Programs for the HP–28S

These programs offer weak emulation of some of the most fundamental of the Turbo Pascal programs discussed in the preceding two sections. If the HP–28S is to be used extensively, then one would want to enlarge the collection to make it more comprehensive. It would also be desirable to have programs that employ binary integer variables (unsigned 64-bit integers). Their use would allow calculations up to 10^{18}. You may find it convenient to place these programs in a user subdirectory called `NOTHY`, say.

CRT

Function Determines the intersection of two arithmetic progressions. That is, it expresses the simultaneous conditions $x \equiv a_1 \pmod{m_1}$, $x \equiv a_2 \pmod{m_2}$ as a single congruence $x \equiv a \pmod{m}$, if the intersection is non-empty. If the intersection is empty then the message `"No such number"` is returned.

Stack Action

$$\begin{array}{c} \vdots \\ a_1 \\ m_1 \\ a_2 \\ m_2 \end{array} \quad \longrightarrow \quad \begin{array}{c} \vdots \\ a \\ m \end{array}$$

Restrictions $|a_i| < 10^{11}$, $0 < m_i < 10^{11}$. If the intersection is non-empty but $m > 10^{11}$, then the message `"Bad mod"` is returned.

Algorithm Same as for the Turbo Pascal program of the same name.

Listing

```
≪→ a1 m1 a2 m2
  ≪ m2 'M' STO m1 a2 a1 m1 MOD DUP 'a1'
     STO - LNCN m1 * SETM m1 * a1 + M
  ≫
≫
```

Calls LNCN, MULT, SETM

Comments This program may be invoked $k-1$ times in order to find the intersection of k given arithmetic progressions.

FAC

Function FACtors n by trial division. Thus $n = p_1 p_2 \ldots p_r$ with $p_1 \le p_2 \le \ldots \le p_r$.

Stack Action

$$\begin{matrix}\vdots \\ n\end{matrix} \longrightarrow \begin{matrix}\vdots \\ p_1 \\ p_2 \\ \vdots \\ p_r\end{matrix}$$

Restrictions $0 < n < 10^{12}$

Algorithm Trial division. After powers of 2, 3, and 5 have been removed, trial divisors are taken from the reduced residue classes modulo 30.

Listing

```
≪ 2 TRIAL 1 + TRIAL 2 + TRIAL 2 +
  WHILE
    DUP2 DUP * ≥
    REPEAT
    TRIAL 4 + TRIAL 2 + TRIAL 4 + TRIAL 2 +
    TRIAL 4 + TRIAL 6 + TRIAL 2 + TRIAL 6 +
  END
  DROP DUP IF 1 == THEN DROP END
  440 .25 BEEP
≫
```

Calls TRIAL

See Also Rho

Comments If $n > 10^{12}$ then the factorization given is erroneous.

GCD

Function Calculates the Greatest Common Divisor g of two given integers b, c. If $b = c = 0$ then the error message `"(0,0) undefined."` is returned.

Stack Action

$$\begin{matrix}\vdots \\ b \\ c\end{matrix} \longrightarrow \begin{matrix}\vdots \\ g\end{matrix}$$

Restrictions $|b| < 10^{12}$, $|c| < 10^{12}$

Algorithm Euclidean algorithm.

Listing

```
≪→ b c
  ≪ b ABS 'b' STO c ABS 'c' STO
     IF b 0 == c 0 == AND
     THEN "(0,0) undefined." KILL END
     WHILE c 0 >
     REPEAT
       b c MOD c 'b' STO 'c' STO
     END
     b
  ≫
≫
```

Comments If $|b| > 10^{12}$ or $|c| > 10^{12}$ then the value returned is likely to be incorrect.

INVERSE

Function Takes a from the stack and returns $\overline{a}$ such that $0 < \overline{a} < m$ and $a\overline{a} \equiv 1 \pmod{m}$, if such a number exists. Otherwise it returns the error message, `"No such number"` and kills the program.

Stack Action

$$\begin{matrix} \vdots & & \vdots \\ a & \longrightarrow & \overline{a} \end{matrix}$$

Restrictions $|a| < 10^{11}$, $0 < m < 10^{11}$

Algorithm Extended Euclidean Algorithm.

Listing

```
≪→ a
  ≪ a 1 LNCN DROP ≫
≫
```

Calls LNCN, MULT

Comments The value of m is specified by using the program SETM.

