Projekt 1 Maskinnær programmering

Sulaiman Kasas s195462 Christine Nielsen s141361 Alaa Haddoud s205477

Gruppe 7

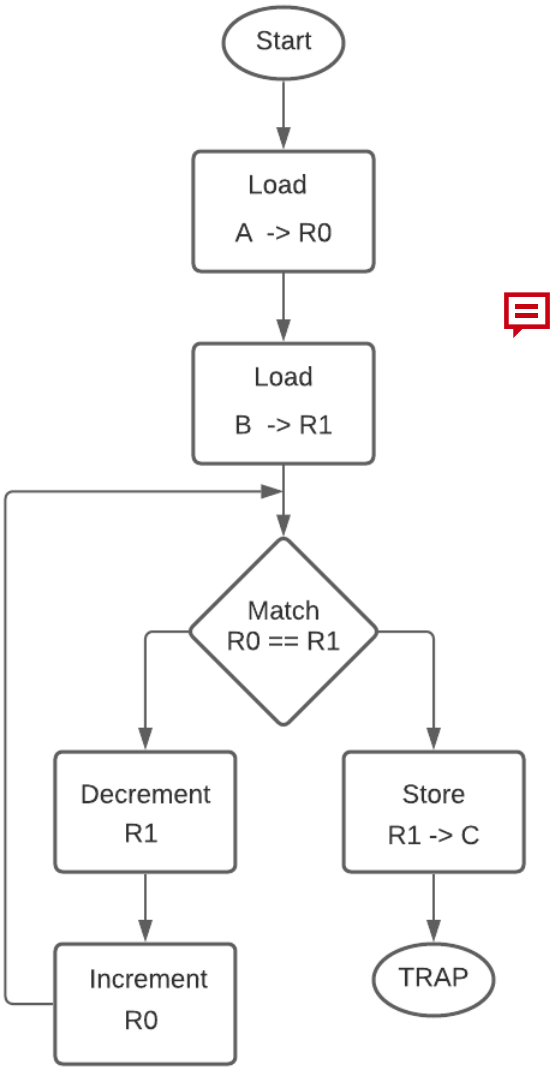
Vi har disse opgaver fra det sidste år

Opgave 1

```
; Der skal findes tallet C der ligger mellem 2 andre tal A og B.
; A og B er begge lige numre og A er mindre end B.
; De manglende instruktioner udfyldes.
   .ORIG x3000 ; Der startes fra adresse x3000
   LD R0,A ; Loader værdien fra A i R0
   LD R1,B ; Loader værdien fra B i R1
X  NOT R2,R0 ; Finder den inverse værdi af R0 og gemmer den i R2 (a)
   ADD R2,R2,#1 ; Tæller R2 1 op så man får den negative værdi af R0 og gem i R2 (b)
   ADD,R2,R2,R1 ; Lægger R2 til R1 og gemmer i R2 (siden R2 er negativt bliver R2 reelt trukket fra R1)
   BRz DONE ; Hvis den forrige instruktion gav 0: gå videre til DONE (c)
   ADD,R1,R1,#-1 ; Tæller B 1 ned
   ADD R0,R0,#1 ; Tæller A 1 op (d)
   BRnzp X ; Hvis den forrige instruktion gav noget negativt, 0 eller positivt: gå til X
DONE ST R1,C ; Gem værdien i R1 i C
   TRAP x25 ; Stop programmet
A  .BLKW #1 ; Reserver 1 lokation i hukkkomelsen og kald den A
B  .BLKW #1 ; Reserver 1 lokation i hukkkomelsen og kald den B
C  .BLKW #1 ; Reserver 1 lokation i hukkkomelsen og kald den C
   .END
```

Gruppe 7

Assignment 1:

