Université de Nanterre jeudi 18 avril 2013
architecture des ordinateurs contrôle continu

```
    .p2align 4,,15
    .globl  p1
    .type   p1, @function
p1:
.LFB0:
    .cfi_startproc
    testq   %rdi, %rdi
    movq    %rdx, %rcx
    je      .L1
    subq    $1, %rdi
    .p2align 4,,10
    .p2align 3
.L3:
    movq    (%rcx,%rdi,8),%rax
    movq    %rax, %rdx
    sarq    $63, %rdx
    idivq   %rsi
    movq    %rdx, (%rcx,%rdi,8)
    subq    $1, %rdi
    cmpq    $-1, %rdi
    jne    .L3
.L1:
    rep
    ret
    .cfi_endproc
.LFE0:
    .size   p1, .-p1
```

```
    .p2align 4,,15
    .globl  p2
    .type   p2, @function
p2:
.LFB1:
    .cfi_startproc
    testq   %rdi, %rdi
    je      .L6
    subq    $1, %rdi
    .p2align 4,,10
    .p2align 3
.L8:
    movq    (%rsi), %rax
    addq    $8, %rsi
    movq    %rax, (%rdx,%rdi,8)
    subq    $1, %rdi
    cmpq    $-1, %rdi
    jne     .L8
.L6:
    rep
    ret
    .cfi_endproc
.LFE1:
    .size   p2, .-p2
```

```
    .p2align 4,,15
    .globl  f
    .type   f, @function
f:
.LFB2:
    .cfi_startproc
    xorl    %eax, %eax
    testq   %rdi, %rdi
    je      .L11
    subq    $1, %rdi
    .p2align 4,,10
    .p2align 3
.L12:
    xorq (%rsi,%rdi,8),%rax
    subq    $1, %rdi
    cmpq    $-1, %rdi
    jne     .L12
.L11:
    rep
    ret
    .cfi_endproc
.LFE2:
    .size   f, .-f
```

Retrouvez le source C des deux procédures `p1` et `p2` et de la fonction `f`.

Récrivez `p1`, `p2` et `f` en supprimant `testq` et `cmpq` et en utilisant le carry après les `subq`.

Récrivez `p1` en supposant que les données traitées sont non signées.

Ecrivez l'équivalent en assembleur de la fonction C ayant pour prototype `long st(long t[],long n);` qui calcule $\sum_{i=0}^{n-1} t[i]/(i+1)$.

Rappels : `imul`, `idiv` et `sar` (Shift Arithmetic Right) sont des opérations signées, tandis que `mul`, `div` et `shr` (SHift Right) sont des opérations non signées.

```
sar : -43>>3 = -6
shr : 42u>>3 = 5u
xor : 45^45 = 0
and : 11&13 = 9
```

```
void p1(long n,long d,long t[])
{ while(n--) t[n]%=d; }                                          4 pt
void p2(long n,long t[],long u[])
{ while(n--) u[n]=*t++; }                                        4 pt
long f(long n,long t[])
{ long s=0; while(n--) s^=t[n]; return s; }                      4 pt
```

Au début des 3 procédures, il faut remplacer

```
    testq   %rdi, %rdi                          subq    $1, %rdi     2 pt
    je      .L1             par                 jc      .L1
    subq    $1, %rdi
```

et en fin de boucle

```
    subq    $1, %rdi                            subq    $1, %rdi     2 pt
    cmpq    $-1, %rdi       par                 jnc    .L3
    jne    .L3
```

Il faut remplacer

```
    movq    %rax, %rdx      par                 xorl    %edx, %edx   2 pt
    sarq    $63, %rdx
    idivq   %rsi                                divq    %rsi         1 pt
```

```
st:
    xorl    %eax,%eax                                                4 pt
    testq   %rdi,%rdi
    je      .L1
    subq    $8,%rsi
.l2:
    movq    (%rsi,%rdi,8),%rdx
    idivq   %rdi,%rdx
    addq    %rdx,%rax
    subq    $1,%rdi
    jne     .L2
.L1:
    ret
```