

[CLAIMS]

[Claim 1]

A program for causing a computer to perform:

5 a first function of identifying the type of a process of a target program which is another program corresponding to an object to be operated by the program depending on a platform in which the target program operates;

a second function of determining whether the type of the process of the target program identified by the first function matches a first type of a process of the program; and

10 a third function of starting a second type of a process of the program matching the type of the process of the target program and ending the first type of the process of the program when it is determined that the type of the process of the target program does not match the first type of the process of the program according to the second function.

[Claim 2]

15 A program for causing a computer to perform:

a first function of identifying the type of a process of a target program which is another program corresponding to an object to be operated by the program depending on a platform in which the target program operates;

20 a second function of determining whether the type of the process of the target program identified by the first function matches a first type of a process of the program; and

a third function of generating a program file of the program matching the type of the process of the target program and starting a process of a second type according to the generated program file when it is determined that the type of the process of the target program does not match the first type of the process of the program according to the

25

second function.

[Claim 3]

The program according to claim 2, wherein the third function starts a process of the second type of the program and ends the process of the first type of the program.

5 [Claim 4]

The program according to any one of claims 1 to 3, wherein the program has resources of programs according to the process of the first type and the process of the second type.

10 [Claim 5]

A system comprising:

at least one memory that stores a software component including first, second, third, fourth, fifth, seventh and eighth program components;

15 at least one processor configured to read the software component out of the at least one memory and to execute the software component to cause the at least one processor to perform:

executing the first program component to generate a ninth program component;

executing the second program component to generate a tenth program component;

20 executing the third program component to cause the ninth program component to operate a target program component in a first type of a bit process in bit number;

executing the fourth program component to cause the tenth program component to operate the target program component in a second type of the bit process in bit number, wherein the second type of the bit process in bit number is different in the bit process in

25 bit number which is executed on a first platform from the first type of the bit process in

bit number;

executing the fifth program component to identify the type of the bit process in bit number of the target program component depending on the first platform in which the target program component is executed;

5 executing the sixth program component to determine whether the type of the bit process in bit number of the target program component identified matches the type of the bit process in bit number of an executed one of the ninth program component and the tenth program component, wherein the executed one is being executed;

executing the seventh program component to make the ninth program component
10 nonexecuted;

executing the eighth program component to make the tenth program component nonexecuted;and

if the sixth program component determined that the type of the bit process in bit number of the target program component does not match the type of the bit process in bit
15 number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program component matches the type of the bit process in bit number of the nonexecuted one of the ninth program component and the tenth program component, wherein the nonexecuted one is being nonexecuted, and wherein a first one of the third program
20 component and the fourth program component executes the executed one, and wherein a second one of the third program component and the fourth program component executes the nonexecuted one, wherein a third one of the seventh program component and the eighth program component makes the executed one nonexecuted, wherein a fourth one of the seventh program component and the eighth program component makes the
25 nonexecuted one nonexecuted,

executing the second one of the third program component and the fourth program component to execute the nonexecuted one; and

executing the third one of the seventh program component and the eighth program component to make the executed one nonexecuted.

5

[Claim 6]

A system comprising:

at least one memory that stores a software component including first, second, third, fourth, fifth, seventh and eighth program components;

10 at least one processor configured to read the software component out of the at least one memory and to execute the software component to cause the at least one processor to perform:

executing the first program component to generate a ninth program component;

executing the second program component to generate a tenth program

15 component;

executing the third program component to cause the ninth program component to operate a target program component in a first type of a bit process in bit number;

executing the fourth program component to cause the tenth program component to operate the target program component in a second type of the bit process in bit number, wherein the second type of the bit process in bit number is different in the bit process in bit number which is executed on a first platform from the first type of the bit process in bit number;

20 executing the fifth program component to identify the type of the bit process in bit number of the target program component depending on the first platform in which the target program component is executed;

25

executing the sixth program component to determine whether the type of the bit process in bit number of the target program component identified matches the type of the bit process in bit number of an executed one of the ninth program component and the tenth program component, wherein the executed one is being executed;

5 executing the seventh program component to make the ninth program component nonexecuted;

 executing the eighth program component to make the tenth program component nonexecuted;and

 if the sixth program component determined that the type of the bit process in bit
10 number of the target program component does not match the type of the bit process in bit number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program component matches the type of the bit process in bit number of the nonexecuted one of the ninth program component and the tenth program component, wherein the
15 nonexecuted one is being nonexecuted, and wherein a first one of the first program component and the second program component generates the executed one, and wherein a second one of the first program component and the second program component generates the nonexecuted one, and wherein a third one of the third program component and the fourth program component executes the executed one, and wherein a fourth one
20 of the third program component and the fourth program component executes the nonexecuted one,

 executing the second one of the first program component and the second program component to generate the nonexecuted one; and

 executing the fourth one of the third program component and the fourth
25 program component to execute the nonexecuted one.

