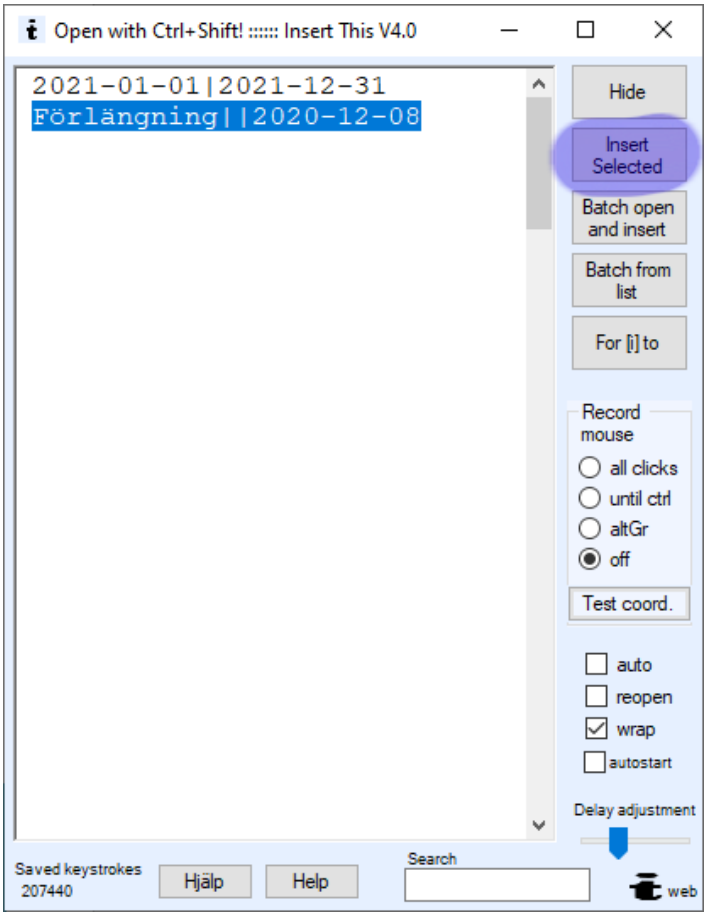
Insert This 4.0

A program by Anders Persson, to help fill in text and automate repetitive tasks on the computer.

Basic operation (Inserting text)

Open and hide the Insert-This window by pressing Ctrl+Shift or by clicking the program icon in the taskbar. (If you close the window on the "x", the program will be terminated).



The Insert-This window mostly consists of a text editor. There you type (or paste) what you want to be able to insert into other programs.

When the Insert-This window is open and active, you can insert the selected text into another program by pressing Enter (or AltGr+Shift) or by using the "Insert Selected" button.

The window will then be hidden and the text will be entered into the program and field where you were before you opened the Insert-This window.

(AltGr+Shift also inserts when the Insert-This window is hidden)

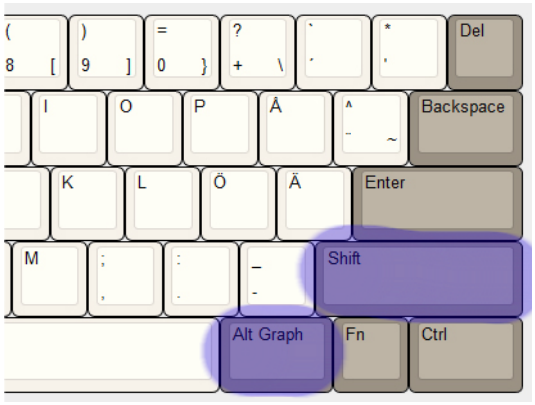
Example 1:

