# Summary

“Cate” is a C-like programming language and also the name of its compiler.

For assembling the generated assembly code, use Asm8 (<https://github.com/inufuto/asm8>).

## Typesetting Conventions In This Manual

To make the reading of this document easier, the following conventions are used.

* Example text is written in monospaced.
* <angle brackets and the text within> will be replaced with appropriate text.
* Text in *italics* are keywords.

## Target CPUs

| CPU | Filename |
| --- | --- |
| Z80 | cate80.exe |
| 6800 | cate68.exe |
| 6809 | cate09.exe |
| 6502 | cate65.exe |
| TMS9900 | Cate99.exe |
| 8080 | Cate80i.exe |
| 8086 | Cate86.exe |

## Changes From C

* Data types are different (smallest is 8-bit).
* Only preprocessor directive is #include.
* No recursive functions.
* No variable arguments/parameters.
* The syntax of pointers and arrays is different.
* The syntax of structs, outside of definition, is different.
* Multiplication and division utilize literals only.

# How to Run

At the command prompt, use the following syntax.

| **CPU** | **Command** |
| --- | --- |
| Z80 | cate80 <source file> |
| 6800 | cate68 <source file> |
| 6809 | cate09 <source file> |
| 6502 | cate65 <source file> |
| TMS9900 | cate99 <source file> |
| 8080 | cate80i <source file> |
| μCOM87 | cate87 <cpu options> <source file> |
| 8086 | cate86 <source file> |

### CPU Options

| -7801 (or omit) | μPD7800～μPD7802 instruction set |
| --- | --- |
| -7805 | μPD78C05, μPD78C06 instruction set |

# Data Types

## Primitives

| **Keyword** | **Data type** |
| --- | --- |
| *byte* | unsigned 8-bit |
| *sbyte* | signed 8-bit |
| *word* | unsigned 16-bit |
| *sword* | signed 16-bit |
| *bool* | boolean |

Booleans can be assigned with the keywords *true* and *false*.

## Pointers

Done differently than C, and expressed such:

ptr<type>

For null pointers, *nullptr* instead of 0.

## Structs

Structs use the *struct* keyword in the type definition, but outside of that, only the type name should be written.

# Peculiar Syntax

## Naming Constants

Constants can be defined using the *constexpr* keyword.

constexpr　<identifier> = <constant expression>

## *for* Statement

Range-based *for* loops can be used. They are limited to using a pointer and an array.

for (<pointer> : <array>)

## *repeat* Statement

The *repeat* keyword can be used to write a loop of fixed count.

repeat (<expression>)

# Assembly Generation

## Name

An underscore is appended to the end of function and variable names that are output into assembly.

# Arguments and Return Values

The arguments/parameters and return values are delivered through registers and fixed address memory. Because only one stack frame is allocated per function, recursive calls are not possible.

| CPU | Size | Return value | Argument  / parameter 1 | Argument  / parameter 2 | Argument  / parameter 3 | Argument  / parameter 4 |
| --- | --- | --- | --- | --- | --- | --- |
| Z80 | 8bit | A | A | E | C | memory,  <function name>\_@Param3 |
| 16bit | HL | HL,  IX for struct pointers | DE, IY for struct pointers | BC | memory, <function name>\_@Param3 |
| 6800 | 8bit | A | A | memory, <function name>\_@Param1 | memory,  <function name>\_@Param2 | memory, <function name>\_@Param3 |
| 16bit | X | memory, <function name>\_@Param0 | memory, <function name>\_@Param1 | memory, <function name>\_@Param2 | memory, <function name>\_@Param3 |
| 6809 | 8bit | A | A | B | memory, <function name>\_@Param2 | memory, <function name>\_@Param3 |
| 16bit | D | X | Y | memory, <function name>\_@Param2 | memory, <function name>\_@Param3 |
| 6502 | 8bit | Y | memory, <function name>\_@Param0 | memory, <function name>\_@Param1 | memory, <function name>\_@Param2 | memory, <function name>\_@Param3 |
| 16bit | low byte in Y,  high byte in X | memory, <function name>\_@Param0 | memory, <function name>\_@Param1 | memory, <function name>\_@Param2 | memory, <function name>\_@Param3 |
| TMS9900 | 8bit | high byte of R0 | high byte of R1 | high byte of R2 | high byte of R3 | high byte of R4 |
| 16bit | R0 | R1 | R2 | R3 | R4 |
| 8080 | 8bit | A | A | E | C | memory, <function name>\_@Param3 |
| 16bit | HL | HL | DE | BC | memory, <function name>\_@Param3 |
| μCOM87 | 8bit | A | A | E | C | memory, <function name>\_@Param3 |
| 16bit | HL | HL | DE | BC | memory, <function name>\_@Param3 |
| 8086 | 8bit | AL | AL | DL | CL | memory, <function name>\_@Param3 |
| 16bit | AX | AX | DX | BX | memory, <function name>\_@Param3 |

## Runtime Libraries

It is necessary to link to one of the libraries below.

| CPU | Filename |
| --- | --- |
| Z80 | cate80.lib |
| 6800 | cate68.lib |
| 6809 | cate09.lib |
| 6502 | cate65.lib |
| TMS9900 | Cate99.lib |
| 8080 | Cate80i.lib |
| μPD7800~7801 | Cate87.lib |
| μPD78C05~78C06 | Cate87c.lib |
| 8086 | Cate86.lib |

## Authorship

Translated from the Feb 24, 2023 version of Inufuto’s (<https://github.com/inufuto/>) original manual.

Translation and edits done by Christen Gottschlich ([christen.got@gmail.com](mailto:christen.got@gmail.com)) and finished on August 10, 2023. Edits were few and added for clarity.