LNCN

Function Given a LiNear CoNgruence $ax \equiv b \pmod{m}$, numbers x_0 and m_0 are found such that the solutions are precisely those x such that $x \equiv x_0 \pmod{m_0}$, if the congruence has a solution. Otherwise the error message `"No such number"` is returned, and the prgram is killed.

Stack Action

$$\begin{array}{ccc} \vdots & & \vdots \\ a & \longrightarrow & x_0 \\ b & & m_0 \end{array}$$

Restrictions $|a| < 10^{11}$, $|b| < 10^{11}$, $0 < m < 10^{11}$

Algorithm Extended Euclidean algorithm.

Listing

```
≪→ a b
  ≪ b M MOD 'b' STO M 1 0 0 → c u0 u1 q
    ≪ WHILE
        c 0 >
      REPEAT
        a a c / FLOOR DUP 'q' STO c * - c 'a' STO 'c' STO
        u0 u1 DUP 'u0' STO q * - 'u1' STO
      END
      b a / DUP DUP FLOOR
      IF > THEN DROP "No such number." KILL END
      M a / M → m1 m2
       ≪ m1 'M' STO
          IF u0 DUP 0 < THEN M + END
          MULT M m2 'M' STO
       ≫
    ≫
  ≫
≫
```

Calls MULT

Comments The value of m is specified by using the program SETM.

M

Function A stored variable, used as the modulus in congruence arithmetic.

Restrictions $0 < M < 10^{11}$

Comments The value of M is specified by the program SETM.

MULT

Function Given a and b, returns the unique number c such that $c \equiv ab \pmod{m}$ and $0 \le c < m$.

Stack Action

$$\begin{matrix} \vdots & & \\ a & \longrightarrow & \vdots \\ b & & c \end{matrix}$$

Restrictions $0 \le a \le m,\ 0 \le b \le m,\ 0 < m < 10^{11}$

Algorithm "Russian multiplication" base $g = [5 \cdot 10^{11}/m]$. See Problem 21 on p. 83. of the text.

Listing

```
≪→ x y
  ≪ 0 5E11 M / FLOOR
      → s g≪ WHILE y 0 >
             REPEAT
               y y g / FLOOR DUP 'y' STO
               g * - x * s + M MOD
               's' STO x g * M MOD 'x' STO
             END
             s
          ≫
    ≫
≫
```

P?

Function Determines whether a number m is prime.

Stack Action

$$\begin{matrix} & & \vdots \\ \vdots & \longrightarrow & m \\ & & \\ m & & \begin{matrix}\text{“is prime.”} \\ \text{“is composite.”} \\ \text{“is too large.”}\end{matrix} \end{matrix}$$

Restrictions $1 \leq m \leq 25 \cdot 10^9$

Algorithm Strong pseudoprime tests are performed until m is proved to be prime, or until m is found to be composite. Use is made of the known fact that if $1 \leq m \leq 25 \cdot 10^9$ and if m is a strong probable prime base 2, 3, 5, 7 then m is prime unless $m = 3{,}215{,}031{,}751$.

Listing

```
≪→ m
  ≪ IF
       m 25000000000 >
     THEN
       m "is too large."
     ELSE
       m SETM 2 SPSP
       IF
         0 ==
       THEN
         m "is composite."
       ELSE
         IF
           m 2047 <
         THEN
           m "is prime."
         ELSE
           3 SPSP
           IF
             0 ==
           THEN
             M "is composite."
           ELSE
IF
  m 1373653 <
THEN
  m "is prime."
ELSE
  5 SPSP
  IF
    0 ==
  THEN
    m "is composite."
  ELSE
    IF
      m 25326001 <
    THEN
      m "is prime."
    ELSE
```

```
                    7 SPSP
                    IF
                      0 == m 3215031751 == OR
                    THEN
                      m "is composite."
                    ELSE
                      m "is prime."
                    END
                  END
                END
              END
            END
          END
        END
      END
      526 .25 BEEP
    ≫
  ≫
```

Calls MULT, POWER, SETM, SPSP

POWER

Function Finds the unique number a such that $0 \le a < m$ and $a \equiv x^k \pmod{m}$.

