University Center of Mila
2nd year bachelor's degree in computer science
Course : Operating Systems 1

Tutorial Series No. 2

Exercise 1:

Q1) Indicate for each of the following tools what the input and output information are: **Text editor**, **Assembler**, **Compiler**, **Interpreter**, **Linker**, **Loader**
Q2) Do interpreters, compilers, and loaders belong to the operating system?

Exercise 2: (C.Carrez Exercices de Systèmes)

To study the functioning of the **assembler**, **linker**, and **loader**, consider a machine using the following instructions:

```
LOAD <adress> (Transfer the content of address into the accumulator register)
ADD <constant> (Add the constant to the accumulator register)
MUL <constant> (Multiply the accumulator register by the constant)
STORE <adress> (Transfer the content of the accumulator register to address)
JMP <adress> (Jump to the instruction at address)
```

```
The machine code for these instructions is as follows:
```

```
  LOAD  10
  ADD   20
  MUL   30
  STORE 40
  JMP   50
```

1) Study of the Assembler

Consider the following small program:

```
     Name Module1 /*program name*/
  A: WORD = 25 /*memory location initialized to 25*/
  B: WORD = 8 /*memory location initialized to 8*/
     LOAD A
     MUL 10
     ADD 1
     STORE B
     END
```

The result of assembling a program is called an **object** module, a file containing: the program name and size, and for each instruction, the relative address and machine code in absolute or relative addressing.
Q1) Complete the following table representing the records of the object module for the program Module1.

Type	**Name or Relative Address**	**Size or Code or Value**	**Source Code Text**
Header	`Module1`		`NOM Module1`
Absolute Code			`A: WORD = 25`
Absolute Code			`B: WORD = 8`
Relocatable Code			`LOAD A`
Absolute Code			`MUL 10`
Absolute Code			`ADD 1`
Relocatable Code			`STORE B`
End Module			`END`

2) Study of the Linker

The program uses another module (**Module2**) by linking to the symbolic address **SUITE** of this module. **Module1** can also be used by another module at the symbolic address **CALCUL**. **SUITE** is called an **unresolved link**, which will be resolved by the linker, while **CALCUL** is a **usable link**. The updated program becomes:

```
          Name Module1 /*program name*/
          PUBLIC CALCUL /*entry point in Module1*/
          EXTERN SUITE /*entry point in another module (Module2) */
       A: WORD = 25 /*memory location initialized to 25*/
       B: WORD = 8 /*memory location initialized to 8*/
  CALCUL: LOAD A
          MUL 10
          ADD 1
          STORE B
          JMP SUITE
          END

          Name Module2 /*program name*/
          PUBLIC SUITE /*entry point in Module2*/
          EXTERN SUITE2 /*entry point in another module*/
     C : WORD = 30 /*memory location initialized to 30*/
   SUITE: LOAD C
          MUL 20
          STORE C
          JMP SUITE2
          END
```

Q2) Modify the object module resulting from assembling Module1 by completing the following table:

Type	**Name or Relative Address**	**Size or Code or Value**	**Source Code Text**
Header	Module1		`NOM Module1`
Usable Link	CALCUL		`PUBLIC CALCUL`
Unresolved Link	SUITE		`EXTERN SUITE`
Absolute Code			`A: WORD = 25`
Absolute Code			`B: WORD = 8`
Relocatable Code			`CALCUL: LOAD A`
Absolute Code			`MUL 10`
Absolute Code			`ADD 1`
Relocatable Code			`STORE B`
Code with Unresolved Link			`JMP SUITE`
End Module			`END`

Q3) Provide the object module resulting from assembling the program Module2.

Q4) During the linking process, Module1 is placed at relative address 200 in the executable program. Module2 is placed immediately after Module1 in the executable. Provide the records given by the linker for both modules by completing the following table. Assume SUITE2 is a usable link located at relative address 350 in the executable program.

Type	**Relative Address**	**Code or Value**	**Source Code Text**

3) Study of the Loader

Q5) The loader places the program starting at memory address 7000. Provide the memory content of the locations containing `Module1` and `Module2`.