```
;
; Factorial example
; returns number! in r10
;

   .data
number: .word 10
title:  .asciiz "factorial program n= "

CONTROL: .word32 0x10000
DATA:    .word32 0x10008

        .text

            r21,CONTROL(r0)
            r22,DATA(r0)
              r24,r0,4      ; ascii output
              r1,r0,title
           r1,(r22)
           r24,(r21)

              r24,r0,8     ; read input
           r24,(r21)
           r1,(r22)

start:        r29,r0,0x80   ; position a stack in data memory, use r29 as stack pointer
            factorial
              r24,r0,1      ; integer output
           r10,(r22)
           r24,(r21)


;
; parameter passed in r1, return value in r10
;

factorial:       r10,r1,2
             r10,out      ; set r10=1 and return if r1=1

           r31,(r29)
              r29,r29,8   ; push return address onto stack

           r1,(r29)
              r29,r29,8   ; push r1 on stack

              r1,r1,-1    ; r1 = r1-1
            factorial     ; recurse...

             r4,r0,r10
              r29,r29,-8
           r3,(r29)       ; pop n off the stack

              r3,r3,r4    ; multiply r1 x factorial(r1-1)
             r10,r0,r3    ; move product r3 to r10

              r29,r29,-8  ; pop return address
           r31,0(r29)
out:       r31
```

```
;
; Example IO program
;

      .data
int:  .word 0xF9876543987625aa   ; a 64-bit integer
mes:    .asciiz "Hello World\n"     ; the message
key:  .asciiz "Press any key to exit\n"

dub:    .double 32.786        ; a double
x:      .byte 0            ; coordinates of a point
y:      .byte 0
col:  .byte 255,0,255,0    ; the colour magenta

;
; Memory Mapped I/O area
;
; Address of CONTROL and DATA registers
;
; Set CONTROL = 1, Set DATA to Unsigned Integer to be output
; Set CONTROL = 2, Set DATA to Signed Integer to be output
; Set CONTROL = 3, Set DATA to Floating Point to be output
; Set CONTROL = 4, Set DATA to address of string to be output
; Set CONTROL = 5, Set DATA+5 to x coordinate, DATA+4 to y coordinate, and DATA to RGB
colour to be output
; Set CONTROL = 6, Clears the terminal screen
; Set CONTROL = 7, Clears the graphics screen
; Set CONTROL = 8, read the DATA (either an integer or a floating-point) from the
keyboard
; Set CONTROL = 9, read one byte from DATA, no character echo.
;

CONTROL: .word32 0x10000
DATA:    .word32 0x10008

        .text

       $t8,DATA($zero)  ; $t8 = address of DATA register
       $t9,CONTROL($zero)  ; $t9 = address of CONTROL register

         $v0,$zero,1 ; set for unsigned integer output
      $t1,int($zero)
      $t1,0($t8)           ; write integer to DATA register
      $v0,0($t9)           ; write to CONTROL register and make it happen

         $v0,$zero,2 ; set for signed integer output
      $t1,int($zero)
      $t1,0($t8)           ; write integer to DATA register
      $v0,0($t9)           ; write to CONTROL register and make it happen

         $v0,$zero,3 ; set for double output
       f1,dub($zero)
       f1,0($t8)     ; write double to DATA register
      $v0,0($t9)     ; write to CONTROL register and make it happen

         $v0,$zero,4       ; set for ascii output
         $t1,$zero,mes
      $t1,0($t8)           ; write address of message to DATA register
      $v0,0($t9)           ; make it happen

         $v0,$zero,5       ; set for graphics output
       $t2,x($zero)
      $t2,5($t8)     ; store x in DATA+5
       $t3,y($zero)
```

```
      $t3,4($t8)     ; store y in DATA+4
       $t1,col($zero)
      $t1,0($t8)          ; store colour in DATA
      $v0,0($t9)     ; draw it

;
; Now draw a line!
;
         $t4,$zero,49

again:         $t2,$t2,1      ; increment x
      $t2,5($t8)     ; store x in DATA+5

         $t3,$t3,1      ; increment y
      $t3,4($t8)     ; store y in DATA+4

      $v0,0($t9)     ; draw it

         $t4,$t4,-1
        $t4,again

;
; Finish off
;

         $v0,$zero,4       ; set for ascii output
         $t1,$zero,key
      $t1,0($t8)           ; write address of message to DATA register
      $v0,0($t9)           ; "Press any key to exit"

         $v0,$zero,9 ;
      $v0,0($t9)     ; Wait for a key press...