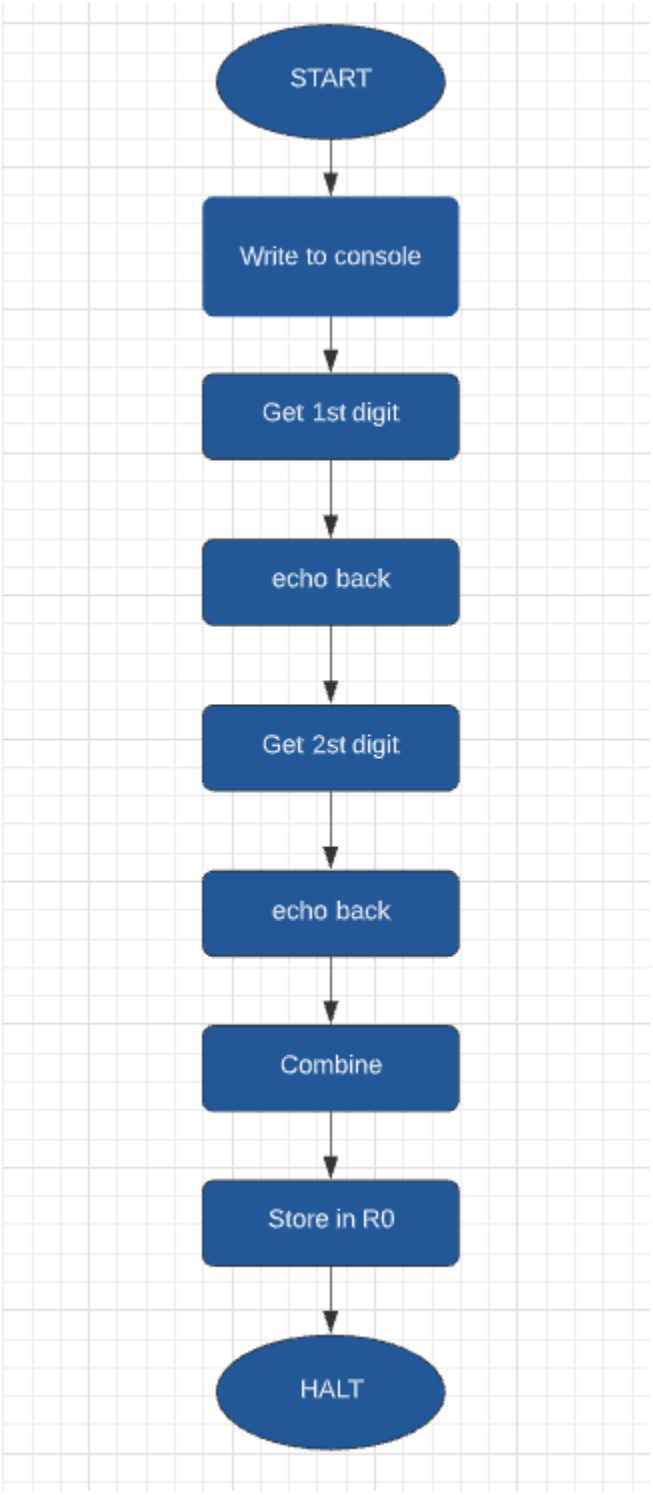


Opgave 2

```
; Skal læse 2 tal fra konsol og gemme dem i R0
      .ORIG x3000
; Selve programmet, der aktiverer readS
      JSR readS ; Kalder readS og returnerer linjen efter
      HALT ; TRAP x25 ; Stopper programmet
;
; Subrutine der skriver til brugeren og tager imod de 2 tal og gemmer dem i R0
readS   ST R7,RETURNreadS ; Gemmer hvor readS skal returnere i RETURN
      LEA R0,MSGr ; Henter adressen til beskeden der skal skrives, til R0
      PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
      ;
      GETC ; TRAP x20 ; Tager imod et tegn i R0
      OUT ; TRAP x21 ; Skriver tegner ud igen (echo)
      ;
      LD R4,fASCII ; Sætter R4 til -48
      ADD R0,R0,R4 ; Trækker 48 fra input, for at konvertere fra ASCII til tal
      AND R4,R4,#0 ; Sætter R4 til 0
      ;
      AND R1,R1,#0 ; Sætter R1 til 0
      ADD R1,R1,#10 ; Sætter R1 til 10
      JSR MULT ; Kører subrutine MULT (R0 * R1) = R2
      ;
      GETC ; TRAP x20 ; Tager imod et tegn i R0
      OUT ; TRAP x21 ; Skriver tegner ud igen (echo)
      ;
      LD R4,fASCII ; Sætter R4 til -48
```

```
      ADD R0,R0,R4 ; Trækker 48 fra input, for at konvertere fra ASCII til tal