Stack Action

$$\begin{array}{ccc} \vdots & & \vdots \\ x & \longrightarrow & \vdots \\ k & & a \end{array}$$

Restrictions $|x| < 10^{12}$, $0 \le k < 10^{12}$, $0 < m < 10^{11}$

Algorithm The exponent k is expanded in binary, and the base x is repeatedly squared.

Listing

```
≪→ x y
  ≪1 → p
    ≪ x M MOD 'x' STO
      WHILE y 0 >
      REPEAT
        y y 2 / FLOOR DUP 'y' STO 2 * -
        IF 1 == THEN p x MULT 'p' STO END
        x x MULT 'x' STO
      END
      p
```

≫
≫
≫

Calls MULT

Comments The value of m is specified by the program SETM.

QUIT

Function Returns the machine to the root directory.

Listing

```
≪ HOME ≫
```

RHO

Function The Pollard RHO method is used to search for factors of m, say $m = ab$. The numbers a, b found are not necessarily prime. This method should only be applied to numbers m that are already known to be composite.

Stack Action

$$\begin{matrix}\vdots\\ m\end{matrix} \longrightarrow \begin{matrix}\vdots\\ a\\ b\end{matrix}$$

Restrictions $0 < m < 10^{11}$

Algorithm The sequence $u_n \pmod m$ is generated by the recurrence $u_{n+1} \equiv u_n^2 + s \pmod m$, with the initial condition $u_0 = r$. The program takes $r = s = 1$. The quantity $(u_{2n} - u_n, m)$ is calculated for successive values of n, until it takes a value strictly between 1 and m. If it takes the value m then the program is terminated with the message `"Test inconclusive."` In such a case one should edit the program to use a different value of s. However, one should avoid $s = 0$, $s = -2$.

Listing

```
≪→ m
  ≪ 1 1 → r s≪ m DUP SETM r DUP 1
                WHILE
                  DUP 1 ==
                REPEAT
                  DROP DUP MULT s + SWAP DUP MULT
                  s + DUP MULT s + SWAP DUP2 - m GCD
                END
                3 ROLLD DROP2 DUP
                IF
```

```
      m ==
    THEN
      "Test inconclusive."
    ELSE
      DUP2 /
    END
    880 .25 BEEP
  ≫
≫
```

Calls MULT, SETM

SETM

Function Assigns a value to the variable M, used as the modulus in congruence arithmetic.

Stack Action

$$\begin{matrix}\vdots \\ M\end{matrix} \longrightarrow \vdots$$

Restrictions $0 < M < 10^{11}$

Listing

```
≪ DUP DUP DUP
   IF
     0 > SWAP 1E11 < AND
     SWAP DUP FLOOR == AND
   THEN
     'M' STO 0 FIX
   ELSE
     "BAD MOD" KILL
   END
≫
```

SPSP

Function The Strong PSeudoPrime test base a is applied to the number m. If it is found that m is a strong probable prime base a then the number 1 is returned. Otherwise m is proved to be composite, and the number 0 is returned.

Stack Action

$$\begin{matrix}\vdots \\ a\end{matrix} \longrightarrow \begin{matrix}\vdots \\ 0 \text{ or } 1\end{matrix}$$

Restrictions $1 < m < 10^{11}$

Algorithm The strong pseudoprime test, as explained on p. 78 of the text.

Listing

```
≪→ a
  ≪ IF
       M DUP DUP 2 / FLOOR 2 * == SWAP 2 > AND
     THEN
       0
     ELSE
       0 0 → j k
        ≪ M 1 -
          WHILE
            DUP DUP 2 / FLOOR DUP 3 ROLLD 2 * ==
          REPEAT
            SWAP DROP 1 k + 'k' STO
          END
          DROP a SWAP POWER DUP DUP
          IF
            1 == SWAP M 1 - == OR
          THEN
            DROP 1
          ELSE
            1 CF
            WHILE
              1 FC? j k 1 - < AND
            REPEAT
              DUP MULT 1 j + 'j' STO
              IF DUP 1 == THEN 1 SF DROP 0 END
              IF DUP M 1 - == THEN 1 SF DROP 1 END
              END
              IF 1 FC?C THEN DROP 0 END
            END
          ≫
     END
  ≫
≫
```

Calls MULT, POWER

Comments The value of m is specified by using the program SETM.

TRIAL

Function This is a subroutine used in the program FAC. It executes TRIAL divisions, so that the integer k is found such that $d^k \| n$.