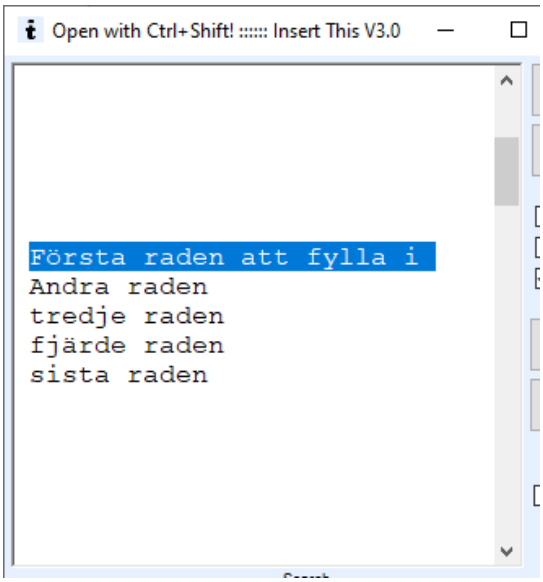
- **Hide the Insert-This window with Ctrl+Shift.**
- **Go to the program and field where you want to fill in something.**
- **Open "Insert This" with Ctrl+Shift.**
- **Select what you want to insert**, *for example with the arrow keys up and down (which marks a whole row at a time)*
- **Then press Enter.**

Example 2 (another way):

- **Open "Insert This" with Ctrl+Shift.**
- **Select what you want to insert.**
- **Go to the program and field where you want to fill in something.**
- **Press AltGr+Shift**

Filling in several fields in the correct order, method 1

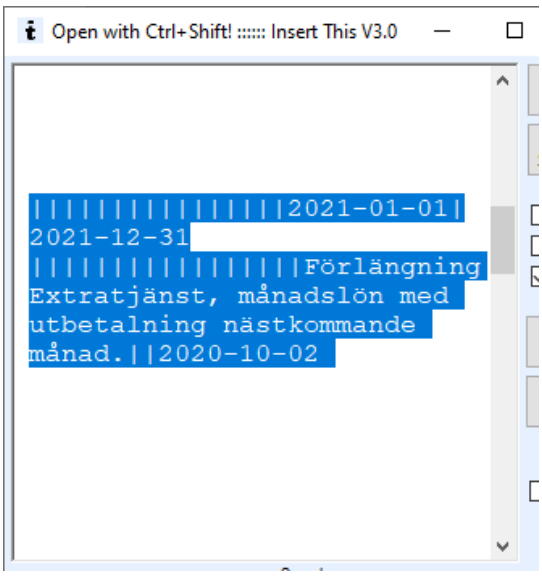




When the window is hidden, you can use AltGr+Shift to insert the next row **without opening the window.** Every time you do this, it automatically takes the next line in the Insert-this window without opening it. (But if the next line is empty, it stays on the last line of content).

This way you can make a list in the Insert-this window of what to insert in the correct order and then just use AltGr+Shift in field after field to insert the rows in the correct order.

When you open the window again, the first inserted line is highlighted again.



Method 2: Automatic tabs

If you type | dashes in the text, these will be translated into the tab key. You can thus fill in several fields with a single keystroke, eg field1|field2|field3

You can also use the character code {TAB}, eg {TAB 12} means 12 tabs. (See section "Character codes").

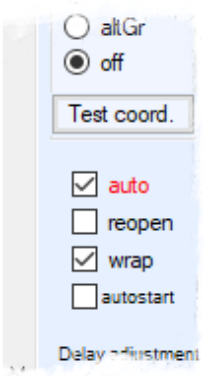
If what to be inserted is written alone on a row with an empty row below, you can use AltGr+Shift to insert the same thing over and over without having to open the insert-this window.

In the example above, the selection is only a single line, but since "Wrap" is checked, the line is wrapped in the window so that you can see the entire line.

You can use the cursor keys to go up and down and select an entire row.

You can use Shift and the cursor keys to select several lines at once or you can use the mouse for that.

The Home and End keys jump to the beginning and the end of the text, respectively.