      AND R4,R4,#0 ; Sætter R4 til 0
      ;
      ADD R0,R2,R0 ; Lægger input til R2 (første tal * 10)
      ;
      LD R7,RETURNreadS ; Sætter R7 tilbage til det readS skal returnere til
      RET ; JMP R7 ; Returnerer
;
; Subrutine som "ganger" 2 tal
; Det ene tal læses fra R0, det andet fra R1, resultatet findes i R2
; Antager at begge tal er positive
MULT    AND R2,R2,#0 ; Sætter R2 til 0
      ADD R2,R0,#0 ; Gemmer det originale tal fra R0 i R2
      BR MULTI ; Sætter "gange" igang
MULTI   ADD R1,R1,#-1 ; Tæller R1 en ned
      BRnz DONE ; Hvis R1 nu er 0 eller negativ, stop
      ADD R2,R2,R0 ; Tæller R2 op med R0
      BR MULTI ; Går til MULTI igen
DONE    AND R1,R1,#0 ; Sætter R1 til 0
      RET ; JMP R7 ; Returnerer
;
; Vores variabler
RETURNreadS .BLKW #1 ; Gemmer adressen, som readS skal returnere til når den er færdig
fASCII      .FILL #-48 ; Gemmer værdien -48, så et et-ciftret til kan konverteres fra ASCII til binær
MSGr        .STRINGZ "Input a 2 digit decimal number:" ; Beskeden readS skriver ud
         .END
```

Gruppe 7


START
Write to console
Get 1st digit
echo back
Get 2st digit
echo back
Combine
Store in R0
HALT