Stack Action

$$\begin{array}{ccc} & & \vdots \\ & & d \\ \vdots & & \vdots \\ & \longrightarrow & d \\ n & & n/d^k \\ d & & d \end{array} \quad \Big\} k$$

Restrictions $0 < d < 10^{12}$, $0 < n < 10^{12}$

Listing

```
≪ WHILE
    DUP2 MOD 0 ==
  REPEAT
    DUP 3 ROLLD / OVER
  END
≫
```

Further Resources

The sources listed below have not been fully tested. They may contain bugs, and should be used at your own risk. All the software listed below can be obtained by using the file transfer program ftp. If you have the ftp software and a link to the internet, then you may connect to a remote host and retrieve software. To do this, type

```
ftp hostname
```

and then press the Return (or Enter) key. Here "hostname" is the electronic address of the remote host. Alternatively, you may type

```
ftp
```

which will give you the prompt `ftp>`, and then type

```
open hostname
```

If you are prompted to provide a login name, then type

```
anonymous
```

Otherwise, type

```
login anonymous
```

Then you will be prompted for a password. Type your own electronic address. In most cases, the software you want is in a subdirectory. To change to the subdirectory "pub", for example, type

`cd pub` or else `cd /pub`

You may have to change directories yet again, say to the subdirectory "msdos". You can accomplish both changes in one step by typing

```
cd /pub/msdos
```

If, in exploring subdirectories, you find that you have wandered too far down the directory tree, and want to return to the parent of the current directory, then type

```
cd ..
```

To ascertain the name of the current working directory on the remote machine, type

```
pwd
```

In any particular directory, you can obtain a list of the files and subdirectories by typing

`ls` or else `dir`

These commands are handy if you are unsure of the exact name of the file you wish to retrieve. Once you are in the appropriate subdirectory, and have determined the name of the file you wish to transfer, say "clint1.tex", type

```
get clint1.tex
```

If you wish to "get" all the files in the current directory, you can ask for a "multiple get" by typing

```
mget *
```

Text files can be transferred in the default ASCII mode, but most software must be transferred in binary. To prepare for binary transfers, issue the command

```
type binary
```

When you wish to terminate your ftp session, type

```
quit
```

Since the availability of software may change with time, the information provided below will gradually become inaccurate. If your system has the ARCHIE program, you may use it to locate sources of software. Try typing

```
archie pari
```

CLINT

This lab manual, and the Turbo Pascal programs that are meant to be used with it, can be obtained by anonymous ftp from ftp.math.lsa.umich.edu in the subdirectory /pub/clint. The following files are available:

FILENAME	**CONTENTS**
readme	general instructions
clint0.tex	title page of lab manual

clint1.tex	forematter (pages ii–x) of lab manual
clint2.tex	the labs (pages 1–68)
clint3.tex	program documentation (pages 69–106)
clint4.tex	programs for HP–28S, Further Resources
error5-1.tex	Known errors in Niven-Zuckerman-Montgomery (Fifth Edition, First Printing)
tpprgms2.exe	Turbo Pascal programs (archived)

The .tex files will run on any system equipped with the TEX software, but the Turbo Pascal programs run only under DOS on IBM-compatible PCs. Both source code and compiled code are provided. If you plan to alter the source code or write your own programs you will need the Borland Turbo Pascal compiler, or else some other Pascal compiler. The file tpprgms2.exe is a self-extracting archive file that compresses into 1.3M what would otherwise be 145 Turbo Pascal files occupying 2.37M. The command `type binary` *must* be given before this file is transferred. The other files should be transferred as ascii files (issue the command `type ascii`). Instructions for decompressing tpprgms2.exe are found in the readme file. Users are encouraged to edit the laboratories, and to customize the manual for their own use. If you are unable to obtain this material by ftp, write to

Mathematics Editor
John Wiley & Sons
605 Third Avenue
New York, NY 10158-0012

Upon receiving your request, you will be sent a printed copy of this lab manual and a 3.5 inch disk containing the tpprgms2.exe and readme files. For instructors only, the publisher will also provide a printed copy of the Solutions Manual for the Niven-Zuckerman-Montgomery text.

GP

GP is an interactive calculator based on the PARI library of routines. See PARI for more details.