If Auto is checked, the selected text is inserted as soon as it is selected, either with the mouse or with Shift and the cursor keys. When you release shift, it is inserted. You can also use the mouse to select an entire row by clicking in front of the start of the row. NOTE: As a precaution, auto shuts off automatically as soon as you edit the text in the Insert-this window.

Re-open: If this option is selected, the window will reopen immediately after something is inserted. This is mostly useful if you prefer to only use the mouse.

Wrap: Affects how the text is displayed in the window. Long lines are broken so that the whole line is visible.

This document is (largely) available if you click on "Help" in the Insert-this window, but then only the text is available, not the images. If you click on "Hjälp", you will instead get a Swedish help text.

Character codes

In the text you can use a number of codes that Microsoft has implemented. The following description is copied from their documentation.

The plus sign (+), caret (^), percent sign (%), tilde (~), and parentheses () have special meanings. To specify one of these characters, enclose it within braces ({}). For example, to specify the plus sign, use "{+}". To specify brace characters, use "{{}" and "{}}". Brackets ([ ]) have no special meaning, but you must enclose them in braces.

To specify characters that aren't displayed when you press a key, such as ENTER or TAB, and keys that represent actions rather than characters, use the codes in the following table.

Table 1

Key	***Code***
BACKSPACE	*{BACKSPACE}, {BS}, or {BKSP}*
BREAK	*{BREAK}*
CAPS LOCK	*{CAPSLOCK}*
DEL or DELETE	*{DELETE} or {DEL}*
DOWN ARROW	*{DOWN}*
END	*{END}*
ENTER	*{ENTER} or ~*
ESC	*{ESC}*
HELP	*{HELP}*
HOME	*{HOME}*
INS or INSERT	*{INSERT} or {INS}*
LEFT ARROW	*{LEFT}*
NUM LOCK	*{NUMLOCK}*
PAGE DOWN	*{PGDN}*
PAGE UP	*{PGUP}*
PRINT SCREEN	*{PRTSC} (reserved for future use)*
RIGHT ARROW	*{RIGHT}*
SCROLL LOCK	*{SCROLLLOCK}*
TAB	*{TAB}*
UP ARROW	*{UP}*
F1	*{F1}*
F2	*{F2}*
F3	*{F3}*

Table 1

Key	***Code***
F4	*{F4}*
F5	*{F5}*
F6	*{F6}*
F7	*{F7}*
F8	*{F8}*
F9	*{F9}*
F10	*{F10}*
F11	*{F11}*
F12	*{F12}*
F13	*{F13}*
F14	*{F14}*
F15	*{F15}*
F16	*{F16}*
Keypad add	*{ADD}*
Keypad subtract	*{SUBTRACT}*
Keypad multiply	*{MULTIPLY}*
Keypad divide	*{DIVIDE}*

To specify keys combined with any combination of the SHIFT, CTRL, and ALT keys, precede the key code with one or more of the following codes.

Table 2

Key	***Code***
SHIFT	+
CTRL	^
ALT	%

To specify that any combination of SHIFT, CTRL, and ALT should be held down while several other keys are pressed, enclose the code for those keys in parentheses. For example, to specify to hold down SHIFT while E and C are pressed, use "+(EC)". To specify to hold down SHIFT while E is pressed, followed by C without SHIFT, use "+EC".

To specify repeating keys, use the form {key number}. You must put a space between key and number. For example, {LEFT 42} means press the LEFT ARROW key 42 times; {h 10} means press H 10 times.

A related command is [A value] which types the character with the corresponding ascii value, eg [A 9] means the same thing as {TAB} or |.

Advanced automation: Introduction

Insert This V4.0 has added a number of things that can be used to create advanced automation:

I. Recording and playback of mouse clicks. The mouse clicks are saved as commands in the Insert-This text window.

II. "Batch open and insert": You select a number of files in a file dialog (eg all files in a folder). These files will then be opened one by one with their default application and the selected text in the Insert-This window will be executed for each of the files.