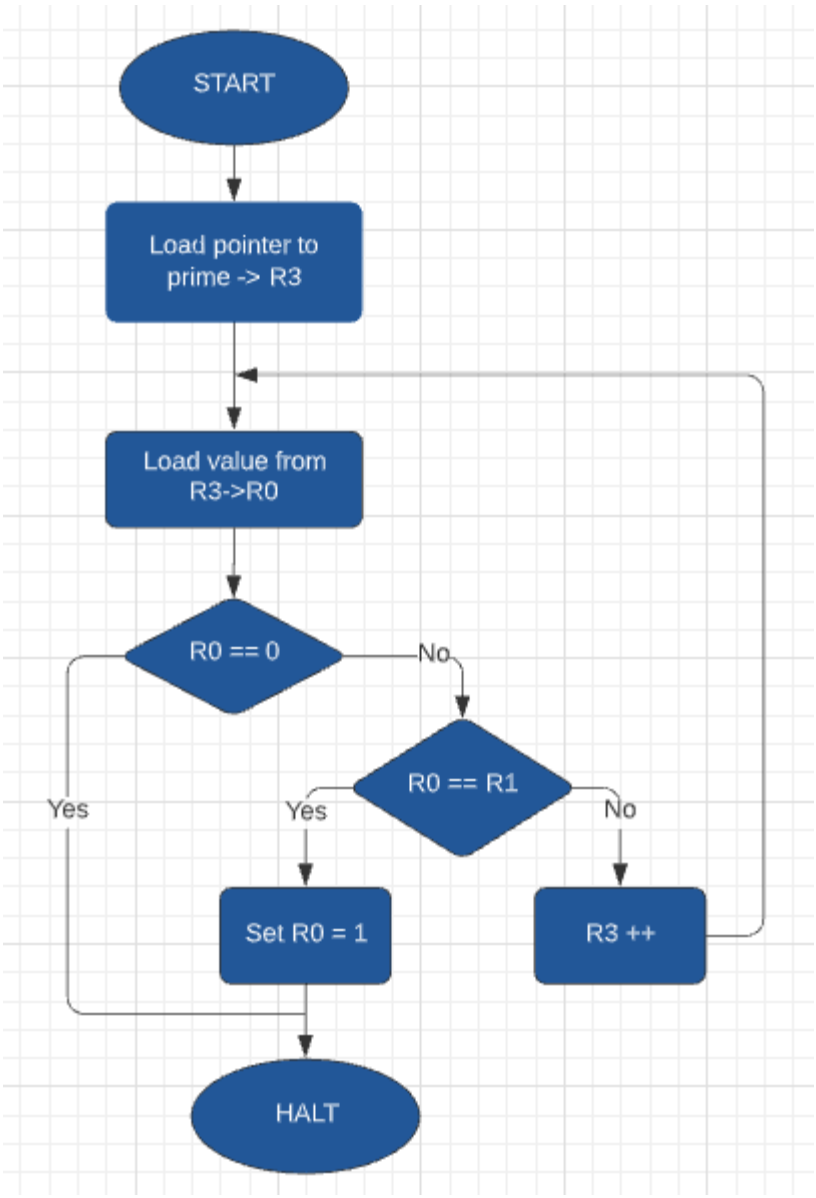
Gruppe 7

Opgave 3

```
; Skal finde ud af om et tal mellem 0 og 99 er et primtal
; Primtal under 99:
; 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97
; Hvis det er, skal der gemmes 1 i R0, hvis ikke, 0 i R0
        .ORIG x3000
        JSR isPrime ; Kalder isPrime og returnerer linjen efter
        HALT ; TRAP x25 ; Stopper programmet
;
; Tjekker værdien i R1, for om det er primtal
isPrime ST R7,RETURNisPrime ; Gemmer hvor isPrime skal returnere i RETURN
        LEA R3, PRIMES ; Henter adressen til første primtal til R3
LOOP    LDR R0,R3,#0 ; Henter værdien fra den adresse i memory, R3 pejer på
        BRz DONE ; Hvis værdien er 0, gå til DONE
        NOT R0,R0 ; Invertering af R0
        ADD R0,R0,#1
        ADD R0,R0,R1 ; Hvis forskellen på de to tal (R1 og R0) er 0, så er de ens og R1 er et primtal
        BRz FOUND ; Hvis den forrige handling gav 0, er de to tal ens og det givne er et primtal, gå til FOUND
        ADD R3,R3,#1 ; Ret R3 til at peje på næste felt i hukommelsen
        BR LOOP ; Hvis det ikke var et primtal, gå til LOOP, igen
FOUND   AND R0,R0,#1 ; Sæt R0 til 0
        ADD R0,R0,#1 ; Sæt R0 til 1
DONE    LD R7,RETURNisPrime ; Sætter R7 tilbage til det isPrime skal returnere til
        RET ; JMP R7 ; Returnerer
;
; Vores variabler
RETURNisPrime   .BLKW #1 ; Gemmer adressen, som isPrime skal returnere til når den er færdig
PRIMES          .FILL #2
                .FILL #3
```

```
            .FILL #5
            .FILL #7
            .FILL #11
            .FILL #13
            .FILL #17
            .FILL #19
            .FILL #23
            .FILL #29
            .FILL #31
            .FILL #37
            .FILL #41
            .FILL #43
            .FILL #47
            .FILL #53
            .FILL #59
            .FILL #61
            .FILL #67
            .FILL #71
            .FILL #73
            .FILL #79
            .FILL #83
            .FILL #89
            .FILL #97
PRIMELAST   .FILL #0
            .END
```