      $t1,0($t8)     ;

         $v0,$zero,6       ;
      $v0,0($t9)     ; clear the terminal screen
         $v0,$zero,7       ;
      $v0,0($t9)     ; clear the graphics screen
```

```
#
# Insertion sort algorithm
# See http://www.cs.ubc.ca/spider/harrison/Java/InsertionSortAlgorithm.java.html
# Note use of MIPS register pseudo-names, and # for comments
#
      .data
array: .word  0x4F6961869342DC99,0x7A0B67101C85D9EE,0x5EF87A2B37CA911D,0x47EF58E8B7E01DD9
       .word  0x79A74EAB20CB53C9,0x6D26753D06F8E483,0x70F313AF126C0B47,0x745232A4035F1EF5
       .word  0x46036BDDE8D095FD,0x4DE3F1D89B5A43EA,0x5279659D102EABBA,0x4496CDA949E29089
       .word  0x6D594E2009B7D04A,0x4CE57C0D55905DE5,0x4115A0AC78A1848B,0x5051DAA648B3BDA6
       .word  0x71C3730CE11593C0,0x425A9FAE68370FC5,0x6B265F8485354426,0x4E935A849C713D01
       .word  0x773110588E5170D7,0x5B133F183803A780,0x49A52D37525C362C,0x4A0C150C49D8A123
       .word  0x7962EC77A41FB066,0x5D3A087AF3417D04,0x7076F96031DC3B2E,0x404EC3D105D02FDD
       .word  0x5484F578189A7A8B,0x65EA86F819037E03,0x4367E6F2AE35B27A,0x63C1CF869394DB43
       .word  0x59421109269E583C,0x6B9F1B529C8598EF,0x4C877DCC129AF1BD,0x58401EDBF56D884F
       .word  0x754C5475E3F8BFCF,0x1111111111111111,0x786213BFF3FAE203,0x53F6C77223F8D4B5
       .word  0x5304A0C74815DFBF,0x701BFCF2B7E84DED,0x72C3DEDE1BA476AD,0x557C05371C0A436C
       .word  0x741CECCDBAEBBBB3,0x577156E9E5C72202,0x641D1FEFF6E59822,0x623B6D2C45E6AFC6
       .word  0x6976994C37A754F0,0x4CE48C6E6963A020,0x4EDDBCD1CF3CD3AC,0x706AAA8FC1AE08E4
       .word  0x674DE62D8E4ACB59,0x791423B583AF7749,0x4589009608F70D0A,0x55159D9A3430F238
       .word  0x70BD250BE3048518,0x6D1B60128C603831,0x5397AB7F0E29CEE8,0x58EF0102374A9A97
       .word  0x625D9DBD94D1E2D1,0x5E8439437165FDF6,0x4F621F3A37353266,0x426B3ACC1149F170
       .word  0x59D789FA7FA3F476,0x4C4353E0D30D6D4B,0x492F120FA02F0B1C,0x720DFD78A97CFF59
       .word  0x5BC2140E14551D39,0x68718C039D4656B9,0x7FFFFFFFFFFFFFFF,0x48F63330CBC9A739
       .word  0x6E47955AFD5F8C20,0x44972B6AD10F9D2A,0x46578121CA1151A1,0x46281A1E7672B320
       .word  0x4094CC803E05BD98,0x5FF5B63C7812A363,0x6AF41E217F7612C5,0x4B7B4452B1E208AC
       .word  0x750F8A67FA5E72E4,0x51C8ECF29B5E8AD1,0x580550353D81B486,0x668CD4C5F3970ABF
       .word  0x480BEE00A16715AD,0x4888D5AC9EE02467,0x77C3DDBA62669040,0x48D55CDF7F706867
       .word  0x720670341FE6E445,0x6CAE4383191C2CC9,0x4F9E28BAD0270344,0x46DAD4328A8A3979
       .word  0x55B7AEB598729716,0x76D0F139C5FF97C5,0x4B876EB39C2DC380,0x781ADC2AD91E6FDF
       .word  0x53BDEAF8F4AA0625,0x624D7EA5B9A73772,0x75A02137A787850D,0x4259BDE1C33A32E6

len:  .word 100


        .text
              $t0,$zero,8  # $t0 = i = 8
           $t1,len($zero)  # $t1 = len
             $t1,$t1,3     # $t1 = len*8
for:        $t2,$t0,$t1    # i < len?
             $t2,out       # yes - exit
             $t3,$zero,$t0 # $t3=j=i
           $t4,array($t0)  # $t4=B=a[i]
loop:       $t2,$zero,$t3  # j>0 ?