III. "Batch from list": This is a method that allows you to repeat a procedure based on a register (a list in a text file). Here you select a text file that contains tab-delimited data. (It can, for example, be exported from Excel). The selected text in the Insert-This window will then be executed for each row in the list and the contents of the list row will be entered by the command [C column_number]. (For example, [c 1] for the text in column 1).

IV. You can run a general loop where a variable is increased. (The "For [i] to" button).

V. Conditional control. Depending on a variety of conditions, including the contents of the clipboard, different sequences can be run through the [if], [else] and [endif] commands. There is also a [break] command that you can use.

These different possibilities will now be described in more detail.

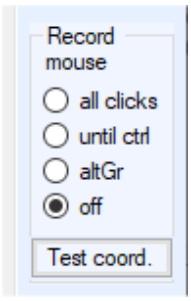
Recording and playback of mouse clicks

Warning: This is an advanced feature that must be used with caution. When you play back your recording, it will actually click on the screen at the recorded positions completely regardless of what is displayed on the screen. If you have a program on top other than the one you recorded on, it will click there instead, on what is displayed on the screen right now. If you have moved the window you are going to click in since you recorded the mouse click, it will not work either because then it will click where the window was before. If an unforeseen error message comes up from a program, it will not work either. You must be able to think logically and predict everything that will happen in order to be able to use this function.

To stop the execution of a sequence of mouse clicks, hold down the left and right mouse buttons at the same time until you get the message "Stopped by user".

You can record right-clicks, left-clicks and double-clicks, but not holding down a mouse button.

To record mouse clicks, first place the cursor in the Insert-This text window where you want to record one or more mouse clicks, then there are three different ways to record the mouse clicks:



All Clicks: All (recordable) clicks (as above) will be recorded until you select "Off" instead. Note that the click to select "Off" also is recorded, so you then have to edit that out manually.

Until Ctrl: All clicks are recorded until you press the Ctrl key. Then it switches to "Off".

AltGr: Only the clicks that takes place while you are holding down the AltGr button will be recorded. Note that AltGr may have some function in the program you are using, in which case this option is not recommended.

The recorded clicks will look something like this:
```
[L 1169 1060][L 1295 962][L 341 165][L 413 159]
```

R means right mouse button. L means left button. And the numbers after are the coordinates of the mouse click. For example, [L 300 200] means a click with the left mouse button at the coordinates X = 300 and Y = 200 pixels.

If you then select this text and press the "Insert" button, the mouse click will be performed.

If you want some text to be typed between the mouse clicks, you have to type it in the Insert-This window. Text is not recorded automatically in the same way as the mouse click. So if for example ".txt" should be typed after the first mouse click in the recording above, it should look like this:

```
[L 1169 1060].txt[L 1295 962][R 341 165][L 413 159]
```

Keep in mind that line breaks between mouse commands will also be inserted as text, so make sure that all mouse commands are on the same line if you don't want unwanted line breaks. It could be a good idea to have "Wrap" checked in the editor options so that you see all of the text.

Note: If some text is selected/marked when you start recording mouse clicks, it will NOT be overwritten by the mouse click. If that was your intention, delete the text first instead.

Wait a moment

Sometimes an extra delay is needed after clicking somewhere, for example for a program to open. Then you can use the command [W seconds] eg [W 4] to wait 4 seconds. The number of seconds does not have to be an integer, so for example [W 0.5] can be used to wait for half a second. Note that this delay is affected by the "Delay Adjustment" slider which affects all delays in Insert This. If you really want to wait the number of seconds you specify, you can instead use WF, eg [WF 1] to wait for one second regardless of how the "Delay Adjustment" control is set. (WF stands for Wait Force).

REM

If you want to write a comment in your program to help remember what a certain sequence does, use the command [rem your comment]. This command is completely skipped during execution. Rem means "remark".