Gruppe 7

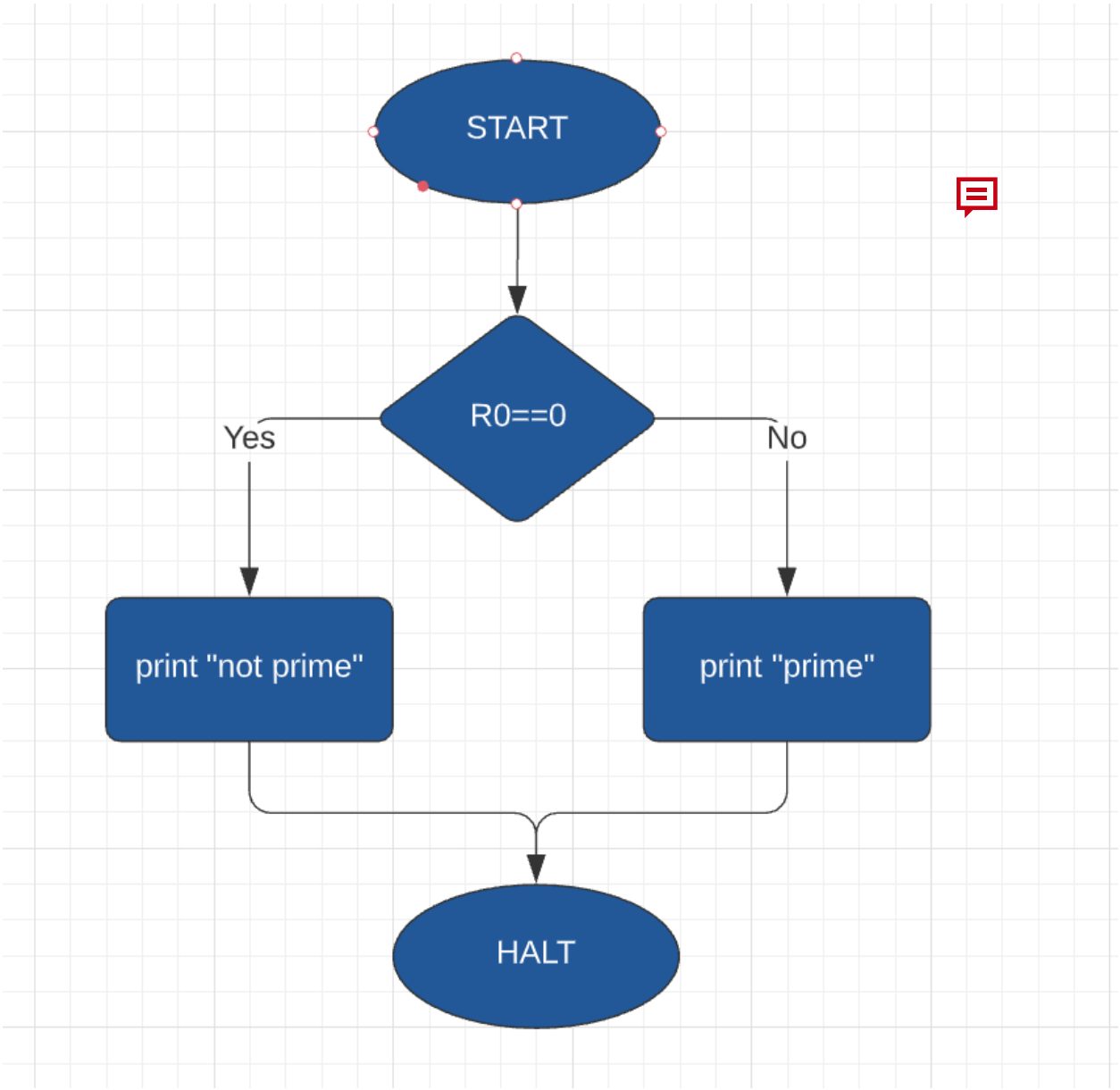

START
Load pointer to prime -> R3
Load value from R3->R0
R0 == 0
No
Yes
R0 == R1
Yes
No
Set R0 = 1
R3 ++
HALT