             $t2,over      # no -exit
              $t5,$t3,-8   # $t5=j-1
           $t6,array($t5)  # get $t6=a[j-1]
            $t2,$t6,$t4    # >B ?
             $t2,over
           $t6,array($t3)  # a[j]=a[j-1]
             $t3,$zero,$t5 # j--
          loop

over:      $t4,array($t3)  # a[j] = B
              $t0,$t0,8    # i++
          for
out:
      .data
A:  .word 10
B:  .word 8
C:  .word 0
CR:  .word32 0x10000
DR:  .word32 0x10008


      .text
main:
         r4,A(r0)
         r5,B(r0)
           r3,r4,r5
         r3,C(r0)

          r1,CR(r0)  ;Control Register
          r2,DR(r0)  ;Data Register
            r10,r0,1
         r3,(r2)     ;r3 output..
         r10,(r1)    ;.. to screen


```

```
; MIPs version of series.cpp

       .data
x:     .double 0.6
s:     .double 1.0

       .text
             F0,x(R0)     ;F0=x
             F2,x(R0)     ;F2=y
             F4,s(R0)     ;F4=s
             R1,R0,2      ;n=2

Loop:          R1,F11    ; convert R1...
               F6,F11    ; to double precision
               F8,F6,F2
               F4,F4,F8
               F2,F0,F2
               R1,R1,1
               R2,R1,22
               R2,Loop
; Result in F8





```

```
;
; Unsigned multiplication of two 64-bit numbers on MIPS64 processor
; Result is 128-bits w=x*y
;
        .data
x:      .word 0xFFFFFFFFFFFFFFFF
y:      .word 0xFFFFFFFFFFFFFFFF
w:      .word 0,0

        .text

start:      mul         ; call subroutine



mul:          r1,r0,64   ; r1=64 bits
              r5,r0,63   ; for shifting
              r2,r0,r0   ; r2=0
              r10,r0,r0  ; r10=0
           r3,x(r0)      ; r3=x
           r4,y(r0)      ; r4=y
             r9,r3,1     ; check LSB of x
        r9,r0,r9    ;; negate it
             r3,r3,1     ; and then shift it right
again:  ;daddu r6,r0,r0
     ;;   movn r6,r4,r9
            r6,r4,r9
              r2,r2,r6
             r7,r2,r6    ; did it overflow?
              r7,r7,r5   ; catch overflowed bit
             r10,r2,1    ; get LSB of r2 ..
              r10,r10,r5 ; .. becomes MSB of r3
             r2,r2,1     ; 64-bit shift of r2,r3
           r2,r2,r7      ; or in overflowed bit
             r9,r3,1     ; catch LSB
        r9,r0,r9    ;; negate it
              r1,r1,-1   ; here to avoid stall
             r3,r3,1
           r3,r3,r10     ; shift it right, and set MSB
             r1,again

           r2,w(r0)     ; store answer
           r3,w+8(r0)
           r31

```

```
;
; Hailstone numbers iteration
; If number is odd, multiply by 3 and add 1
; If number is even, divide it by 2
; repeat this iteration until number is 1
; What is the maximum value during this process?
;
        .data
max:    .word 0         ; max number so far

title:  .asciiz "Hailstone Numbers\n"
prompt:  .asciiz "Number= "
str:    .asciiz "Maximum= "

;
; Memory Mapped I/O area
;
; Address of CONTROL and DATA registers
;
; Set CONTROL = 1, Set DATA to Unsigned Integer to be output
; Set CONTROL = 2, Set DATA to Signed Integer to be output
; Set CONTROL = 3, Set DATA to Floating Point to be output
; Set CONTROL = 4, Set DATA to address of string to be output
; Set CONTROL = 5, Set DATA+5 to x coordinate, DATA+4 to y coordinate, and DATA to RGB colour to be output
; Set CONTROL = 6, Clears the terminal screen
; Set CONTROL = 7, Clears the graphics screen
; Set CONTROL = 8, read the DATA (either an integer or a floating-point) from the keyboard
; Set CONTROL = 9, read one byte from DATA, no character echo.