Mouse clicks based on the current mouse position

The L+ and R+ commands can be used to move the mouse in relation to the current position and perform a mouse click there. For example, [L+ 0 16] will move the mouse 16 pixels down and then click the left mouse button while [R+ -8 0] will move the mouse 8 pixels to the left and then right-click.

Test coordinates

The "Test coord." button takes the first mouse coordinates in the selected text and puts the mouse there without clicking. The purpose of this is to test and see if a command clicks in the right place without it clicking there. Note that this function ignores which command is used for the coordinates, so coordinates for the commands L+ and R+ (above) will be seen as normal (absolute) coordinates instead.

Batch open and insert

With this function you can perform the same text change, mouse click playback (or a combination) on a number of similar files, all of which will be opened with their default application.

This is advanced, so please read this instruction carefully and then decide if you dare to do this or not.

Procedure:

First, make a backup copy of all of the files that you intend to modify, in case something goes wrong with your programming (and it probably will while you are testing it out). Make a copy of the entire folder.

Then you have to check that the program you want to use really is the default application for the file type in question. To do this, double-click one of the files in Windows Explorer. If the correct program doesn't open, you have to change the default application for the file type in Windows settings. (Search the Internet if you don't know how to do that).

Then create an appropriate change sequence in Insert This, which can be performed right after the file is opened. If your purpose is to change the file, then the sequence must of course contain commands to save the file at the end, perhaps with a new name. After that, you must also include something that closes the file again, otherwise you will have as many open programs as files in the end.

You can test your sequence by selecting the sequence in Insert This, then double-click a test file in Windows Explorer, and then press AltGr+Shift to execute the sequence.

Then test with just a few files in a test folder. For example 10 files. Select the sequence in Insert This and then press "Batch open and instert", select your test files, observe and check if the result was what you expected. While the files are opened one by one and the sequence is run for each of them, a green bar will go over the top of the screen to show the progress. If you notice that something goes wrong, you can hold down both the right and left mouse buttons at the same time. Then the whole thing is stopped. Hold down the buttons continuously until it is interrupted.

There is automatically a pause of a few seconds when each file is opened. If this pause is not enough for the file to open, you can set a wait command as the first command in your sequence. For example [WF 5] (see previous description).

Once you have completed a successful test with multiple files, you can run the sequence for all of the files you intended. But first make sure that you don't have any other program running that could interfere with the whole thing, for example by suddenly displaying a message on the screen.

Here is an example of what it might look like. Here is a sequence that changes two dates in a form in Word and then saves the file with a new name through a combination of mouse clicks and text:

```
[L 157 14][L 204 55][L 617 237]{TAB 16}2021-01-01|2021-12-31[L
29 48][L 62 249][L 566 194][w 1][L 1052 515]{left 5}-
{(}changed{)}{enter}[w 1][L 1897 13]
```

Here, both Insert-This commands and Microsoft character codes have been used (see previous sections). Insert-this commands are always enclosed in [] parentheses, while microsoft character codes are enclosed in {} parentheses. If you want to write a normal parenthesis somewhere, it must be written {(} because ordinary parentheses have a special meaning otherwise (see the section on Microsoft character codes).

I will now show which parts do what in the example above, by using different colors:

```
[L 157 14][L 204 55][L 617 237]{TAB 16}2021-01-01|2021-12-31[L
29 48][L 62 249][L 566 194][w 1][L 1052 515]{left 5}-
{(}changed{)}{enter}[w 1][L 1897 13]
```

This color: Uses the mouse to click View and then Edit Document (in Word).

Clicks in the file's first form field (probably unnecessary).

Navigates to the field to be filled in (by pressing tab) and fills in a date, moves to the next field and fills in another date.

Opens the file dialog to save in the current folder and waits an extra second for it.

Click in the file name field after the file name. Go five steps to the left with the cursor arrows to go past the .docx extension and type there -(changed) and then press enter to save. Then wait an extra second for this.

