

Re: Fastcode UpperCase B&V 0.2

Source: <http://coding.derkeiler.com/Archive/Delphi/borland.public.delphi.language.basm/2003-10/0019.html>

From: John O'Harrow (john_at_elmcrest.demon.co.uk)

Date: 10/01/03

Date: Wed, 1 Oct 2003 15:48:21 +0100

Here are my first entries for this challenge.

Possibly of interest to others – In developing these, I have invented a new fast technique for converting an 'a'..'z' character into uppercase without any branches (It's so simple, I don't know why it hasn't been done before)

:-

```
{ Convert character in AL to Uppercase }
sub al, 'a'
sub al, 26
sbb ecx, ecx
and cl, -$20
add al, 'a'+26
add al, cl
```

regards,
John.

```
unit UpperCaseJOHUnit;
```

```
interface
```

```
function UpperCaseJOH_PAS (const S: string): string;
function UpperCaseJOH_IA32_1(const S: string): string;
function UpperCaseJOH_IA32_2(const S: string): string;
```

```
implementation
```

```
uses
```

```
  SysUtils;
```

```
function UpperCaseJOH_PAS(const S: string): string;
```

```
const
```

```
  Lookup : array[Byte] of Byte =
```

```
    ($00,$01,$02,$03,$04,$05,$06,$07,$08,$09,$0A,$0B,$0C,$0D,$0E,$0F,
     $10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$1A,$1B,$1C,$1D,$1E,$1F,
     $20,$21,$22,$23,$24,$25,$26,$27,$28,$29,$2A,$2B,$2C,$2D,$2E,$2F,
     $30,$31,$32,$33,$34,$35,$36,$37,$38,$39,$3A,$3B,$3C,$3D,$3E,$3F,
```

```

$40,$41,$42,$43,$44,$45,$46,$47,$48,$49,$4A,$4B,$4C,$4D,$4E,$4F,
$50,$51,$52,$53,$54,$55,$56,$57,$58,$59,$5A,$5B,$5C,$5D,$5E,$5F,
$60,$41,$42,$43,$44,$45,$46,$47,$48,$49,$4A,$4B,$4C,$4D,$4E,$4F,
$50,$51,$52,$53,$54,$55,$56,$57,$58,$59,$5A,$7B,$7C,$7D,$7E,$7F,
$80,$81,$82,$83,$84,$85,$86,$87,$88,$89,$8A,$8B,$8C,$8D,$8E,$8F,
$90,$91,$92,$93,$94,$95,$96,$97,$98,$99,$9A,$9B,$9C,$9D,$9E,$9F,
$A0,$A1,$A2,$A3,$A4,$A5,$A6,$A7,$A8,$A9,$AA,$AB,$AC,$AD,$AE,$AF,
$B0,$B1,$B2,$B3,$B4,$B5,$B6,$B7,$B8,$B9,$BA,$BB,$BC,$BD,$BE,$BF,
$C0,$C1,$C2,$C3,$C4,$C5,$C6,$C7,$C8,$C9,$CA,$CB,$CC,$CD,$CE,$CF,
$D0,$D1,$D2,$D3,$D4,$D5,$D6,$D7,$D8,$D9,$DA,$DB,$DC,$DD,$DE,$DF,
$E0,$E1,$E2,$E3,$E4,$E5,$E6,$E7,$E8,$E9,$EA,$EB,$EC,$ED,$EE,$EF,
$F0,$F1,$F2,$F3,$F4,$F5,$F6,$F7,$F8,$F9,$FA,$FB,$FC,$FD,$FE,$FF);

var
  Len, I : Integer;
begin
  Len := Length(S);
  SetLength(Result, Len);
  for I := 1 to Len do
    PByteArray(Result)[I-1] := Lookup[Byte(S[I])];
  end;

```

```

function UpperCaseJOH_IA32_1(const S: string): string;
asm

```

```

  test eax, eax
  jz @@Done
  push ebx
  push edi
  push esi
  mov esi, eax {Addr(S)}
  mov edi, edx {Addr(Result)}
  mov edx, [eax-4] {Length}
  test edx, edx
  jle @@Exit {Length <= 0}
  lea ebx, [edx-1] {Length - 1}
  mov eax, edi {Addr(Result)}
  call system.@LStrSetLength {Create Result String}
  mov edx, [edi]
  mov ecx, 'a' shl 8 + 'z' {ch = 'a', cl = 'z'}
@@Loop:
  movzx eax, [esi+ebx]
  cmp al, ch
  jb @@CharDone
  cmp al, cl
  ja @@CharDone
  sub al, 'a' - 'A'
@@CharDone:
  mov [edx+ebx], al
  sub ebx, 1
  jge @@Loop
@@Exit:
  pop esi

```

```
pop edi
pop ebx
@@Done:
end;
```

```
function UpperCaseJOH_IA32_2(const S: string): string;
```

```
asm
  test eax, eax
  jz @@Done
  push ebx
  push edi
  push esi
  mov esi, eax {Addr(S)}
  mov edi, edx {Addr(Result)}
  mov edx, [eax-4] {Length}
  test edx, edx
  jle @@Exit {Length <= 0}
  lea ebx, [edx-1] {Length - 1}
  mov eax, edi {Addr(Result)}
  call system.@LStrSetLength {Create Result String}
  mov edx, [edi]
@@Loop:
  movzx eax, [esi+ebx] {Copy Character}
  mov [edx+ebx], al
  sub al, 'a'
  sub al, 26
  sbb ecx, ecx
  and cl, -$20
  add [edx+ebx], cl {Convert Character}
  sub ebx, 1
  jge @@Loop
@@Exit:
  pop esi
  pop edi
  pop ebx
@@Done:
end;
```

```
end.
```