;

CONTROL: .word32 0x10000
DATA:    .word32 0x10008

        .text
            r8,DATA(r0)    ; get data
            r9,CONTROL(r0) ; and control registers

              r11,r0,4     ; set for string output

              r1,r0,title  ; get title address
           r1,(r8)         ; print title
           r11,(r9)

              r1,r0,prompt ; get prompt address
           r1,0(r8)        ; print prompt
           r11,0(r9)

              r1,$zero,8   ; set for input
           r1,0(r9)     ; get the hailstone start number
           r1,0(r8)
           r1,max(r0)    ; first maximum
              r12,r0,1   ; set for integer output

loop:        r3,r1,1     ; test odd or even
             r3,even
odd:          r2,r1,r1    ; times 2
             r1,r2,r1    ; times 3
              r1,r1,1    ; plus 1
          over
even:        r1,r1,1     ; divide by 2
over:      r1,(r8)
           r12,(r9)      ; display it
```

```
            r4,max(r0)
             r3,r4,r1     ; compare with max
              r3,skip
            r1,max(r0)    ; new maximum?
skip:         r3,r1,2     ; test for finished
              r3,loop

            r2,max(r0)    ; get max
               r1,r0,str  ; get address of "Maximum= " string
            r1,(r8)       ; display "Maximum"
            r11,(r9)
            r2,(r8)       ; output maximum
            r12,(r9)


```

```
;
; Example IO program
;

      .data
int:  .word 0xF9876543987625aa   ; a 64-bit integer
mes:    .asciiz "Hello World\n"     ; the message
key:  .asciiz "Press any key to exit\n"

dub:    .double 32.786        ; a double
x:      .byte 0            ; coordinates of a point
y:      .byte 0
col:  .byte 255,0,255,0    ; the colour magenta

;
; Memory Mapped I/O area
;
; Address of CONTROL and DATA registers
;
; Set CONTROL = 1, Set DATA to Unsigned Integer to be output
; Set CONTROL = 2, Set DATA to Signed Integer to be output
; Set CONTROL = 3, Set DATA to Floating Point to be output
; Set CONTROL = 4, Set DATA to address of string to be output
; Set CONTROL = 5, Set DATA+5 to x coordinate, DATA+4 to y coordinate, and DATA to RGB
colour to be output
; Set CONTROL = 6, Clears the terminal screen
; Set CONTROL = 7, Clears the graphics screen
; Set CONTROL = 8, read the DATA (either an integer or a floating-point) from the
keyboard
; Set CONTROL = 9, read one byte from DATA, no character echo.
;

CONTROL: .word32 0x10000
DATA:    .word32 0x10008

        .text

       $t8,DATA($zero)  ; $t8 = address of DATA register
       $t9,CONTROL($zero)  ; $t9 = address of CONTROL register

         $v0,$zero,1 ; set for unsigned integer output
      $t1,int($zero)
      $t1,0($t8)           ; write integer to DATA register
      $v0,0($t9)           ; write to CONTROL register and make it happen

         $v0,$zero,2 ; set for signed integer output
      $t1,int($zero)
      $t1,0($t8)           ; write integer to DATA register
      $v0,0($t9)           ; write to CONTROL register and make it happen

         $v0,$zero,3 ; set for double output
       f1,dub($zero)
       f1,0($t8)     ; write double to DATA register
      $v0,0($t9)     ; write to CONTROL register and make it happen

         $v0,$zero,4       ; set for ascii output
         $t1,$zero,mes
      $t1,0($t8)           ; write address of message to DATA register
      $v0,0($t9)           ; make it happen

         $v0,$zero,5       ; set for graphics output
       $t2,x($zero)
      $t2,5($t8)     ; store x in DATA+5
       $t3,y($zero)
```

```
      $t3,4($t8)     ; store y in DATA+4
       $t1,col($zero)
      $t1,0($t8)          ; store colour in DATA
      $v0,0($t9)     ; draw it

;
; Now draw a line!
;
         $t4,$zero,49

again:         $t2,$t2,1      ; increment x
      $t2,5($t8)     ; store x in DATA+5

         $t3,$t3,1      ; increment y
      $t3,4($t8)     ; store y in DATA+4

      $v0,0($t9)     ; draw it

         $t4,$t4,-1
        $t4,again

;
; Finish off
;

         $v0,$zero,4       ; set for ascii output
         $t1,$zero,key
      $t1,0($t8)           ; write address of message to DATA register
      $v0,0($t9)           ; "Press any key to exit"

         $v0,$zero,9 ;
      $v0,0($t9)     ; Wait for a key press...
      $t1,0($t8)     ;

         $v0,$zero,6       ;
      $v0,0($t9)     ; clear the terminal screen
         $v0,$zero,7       ;
      $v0,0($t9)     ; clear the graphics screen


```