Gruppe 7

Opgave 4

```
; Skal læse R0, hvis der står 0 skal der skrives "The number is not prime", ellers skal den skrive "The number is prime" til konsollen
        .ORIG x3000
        JSR resultS ; Kalder resultS og returnerer linjen efter
        HALT ; TRAP x25 ; Stopper programmet
;
;
resultS ST R7,RETURNresultS ; Gemmer hvor resultS skal returnere i RETURNresultS
        ADD R0,R0,#0 ; Lægger 0 til R0
        BRz NOPE ; Hvis 0 + R0 er lig 0, så stod der 0, gå til NOPE
        BR PRIM ; Ellers stod der ikke 0, gå til PRIM
NOPE    LEA R0,MSGnp ; Henter adressen til beskeden der skal skrives, til R0
        PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
        BR DONE ; Gå til DONE
PRIM    LEA R0,MSGp ; Henter adressen til beskeden der skal skrives, til R0
        PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
DONE    LD R7,RETURNresultS ; Sætter R7 tilbage til det resultS skal returnere til
        RET ; JMP R7 ; Returnerer
;
; Vores variabler
RETURNresultS   .BLKW #1 ; Gemmer adressen, som resultS skal returnere til når den er færdig
MSGp            .STRINGZ "The number is prime"
MSGnp           .STRINGZ "The number is not prime"
            .END
```