Malm

Donald E. G. Malm has written a large number of instructive programs, some in UBASIC, others in Turbo Pascal. These files can be obtained by anonymous ftp. Connect to the address vela.acs.oakland.edu, and then change to the subdirectory /pub/numbdroid. These are text files, so transfers should be made with the type ascii. If you have difficulty obtaining these files, send an email message to malm@argo.acs.oakland.edu, or write to

Professor Donald E. G. Malm
Department of Mathematical Sciences
Oakland University
Rochester, MI 48309-4401

Numbers

Numbers (Version 2.01) is a program written in Turbo Pascal 5.0 for IBM-compatible PCs by Ivo Düntsch of the Mathematics Department, Universität Osnabruck, Moorlandstr. 59, 4500 Osnabruck, Germany. It offers a wide variety of routines, accessed by a menu hierarchy. The program is aimed at students rather than researchers, and is available free of charge for educational purposes by anonymous ftp from clione.t2.uni-osnabrueck.de in the directory publ/msdos/math. For further information see the review in the *AMS Notices* **39** (1992), 840–841, by Louis D. Grey.

PARI

PARI takes its name from Pascal arithmetic, although eventually the code was written in C and assembler. It does not offer symbolic manipulation to the extent of Maple, for example, but it provides powerful routines for use in number-theoretic investigations. It was designed by C. Batut, D. Bernardi, H. Cohen and M. Olivier, with researchers in mind. It runs on the SUN3, SPARC, Macintosh II, NeXT, Amiga, on 32-bit PCs running MSDOS or OS2. It is available by anonymous ftp from megrez.ceremab.u-bordeaux.fr and also from math.ucla.edu, in the subdirectory /pub/pari. The transfer must be made in binary. Version 1.38, released in September, 1993, differs from the former Version 1.37 in several important ways, most notably in the introduction of a package for computations in algebraic number fields. The package includes TEX files that produce a 130 page manual and a 30 tutorial. For more information, see the review of Q. Gouvêa, *AMS Notices* **38** (1991), 903–904.

SEQUENCES

N. J. A. Sloane's book *A Handbook of Integer Sequences*, Academic Press, New York, 1973 can now be queried by email. To submit sequences for examination, send an email message to `sequences@research.att.com`. Each line of your message should be of the following form:

```
lookup 4 9 16 25 36
lookup 3 5 8 13 21
```

If your message is not of this form, then in reply you will recevie an explanation of the correct notation. A second program is also available, for the purpose of trying to identify a given sequence. To use it, send an email message to `superseeker@research.att.com`. Only one request may be made per message; your message should consist of a single line of the form

```
lookup 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37
```

Start at the beginning, and give lots of terms. Negative entries are acceptable. The program will try *very* hard to find an explanation. The program invoked is complicated, and still under development. Because some serious computing is executed, users are limited to one query per hour. If you encounter a bug with the program, please report it to Neil Sloane at `njas@research.att.com`.

SIMATH

SIMATH is written in C, and offers many C-functions over algebraic structures such as arbitrarily long integers, rational numbers, floating point numbers, polynomials, Galois fields, matrices, elliptic curves, algebraic number fields, congruence arithmetic, etc. The package includes an interactive calculator, SIMCALC, that can access many of these routines, including arithmetic of finite fields and elliptic curves. In Version 3.8.1, released in March of 1994, includes an algorithm for calculating Groebner bases. The package runs successfully on a variety of machines, such as Sun 3/140 under SunOS 3.4, and Sun SPARCstation under SunOS 4.1.1. The package is available by anonymous ftp from ftp.math.uni-sb.de and from ftp.math.orst.edu

UBASIC

UBASIC is a high-precision version of BASIC for IBM-compatible PCs, written by Yuji Kida. It provides fast arithmetic for integers of up to 2600 digits, and is aimed at students of number theory. It comes with on-line help and sample programs. It is available free of charge by anonymous ftp from math.mps.ohio-state.edu in the subdirectory /pub/msdos/ubasic. The transfer must be made in binary. The main files are archived as .zip files. A suitable unzip program is provided in the same directory. The transfer must be made in binary. For further information see the reviews of Walter D. Neumann in the *AMS Notices* **36** (1989), 557–559; **38** (1991), 196–197. Version 8.74 was released in May, 1994. If you are using Version 8.30 or earlier and DOS5 or later then you should upgrade to avoid a memory-handling problem.