```
\ Challenge: Compute PI by your favorite method(s). Consider,
\   1) Integer to your stack width,
\   2) Floating point to your processorâ\200\231s limit, and/or
\   3) Integer beyond your stack width.

\ Approach: Use the spigot alogorthm described in
\
\   Rabinowitz, Stanley and Wagon, Stan (1995) â\200\234A Spigot Algorithm for the
\   Digits of Ï\200,â\200\235 The American Mathematical Monthly, 102(3), pp. 195â\200\223203.
\   doi: 10.2307/2975006

\ The base 10 digits of pi are 3.14159... . This can be represented as:
\
\   pi = 3 + 1/10 (1 + 1/10 (4 + 1/10 (1 + 1/10 (5 + 1/10 (9 + 1/10 ( ... ))))))

\ One of many series for pi can be rewritten as:
\
\   pi = 2 + 1/3 (2 + 2/5 (2 + 3/7 (2 + 4/9 (...))))

\ The spigot algorithm converts (2, 2, 2, 2, ...) with mixed base
\ (1/3, 2/5, 3/7, 4/9, ...) into (3, 1, 4, 1, 5, 9, ...) base 1/10.


\ Simple state machine skips the first digit, prints the second digit
\ followed by a decimal point, and prints all subsequent digits in groups
\ of 10.

variable digit._word

: digit. ( n -- ) digit._word @ execute digit._word ! ;

defer decimal_digit_0. ( n -- addr )

: decimal_digit_9. ( n -- addr ) 1 .r space ['] decimal_digit_0. ;
: decimal_digit_8. ( n -- addr ) 1 .r ['] decimal_digit_9. ;
: decimal_digit_7. ( n -- addr ) 1 .r ['] decimal_digit_8. ;
: decimal_digit_6. ( n -- addr ) 1 .r ['] decimal_digit_7. ;
: decimal_digit_5. ( n -- addr ) 1 .r ['] decimal_digit_6. ;
: decimal_digit_4. ( n -- addr ) 1 .r ['] decimal_digit_5. ;
: decimal_digit_3. ( n -- addr ) 1 .r ['] decimal_digit_4. ;
: decimal_digit_2. ( n -- addr ) 1 .r ['] decimal_digit_3. ;
: decimal_digit_1. ( n -- addr ) 1 .r ['] decimal_digit_2. ;
:noname ( n -- addr ) 1 .r ['] decimal_digit_1. ;
is decimal_digit_0.
: first_digit. ( n -- addr ) 1 .r ." ." ['] decimal_digit_0. ;
: skip_digit. ( n -- addr ) drop ['] first_digit. ;
: reset_digit. ( -- ) ['] skip_digit. digit._word ! ;


\ Use default number of digits or override from command line.

: override_from_args ( default --- n )
    1 arg 2dup 0 0 d<>
    if s>number?
        if d>s swap drop
        else 2drop
        endif
    else 2drop
    endif ;

101 override_from_args  constant NUM_DIGITS     \ number of digits to report
```

```
\ Provide an array to hold the (10n/3 + 1) remainders needed to generate
\ the selected number of digits.

: :array ( n "name" -- )
    create 1+ cells allot
    does> ( i -- addr ) swap cells + ;

NUM_DIGITS 10 3 */ 1+   constant NUM_PREDIGITS  \ number of predigits
NUM_PREDIGITS :array predigits


\ An output digit queue allows carries from later remainders.
\ Queue consists of a single non-nine predigit and a count of nines.

variable nines          \ number of nines seen
variable predigit       \ digit before nines analysis

: initialize_predigits ( -- )
    NUM_PREDIGITS 1+ 1 u+do 2 i predigits ! loop ;

: reset_predigit_queue ( q -- )
    predigit !
    0 nines ! ;


\ Adjust predigits, which range from 0 to 10. Carry tens to earlier predigits.
\ Count nines and queue other predigits to prepare for a carry.

: flush_nines ( n -- )
    nines @ 0 ?do dup digit. loop drop ;

: predigit. ( -- )
    predigit @ digit. ;

: count_nines ( -- ) 1 nines +! ;

: carry_and_release ( -- )
    predigit @ 1+ digit.
    0 flush_nines
    0 reset_predigit_queue ;

: release ( q --- )
    predigit.
    9 flush_nines
    reset_predigit_queue ;


: adjust_predigit ( q -- ? )
    dup case
        9 of drop count_nines endof
        10 of drop carry_and_release endof
        release
    endcase ;


\ Generate some number of pi's digits by reducing predigit remainders
\ and adjusting the resulting predigit.


: reduce ( q -- q )
    0 NUM_PREDIGITS -do
       i * i predigits @ 10 * +     \ x = ...
       i 2* 1- /mod swap i predigits !
```

```
    1 -loop ;

: get_predigit ( q -- q )
    10 /mod swap 1 predigits ! ;

: next_digit. ( -- n )
    0 reduce
    get_predigit
    adjust_predigit ;

: pi. ( -- )
    reset_digit.
    initialize_predigits
    0 reset_predigit_queue
    NUM_DIGITS 0 ?do next_digit. loop
    0 release ;

pi. cr

bye

\ vim: tabstop=8 expandtab shiftwidth=4 softtabstop=4 autoindent
```

```
pi@raspberrypi:~/forth/svfig-challenge-2 $ gforth pi.fs 1001
3.1415926535 8979323846 2643383279 5028841971 6939937510 5820974944
5923078164 0628620899 8628034825 3421170679 8214808651 3282306647 0938446095
5058223172 5359408128 4811174502 8410270193 8521105559 6446229489 5493038196
4428810975 6659334461 2847564823 3786783165 2712019091 4564856692 3460348610
4543266482 1339360726 0249141273 7245870066 0631558817 4881520920 9628292540
9171536436 7892590360 0113305305 4882046652 1384146951 9415116094 3305727036
5759591953 0921861173 8193261179 3105118548 0744623799 6274956735 1885752724
8912279381 8301194912 9833673362 4406566430 8602139494 6395224737 1907021798
6094370277 0539217176 2931767523 8467481846 7669405132 0005681271 4526356082
7785771342 7577896091 7363717872 1468440901 2249534301 4654958537 1050792279
6892589235 4201995611 2129021960 8640344181 5981362977 4771309960 5187072113
4999999837 2978049951 0597317328 1609631859 5024459455 3469083026 4252230825
3344685035 2619311881 7101000313 7838752886 5875332083 8142061717 7669147303
5982534904 2875546873 1159562863 8823537875 9375195778 1857780532 1712268066
1300192787 6611195909 2164201989
```