Please consider a binary integer register named ui, of unspecified length.

Please consider a binary integer register named base that can contain either the value 10 or the value 16.

Please consider the following (here suggested, not implemented at this time) commands encoded as if Unicode characters: where X and Y are each used to represent a herein unspecified hexadecimal character that would need to be suggested in a formal application for encoding of the characters in regular Unicode.

```
U+XY000 ui:=ui*base;
U+XY001 ui:=ui*base + 1;
U+XY002 ui:=ui*base + 2;
U+XY003 ui:=ui*base + 3;
U+XY004 ui:=ui*base + 4;
U+XY005 ui:=ui*base + 5;
U+XY006 ui:=ui*base + 6;
U+XY007 ui:=ui*base + 7;
U+XY008 ui:=ui*base + 8;
U+XY009 ui:=ui*base + 9;
U+XY00A ui:=ui*base + 10;
U+XY00B ui:=ui*base + 11;
U+XY00C ui:=ui*base + 12;
U+XY00D ui:=ui*base + 13;
U+XY00E ui:=ui*base + 14;
U+XY00F ui:=ui*base + 15;

U+XY010 ui:=0;
U+XY011 base:=10;
U+XY012 base:=16;
```

Thus a sequence of the character codes may be used to enter any non-negative integer number into the ui register, provided that the register is long enough to contain the number. The first character of the sequence needs to set the value of the base being used, unless the value of the base has been set previously and there is no scope for ambiguity. The sequence needs to set the value of ui to 0 before any digits are entered..

TO DO Formal names for the characters may be needed.

Please consider a binary integer register named ai, of unspecified length.

Please consider the following commands encoded as if Unicode characters, in the same manner as before.

```
U+XY013 ai:=0;
U+XY014 ai:=ui;
U+XY015 ui:=ai;
U+XY016 ai:=ai + ui;
U+XY017 ai:=ai - ui;
U+XY018 ai:=ai * ui;
U+XY019 ai:=ai / ui;
U+XY01A ai:=ai % ui;
```

Thus a sequence of the character codes may be used to carry out integer arithmetic operations.

Please consider a binary integer array named mi, each element of the array being a binary integer of unspecified length.

Please consider a binary integer register named j, of unspecified length.

Please consider the following commands encoded as if Unicode characters, in the same manner as before.

```
U+XY01B j:=ai;
U+XY01C j:=ui;
U+XY01D ai:=j;
U+XY01E ui:=j;

U+XY01F mi[j]:=ai;
U+XY020 ui:=mi[j];
```

Please note how storage into memory is from ai and loading from memory is into ui.

* * *

2.0

2.1 The link flag

Please consider a Boolean register named link that can have the value false or the value true.

Please consider the following commands encoded as if Unicode characters, in the same manner as before.

```
U+XY021 link:=false;
U+XY022 link:=true;
U+XY023 link:=not link;
U+XY024 link:=ai > 0;
U+XY025 link:=ai = 0;
U+XY026 link:=ai < 0;
U+XY027 link:=ai > ui;
U+XY028 link:=ai = ui;
U+XY029 link:=ai < ui;
```

2.2 The if statement

```
U+XY02A if
U+XY02B then
U+XY02C elsif
U+XY02D else
U+XY02E endif;
```

The five commands if, then, elsif, else, endif; are used in the following manner, where c stands for one or more commands computing a value of link and o stands for one or more commands performing processing.

if c then o elsif c then o elsif c then o else o endif;

2.2.1 The character if

The character if does nothing. It is used for computing relative_depth, which is described later in the description of the character then.

2.2.2 The character then

The character then has different effects depending upon the state of link when the character then is obeyed.

If the value of link is true it does nothing. If the value of link is false, the value of an internal variable named relative_depth is set to zero and then a forward search is to be made. The forward search is for the next occurrence of any of the characters elsif, else or endif; when the value of relative_depth is zero. Computation then continues from that character. However, during the forward search, whenever the character if is encountered the value of relative_depth is increased by 1 and whenever the character endif; is encountered and the value of relative_depth is above zero the value of depth is decreased by 1. In this manner, multiple if structures may be nested. The character then is not a marker.

2.2.3 The character elsif

The character elsif does nothing. It is a marker for a forward search from the character then.

2.2.4 The character else

The character else does nothing. It is a marker for a forward search from the character then.

2.2.5 The character endif;

The character endif; does nothing. It is a marker for a forward search from the character then. When encountered in a forward search from a character then and the value of relative_depth is above zero then the value of relative_depth is decreased by 1.

* * *

U+XY02F unused at present

2.3 The while statement

U+XY030 while
U+XY031 do
U+XY032 endwhite;

2.3.1 The character while

The character while does nothing. It is a marker for a reverse search from the character endwhile;. When encountered in a reverse search from the character endwhile; and the value of relative_depth is above zero, the value of relative_depth is decreased by 1.

MORE NEEDED. EXPLAIN THAT relative_depth is first set to one for a reverse search from the character endwhile;.

2.3.2 The character do

MORE NEEDED

2.3.3 The character endwhile;

MORE NEEDED

2.4 The repeat statement

```
U+XY033 repeat
U+XY034 until
U+XY035 endrepeat;
```

2.4.1 The character repeat

MORE NEEDED

2.4.2 The character until

MORE NEEDED

2.4.3 The character endrepeat;

MORE NEEDED