT) THEN
15  00147'000456        JMP         A514              ;     CHANGEBLOCK;
16                A512:
17  00150'035021        LDA     3   ZUSED,2          ;
18  00151'035406        LDA     3   SFIRST,3         ;
19  00152'025020        LDA     1   ZTOP,2           ;
20  00153'021030        LDA     0   Z4,2            ;
21  00154'166405        SUB     3,1 SNR               ;   IF ZTOP=SFIRST THEN
22  00155'004700        JSR         A504              ;   UPDATE(COUNT);
23  00156'021016        LDA     0   ZLENGTH,2        ;
24  00157'004676        JSR         A504              ;     UPDATE(BYTES,FIRST);
25  00160'035030        LDA     3   Z4,2            ;
26  00161'175005        MOV     3,3 SNR               ;   IF COUNT=0 THEN
27  00162'003031        JMP@        Z5,2              ;     RETURN;
28  00163'135220        MOVZR   1,3                   ;   ADDR:=FIRST//2;
29  00164'126463        SUBC    1,1 SNC               ;   IF FIRST(15) THEN
30  00165'000414        JMP         A5135             ;     BEGIN
31  00166'045402        STA     1   2,3               ;     WORD(ADDR+2):=0;
32  00167'020147        LDA     1  .M256              ;
33  00170'107400        AND     0,1                   ;     B1:=BYTES(0:7) SHIFT 8;
34  00171'122700        SUBS    1,0                   ;     B2:=BYTES(8:15) SHIFT 8;
35  00172'041401        STA     0   1,3               ;     WORD(ADDR+1):=B2;
36  00173'021400        LDA     0   0,3               ;
37  00174'030147        LDA     3  .M256              ;
38  00175'163700        ANDS    3,0                   ;     B0:=WORD(ADDR)(0:7);
39  00176'035017        LDA     3   ZFIRST,2          ;
40  00177'175220        MOVZR   3,3                   ;
41  00200'123401        ADDS    1,0 SKP               ;     BYTES:=B0 SHIFT 8+(B1 SHIFT(-8))
```

```
1 4200  MARCS
01 0@201'2454A1 A5135:STA     1    1,3          ;     END ELSE WORD(ADDR+1):=0;
02 0@202'041400         STA     0    3,3          ;   WORD(ADDR):=BYTES;
03 0@203'021@16         LDA     0    ZLENGTH,2    ;
04 0@204'027116         LDA     1    .4           ;
05 0@205'122400         SUB     1,0              ;
06 0@206'041016         STA     0    ZLENGTH,2    ;   LENGTH.ZONE:=LENGTH.ZONE-4;
07 0@207'021417         LDA     0    ZFIRST,2     ;
08 0@210'123000         ADD     1,0              ;   FIRST.ZONE:=FIRST.ZONE+4;
09 0@211'041017         STA     0    ZFIRST,2     ;
10 0@212'035021         LDA     3    ZUSED,2      ;   SH:=USED.ZONE;
11 0@213'025416         LDA     1    SFIRST,3     ;   FIRST:=FIRST.SH;
12 0@214'135220         MOVZR   1,3              ;   ADDR:=FIRST//2;
13 0@215'021020         LDA     0    ZTOP,2       ;
14 0@216'122400         SUB     1,3              ;   LENGTH:=TOP.ZONE-FIRST.SHARE;
15 0@217'041400         STA     0    3,3          ;   WORD(ADDR):=LENGTH;
16 0@220'102400         SUB     0,0              ;
17 0@221'041401         STA     0    1,3          ;   WORD(ADDR+1):=0;
18 0@222'025017         LDA     1    ZFIRST,2     ;
19 0@223'021016         LDA     0    ZLENGTH,2    ;
20 0@224'003031         JMP@    25,2             ;
21
22
23 0@225'006206 A514: OUTBLOCK                   ; CHANGEBLOCK:
24 0@226'035023         LDA     3    ZREM,2       ;
25 0@227'021016         LDA     0    ZLENGTH,2    ;
26 0@230'162033         ADCZ#   3,3  SNC          ;   IF REM.ZONE<LENGTH.ZONE THEN
27 0@231'000717         JMP             A512      ;
28 0@232'000670         JMP             A516      ;   GOTO BLOCK ERROR;
29
30
```

```
                ; PROCEDURE MOVE(PARAMADDR);
                ; MOVES A NUMBER OF BYTES FROM ONE LOCATION TO ANOTHER.
                ; THE ADDRESSES MUST BE BYTE ADDRESSES.
                ;     CALL:                     RETURN:
                ; AC0                           DESTROYED
                ; AC1                           DESTROYED
                ; AC2 PARAMADDR                 PARAMADDR
                ; AC3 LINK                      DESTROYED
                ;
                ; PARAMADDR: COUNT
                ;         +1: TO ADDRESS
                ;         +2: FROM ADDRESS
                ;         +3: WORK LOCATION