Gruppe 7



Gruppe 7

Opgaver 5

```
; Kombinerer A2, A3 og A4 og kører et uendeligt loop
      .ORIG x3000
INF    JSR readS
      AND R1,R1,#0
      ADD R1,R1,R0
      JSR isPrime
      JSR resultS
      BR INF
      HALT
;
; Subrutine der skriver til brugeren og tager imod de 2 tal og gemmer dem i R0
readS   ST R7,RETURNreadS ; Gemmer hvor readS skal returnere i RETURN
      LEA R0,MSGr ; Henter adressen til beskeden der skal skrives, til R0
      PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
      ;
      GETC ; TRAP x20 ; Tager imod et tegn i R0
      OUT ; TRAP x21 ; Skriver tegner ud igen (echo)
      ;
      LD R4,fASCII ; Sætter R4 til -48
      ADD R0,R0,R4 ; Trækker 48 fra input, for at konvertere fra ASCII til tal
      AND R4,R4,#0 ; Sætter R4 til 0
      ;
      AND R1,R1,#0 ; Sætter R1 til 0
      ADD R1,R1,#10 ; Sætter R1 til 10
      JSR MULT ; Kører subrutine MULT (R0 * R1) = R2
      ;
      GETC ; TRAP x20 ; Tager imod et tegn i R0
```

```
        OUT ; TRAP x21 ; Skriver tegner ud igen (echo)
        ;
        LD R4,fASCII ; Sætter R4 til -48
        ADD R0,R0,R4 ; Trækker 48 fra input, for at konvertere fra ASCII til tal
        AND R4,R4,#0 ; Sætter R4 til 0
        ;
        ADD R0,R2,R0 ; Lægger input til R2 (første tal * 10)
        ;
        LD R7,RETURNreadS ; Sætter R7 tilbage til det readS skal returnere til
        RET ; JMP R7 ; Returnerer
;
; Tjekker værdien i R1, for om det er primtal
isPrime ST R7,RETURNisPrime ; Gemmer hvor isPrime skal returnere i RETURN
        LEA R3, PRIMES ; Henter adressen til første primtal til R3
LOOP    LDR R0,R3,#0 ; Henter værdien fra den adresse i memory, R3 pejer på
        BRz DONE ; Hvis værdien er 0, gå til DONE
        NOT R0,R0 ; Invertering af R0
        ADD R0,R0,#1
        ADD R0,R0,R1 ; Hvis forskellen på de to tal (R1 og R0) er 0, så er de ens og R1 er et primtal
        BRz FOUND ; Hvis den forrige handling gav 0, er de to tal ens og det givne er et primtal, gå til
FOUND
        ADD R3,R3,#1 ; Ret R3 til at peje på næste felt i hukommelsen
        BR LOOP ; Hvis det ikke var et primtal, gå til LOOP, igen
FOUND   AND R0,R0,#1 ; Sæt R0 til 0
        ADD R0,R0,#1 ; Sæt R0 til 1
DONE    LD R7,RETURNisPrime ; Sætter R7 tilbage til det isPrime skal returnere til
        RET ; JMP R7 ; Returnerer
;
; Læser R0, hvis der står 0 Skrives MSGnp, ellers skrives MSGp
```

```
resultS ST R7,RETURNresultS ; Gemmer hvor resultS skal returnere i RETURNresultS
        ADD R0,R0,#0 ; Lægger 0 til R0
        BRz NOPE ; Hvis 0 + R0 er lig 0, så stod der 0, gå til NOPE
        BR PRIM ; Ellers stod der ikke 0, gå til PRIM
NOPE    LEA R0,MSGnp ; Henter adressen til beskeden der skal skrives, til R0
        PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
        BR DONES ; Gå til DONES
PRIM    LEA R0,MSGp ; Henter adressen til beskeden der skal skrives, til R0
        PUTS ; TRAP x22 ; Skriver beskeden ud fra R0
DONES   LD R7,RETURNresultS ; Sætter R7 tilbage til det resultS skal returnere til
        RET ; JMP R7 ; Returnerer
;
; Subrutine som "ganger" 2 tal
; Det ene tal læses fra R0, det andet fra R1, resultatet findes i R2
; Antager at begge tal er positive
MULT    AND R2,R2,#0 ; Sætter R2 til 0
        ADD R2,R0,#0 ; Gemmer det originale tal fra R0 i R2
        BR MULTI ; Sætter "gange" igang
MULTI   ADD R1,R1,#-1 ; Tæller R1 en ned
        BRnz DONEM ; Hvis R1 nu er 0 eller negativ, stop
        ADD R2,R2,R0 ; Tæller R2 op med R0
        BR MULTI ; Går til MULTI igen
DONEM   AND R1,R1,#0 ; Sætter R1 til 0
        RET ; JMP R7 ; Returnerer
;
; Vores variabler
RETURNreadS     .BLKW #1 ; Gemmer adressen, som readS skal returnere til når den er færdig
RETURNisPrime   .BLKW #1 ; Gemmer adressen, som isPrime skal returnere til når den er færdig
RETURNresultS   .BLKW #1 ; Gemmer adressen, som resultS skal returnere til når den er færdig
```

```
fASCII          .FILL #-48 ; Gemmer værdien -48, så et et-ciftret til kan konverteres fra ASCII til binær
MSGr            .STRINGZ "Input a 2 digit decimal number:" ; Beskeden readS skriver ud
PRIMES          .FILL #2
                .FILL #3
                .FILL #5
                .FILL #7
                .FILL #11
                .FILL #13
                .FILL #17
                .FILL #19
                .FILL #23
                .FILL #29
                .FILL #31
                .FILL #37
                .FILL #41
                .FILL #43
                .FILL #47
                .FILL #53
                .FILL #59
                .FILL #61
                .FILL #67
                .FILL #71
                .FILL #73
                .FILL #79
                .FILL #83
                .FILL #89
                .FILL #97
PRIMELAST       .FILL #0
MSGp            .STRINGZ "\nThe number is prime\n"
```

```
MSGnp           .STRINGZ "\nThe number is not prime\n"

                .END
```