



# Tech Info Library

## Pascal III: Accessing the extra memory (5 of 5)

Revised: 11/30/84  
Security: Everyone

Pascal III: Accessing the extra memory (5 of 5)

=====

```
PROCEDURE FreeStringSpace;
BEGIN
    IF SegNum <> 1 THEN De_Allocate(SegNum);
END;

PROCEDURE Convert(Who:STRPTR;
                  VAR TempBank,TempAddr:INTEGER);
BEGIN
    TempBank := Bank;
    IF Who >= 0 THEN BEGIN
        {must be in second half of chunk}
        TempBank := TempBank+2;
        {$IFC  DEBUG}
        WRITE('+');
        {$ENDC  DEBUG}
    END;
    TempAddr := Who+Who+Base;
    IF (TempAddr < Base) AND (TempAddr >= 0) THEN BEGIN
        {must be in third bank of this half}
        TempBank := TempBank+2;
        {$IFC  DEBUG}
        WRITE('2');
        {$ENDC  DEBUG}
    END;
    {$IFC  DEBUG}
    WRITE('(',TempBank,':',TempAddr,')');
    {$ENDC  DEBUG}
END;

FUNCTION PutString {(VAR S:STRING1;
                    VAR WHERE:STRPTR): BOOLEAN};
VAR
    NewTos:INTEGER; {if this succeeds, where will Tos be?
                    (base relative word pointer)}
    TempBank,
    TempAddr:  INTEGER;    {real bank address of string}
```

```

BEGIN
{check for space overflow; this is tricky due to negative
addresses:
(if limit is positive (i.e. we have at least 32k words))
    Tos +      -      (note: 0 is +)
    ===== CMP means Overflow if newtos > limit
n      |      EW  means impossible situation
e +   |      CMP      CMP      (must have already overflowed)
w      |      OVFL means overflow
t      |      OK   means no overflow possible
o -   |      OVFL      CMP
s      |
(if limit is negative (i.e. we have less than 32k words))
    Tos +      -
    =====
n      |
e +   |      EW      OVFL (=CMP)
w      |
t      |
o -   |      EW      CMP
s      |
}
{$IFC  DEBUG}
WRITE('Storing "',S,'" at ',tos);
{$ENDC  DEBUG}
NewTos := Tos+(LENGTH(S)+2) DIV 2;
IF (Tos < NewTos) AND (NewTos < Limit) THEN BEGIN
    PutString := TRUE;
    Convert(Tos,TempBank,TempAddr);
    FetchBytes(-1,AtSign(S),TempBank,TempAddr,0,Length(s)+1);
    Where := Tos;                {hand back pointer}
    Tos := NewTos;
END ELSE BEGIN
    PutString := FALSE;
END;
{$IFC  DEBUG}
WRITELN;
{$ENDC  DEBUG}
END;

```

```

PROCEDURE GetString{(Who:INTEGER; VAR S:STRING255)};
VAR
    TempBank,
    TempAddr:  INTEGER;      {real bank address of string}
BEGIN
    {compute real address of string in memory}
    {$IFC  DEBUG}
    WRITE('Getting ',Who);
    {$ENDC  DEBUG}
    Convert(Who,TempBank,TempAddr);
    FetchBytes(TempBank,TempAddr,-1,AtSign(S),1,0);
    {$IFC  DEBUG}
    WRITELN(' ==>"',S,'"');

```

```
        {$ENDC DEBUG}  
    END;
```

```
BEGIN  
    SegNum := -1;  
END.
```

Apple Tech Notes

Tech Info Library Article Number:643