Closes the window with a click on the x.

Batch from list

This is a method that allows you to repeat a procedure based on a register (a list in a text file). You first select the text in the insert-this window to be executed for each line in the register file. Then press the "Batch from list" button and select the register file in question.

The register file must be a text file that contains tab-delimited data. (Which can, for example, be exported from Excel).

How data is entered from the register file is by using the [C column number] command in instert-this. (For example, [c 1] for the text in column 1).

Here is an explanatory example:

A register file looks like this:

Olof	*Petterson*	*Ekeby*
Ölme	*Ylleman*	*Grabohusgärde*
Erik	*Svennhög*	*Grumsbro*

Selected text in the Instert-This window looks like this:

```
First name: [c 1]
Surname: [c 2]
Location: [c 3]
---- {enter}
```

If you then "execute" the selected text in a text editor, you will get the following results:

First name: Olof
Surname: Petterson
Location: Ekeby

First name: Ölme
Surname: Ylleman
Location: Grabohusgärde

First name: Erik

Surname: Svennhög
Location: Grumsbro

The idea is that by also using mouse clicks in your procedure, for example, you can create a number of documents based on a template. The procedure in the Insert-This text could then load a template in eg word, fill in data from the list, save the file with one of the fields in the list used as part of the file name, eg social security number and then close the file. Then you get as many files as rows in the list file.

It could also be used to go into a database, such as Pro Capita, and change something for all the people in the list. Then the list would contain social security numbers and you would use mouse click programming to search for each person and make the change in the database.

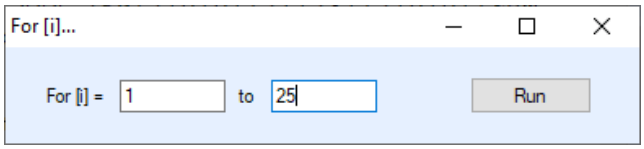
The execution can be interrupted the same way as the above function, by holding down both mouse buttons simultaneously until it is stopped.

Warning: The same caution must be applied as described under "Batch open and insert"!

FOR-loop

Both "Batch open and insert" and "Batch from list" in the previous sections are different ways of repeating a sequence several times. Here comes a third method which is more of a general loop.

First select the sequence in the text to be executed. Then press the button "For [i] to" and you will get a box that looks like this:



If you leave the first box at 1, then the second box is the number of times the loop will be run, in this case 25 times. For each revolution, the variable [i] is increased by 1 and this variable can be used both in text and in conditional control (see next section).

For example, if you type 20 in the first box and 24 in the second box and run the sequence "Hello [i]{enter}", the following will be sent to the keyboard:

Hello 20
Hello 21
Hello 22
Hello 23
Hello 24

It's possible to stop the execution the same way as the other loop functions, by holding down both mouse buttons simultaneously until it is stopped.

Conditional control

The conditional control commands look like this:

> [if expression] … Sequence executed if the expression is true…[else] … sequence executed if the expression is false… [endif]

Without else:

> [if expression] … Sequence executed if the expression is true…[endif]

The ”expression" is structured according to:

Parameter_A comparison_method parameter_B

Possible parameters are:

Clip	It is the text content on the clipboard that is compared.
Filename	It is the current file name that is compared. (Only when “Batch open and insert” is used)
C number	Data from column number number. (Only when “Batch from list” is used)
i	Current number if you are running a FOR [i] loop.
Another_word	Any text or number.
"More words"	If there are spaces in the text you want to compare, enclose the text with "-signs.

Note: If you want to remove the meaning of the special parameters, enclose them with the "-sign. For example, "filename" means the text filename and not the current filename.

Also note: Microsoft character codes cannot be used inside of parameters.