0233'055003 A83:  STA     3  +3,2        ; MOVE:
0234'034040       LDA     3   CUR        ;
0235'051424       STA     2   SAVE,3     ;   SAVE(PARAMADDR);
0236'035000 A831: LDA     3   +0,2       ; START:
0237'174513       NEGLA   3,3 SNC        ;   IF COUNT<=0 THEN
0240'000003       JMP@        +3,2       ;     RETURN;
0241'025001       LDA     1   +1,2       ;
0242'125223       MOVZR   1,1 SNC        ;   IF IO(15) OR
0243'175225       MOVZR   3,3 SNR        ;      COUNT=1 THEN
0244'000402       JMP         .+2        ;
0245'000413       JMP         A832       ;     BEGIN
0246'025002       LDA     1   +2,2       ;
0247'000174       GETBYTE                ;     GETBYTE(FROMADDR,BYTE);
0250'051024       LDA     2   SAVE,2     ;
0251'025001       LDA     1   +1,2       ; LASTBYTE:
0252'000175 A8315:PUTBYTE              ;     PUTBYTE(TOADDR,BYTE);
0253'051024       LDA     2   SAVE,2     ;
0254'015000       DSZ         +0,2       ;     DECR(COUNT);
0255'011001       ISZ         +1,2       ;     INCR(TO);
0256'011002       ISZ         +2,2       ;     INCR(FROM);
0257'000757       JMP         A831       ;     GOTO START;
                A832:                    ;     END;
0260'164000       NEG     3,0            ;   ODDRETURN:=COUNT EXTRACT 1;
0261'035003       LDA     3   3,2        ;   COUNT:=-COUNT//2;
0262'041000       STA     0   0,2        ;   SAVE(COUNT);
0263'175130       MOVL    3,3            ;
0264'055003       STA     3   3,2        ;
0265'035002       LDA     3   +2,2       ;   FROM:= FROM ADDR//2;
0266'175223       MOVZR   3,3 SNC        ;   IF FROM ADDR(15)=0 THEN
0267'000427       JMP         A834       ;     GOTO MATCH;
0270'055002       STA     3   +2,2       ;   SAVE(FROM);
0271'045001       STA     1   +1,2       ;   SAVE(TO);
0272'030143       LDA     3   .255       ;
0273'027002       LDA@    1   +2,2       ;   WORD:= 0.FROM;
0274'167400       ANL     3,1            ;   LEFT:= WORD(8:15);
```

```
                A833:                           ; NO MATCH:
                        ISZ         +2,2        ;   INCR(FROM);
                        LDA@    0   +2,2        ;   WORD:= @.FROM;
                        ADL     0,1             ;   BOTH:= WORD+LEFT;
                        AND     3,0             ;   BYTE:= WORD(8:15);
                        SUBS    0,1             ;   WORD:= BOTH-BYTE;
                        STA@    1   +1,2        ;   @.TO:= WORD;
                        ISZ         +1,2        ;   INCR(TO);
                        MOV     0,1             ;   LEFT:= BYTE;
                        ISZ         +0,2        ;   IF INCR(COUNT)<>0 THEN
                        JMP         A833        ;     GOTO NO MATCH;
                        LDA     3   1,2         ;
                        MOVZL   3,1             ;   TO:=TO*2;
                A8335:  LDA     3   3,2         ; CHECKLAST:
                        MOVZR   3,3 SNC         ;   IF -,ODDRETURN THEN
                        JMP         0,3         ;     RETURN;
                        STA     3   3,2         ;
                        JMP         A8315       ;   GOTO LASTBYTE;
                A834:   MOV     1,2             ; MATCH:
                A835:   LDA     1   +1,3        ;
                        STA     1   +1,2        ;   @.TO:= @.FROM;
                        INC     3,3             ;   INCR(FROM);
                        INC     2,2             ;   INCR(TO);
                        INC     0,0 SZR         ;   IF INCR(COUNT)<>0 THEN
                        JMP         A835        ;     GOTO MATCH;
                        LDA     0   0,3         ;
                        LDA     1   .M256       ;   BYTE:=FROM(0:7);
                        AMDS    1,0             ;
                        MOVZL   2,1             ;   TO:=TO*2;
                        LDA     2   CUR         ;
                        LDA     2   SAVE,2      ;   RESTORE(PARAMADDR);
                        JMP         A8335       ;   GOTO  CHECKLAST;

                ;   ******   END OF RECORD I/O PROCEDURES   ******

                .LOC GETREC-GOS
                        A50
                .LOC PUTREC-GOS
                        A51
                .LOC MOVE-GOS
                        A83

                .NREL

                .END
```

A50	0000000'	4/01	11/37				
A5009	0000010'	4/03	4/09				
A5012	0000013'	4/04	4/12	4/15			
A5011	0000020'	4/05	4/06	4/17	4/22		
A5012	0000027'	4/07	4/24	4/38			
A5014	0000043'	4/08	4/29	4/36			
A502	0000046'	4/11	4/16	4/23	4/40		
A5021	0000047'	4/42	6/56				
A503	0000051'	4/02	4/46				
A504	0000055'	4/43	5/10	6/29	8/22	8/24	
A505	0000066'	4/09	4/12	4/19	4/24	6/09	
A506	0000075'	4/25	4/39	6/24			
A5064	0000114'	6/30	6/39				
A508	0000122'	4/18	4/27	4/33	6/28	6/46	9/28
A51	0000132'	8/01	11/39				
A512	0000154'	8/16	9/27				
A5135	0000201'	8/30	9/01				
A514	0000225'	8/15	9/23				
A83	0000253'	10/15	11/41				
A831	0000256'	10/18	10/35				
A8315	0000252'	10/30	11/18				
A832	0000260'	10/25	10/36				
A833	0000275'	11/01	11/11				
A8335	0000311'	11/14	11/32				
A834	0000316'	10/14	11/19				
A835	0000317'	11/21	11/25				