Comparison methods:

A = B	True if A is equal to B
A <> B	True if A is not equal to B
A contains B	True if A contains B, eg if B is "hello" and A is "Hello to you" then A contains B..
A > B	True if A is greater than B, eg if A is 5 and B is 2.
A < B	True if A is less than B
A >= B	True if A is greater than or equal to B
A =< B	True if A is equal to or less than B

There must be spaces between the parameters and the comparison method.

NOTE: Conditional statements cannot be nested. In other words, you cannot have a new if statement inside another if-else endif construct.

The command **[break comment]** is used to interrupt the execution and any ongoing loop or batch and is mainly used together with conditions, e.g.

```
[if clip <> 201231][break Incorrect date][endif]
```

Some examples:

This writes different texts depending on whether the clipboard contains the word "summer" or not:

```
[if clip contains summer] The word summer is in the text on
the clipboard [else] Summer is not on the clipboard [endif]
```

To put something on the clipboard, use the character code ^C (see character codes section). A sequence that first puts something on the clipboard and then compares it could look like this:

```
[L 617 237]^C[if clip = 2020-07-01] …Do something…[endif]
```

Comparison of numbers in column 2 of a data list (when using Batch from list)

```
[if c 2 > 200] Data in column 2 is greater than 200 [endif]
```

Different sequences depending on the file type when using Batch open and insert:

```
[if filename contains .docx] …sequence for word…[endif][if
filename contains .txt] …Sequence for Notepad…[endif][if
filename contains .xlsx]… Sequence for excel…[endif]
```

Different sequences depending on the value of variable i when using a FOR-loop:

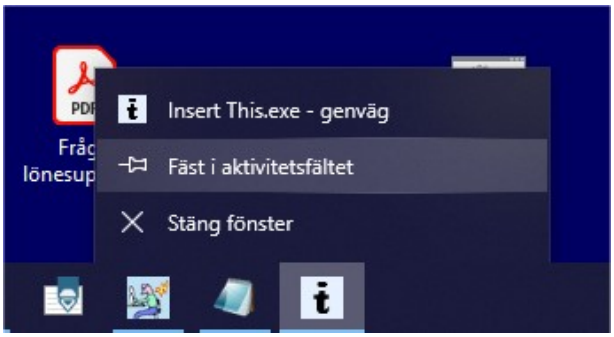
```
[if i > 99][break Too large number][endif][if i < 10]
…sequence for numbers below 10… [else] …Sequence for numbers
10 to 99…[endif]
```

The following advanced example is probably a little difficult to understand, but it's there to show the possibilities. Based on a list of people in Excel, it goes over to Pro Capita and ends an "activity" in a plan and starts a new one. Then it goes back to Excel and puts an x for the person in the list. The sequence is run by setting the cursor at the first social security number in Excel and then you start the sequence with the "For [i] to" button where you enter the first excel row number and the last row number. The clipboard is used both to get the social security number from Excel to Pro Capita and to check that you have ended up in the right "activity".

```
Extension in Pro Capita
^c[L 857 1061][L 892 973][L 191 88]^A^V{enter}[w 1][L 668
434][L 928 670][L 186 203][w 2][L 347 133][w 4][L 438 288]|[w
.3]^c[w .3][if clip <> 201231][break Wrong date or
activity][endif][L 974 288]201231[L 976 309][L 872 356][L 43
55][w 1][L 16 58][w 1][L 609 165][L 503 199][L 907 165][L 862
199][L 885 197][L 838 224][L 444 289]210101|211231|210101[rem
place and manager][L 514 195][L 493 339][L 516 223][L 616
569][L 499 498][rem spara][L 40 57][w 5][rem put a mark][L 469
1064]{tab 7}x{enter}
```

Suggested installation

1. Download the program from http://www.boray.se/software/insertthis/
2. Save it to your desktop.
3. Start the program by double clicking it's icon on the desktop.
4. Go down to the taskbar and right-click the Insert-this icon. Select "Pin to taskbar"



5. Ready