[Claim 7]

The system according to claim 6,

if a fifth one of the seventh program component and the eighth program

component makes the executed one nonexecuted, wherein a sixth one of the seventh

5 program component and the eighth program component makes the nonexecuted one
nonexecuted,

further executing the fifth one of the seventh program component and the

eighth program component to make the executed one nonexecuted.

10 [Claim 8]

A computer-implemented method by a system comprising: at least one memory

that stores a software component including first, second, third, fourth, fifth, seventh and
eighth program components; and at least one processor, the method comprising:

reading, by the at least one processor, the software component out of the at least

15 one memory;

executing, by the at least one processor, the first program component to generate
a ninth program component;

executing, by the at least one processor, the second program component to
generate a tenth program component;

20 executing, by the at least one processor, the third program component to cause the
ninth program component to operate a target program component in a first type of a bit
process in bit number;

executing, by the at least one processor, the fourth program component to cause
the tenth program component to operate the target program component in a second type

25 of the bit process in bit number, wherein the second type of the bit process in bit number

is different in the bit process in bit number which is executed on a first platform from the first type of the bit process in bit number;

executing, by the at least one processor, the fifth program component to identify the type of the bit process in bit number of the target program component depending on
5 the first platform in which the target program component is executed;

executing, by the at least one processor, the sixth program component to determine whether the type of the bit process in bit number of the target program component identified matches the type of the bit process in bit number of an executed one of the ninth program component and the tenth program component, wherein the
10 executed one is being executed;

executing, by the at least one processor, the seventh program component to make the ninth program component nonexecuted;

executing, by the at least one processor, the eighth program component to make the tenth program component nonexecuted; and

15 if the sixth program component determined that the type of the bit process in bit number of the target program component does not match the type of the bit process in bit number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program component matches the type of the bit process in bit number of the nonexecuted one of
20 the ninth program component and the tenth program component, wherein the nonexecuted one is being nonexecuted, and wherein a first one of the third program component and the fourth program component executes the executed one, and wherein a second one of the third program component and the fourth program component executes the nonexecuted one, wherein a third one of the seventh program component and the
25 eighth program component makes the executed one nonexecuted, wherein a fourth one of

the seventh program component and the eighth program component makes the nonexecuted one nonexecuted,

executing, by the at least one processor, the second one of the third program component and the fourth program component to execute the nonexecuted one; and

executing, by the at least one processor, the third one of the seventh program component and the eighth program component to make the executed one nonexecuted.

10 [Claim 9]

The computer-implemented method according to claim 8, wherein reading, by the at least one processor, the software component out of the at least one memory comprises reading the first, second, third, fourth, fifth, seventh and eighth program components.

15

[Claim 10]

A computer-implemented method by a system comprising: at least one memory that stores a software component including first, second, third, fourth, fifth, seventh and eighth program components; and at least one processor, the method comprising:

20 reading, by the at least one processor, the software component out of the at least one memory;

executing, by the at least one processor, the first program component to generate a ninth program component;

25 executing, by the at least one processor, the second program component to generate a tenth program component;

executing, by the at least one processor, the third program component to cause the ninth program component to operate a target program component in a first type of a bit process in bit number;

executing, by the at least one processor, the fourth program component to cause
 5 the tenth program component to operate the target program component in a second type of the bit process in bit number, wherein the second type of the bit process in bit number is different in the bit process in bit number which is executed on a first platform from the first type of the bit process in bit number;

executing, by the at least one processor, the fifth program component to identify
 10 the type of the bit process in bit number of the target program component depending on the first platform in which the target program component is executed;

executing, by the at least one processor, the sixth program component to determine whether the type of the bit process in bit number of the target program component identified matches the type of the bit process in bit number of an executed
 15 one of the ninth program component and the tenth program component, wherein the executed one is being executed;

executing, by the at least one processor, the seventh program component to make the ninth program component nonexecuted;

executing, by the at least one processor, the eighth program component to make
 20 the tenth program component nonexecuted; and

if the sixth program component determined that the type of the bit process in bit number of the target program component does not match the type of the bit process in bit number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program
 25 component matches the type of the bit process in bit number of the nonexecuted one of

the ninth program component and the tenth program component, wherein the nonexecuted one is being nonexecuted, and wherein a first one of the first program component and the second program component generates the executed one, and wherein a second one of the first program component and the second program component
5 generates the nonexecuted one, and wherein a third one of the third program component and the fourth program component executes the executed one, and wherein a fourth one of the third program component and the fourth program component executes the nonexecuted one,

executing, by the at least one processor, the second one of the first
10 program component and the second program component to generate the nonexecuted one; and

executing, by the at least one processor, the fourth one of the third program component and the fourth program component to execute the nonexecuted one.

15

[Claim 11]

The computer-implemented method according to claim 10,
wherein reading, by the at least one processor, the software component out of the at least one memory comprises reading the first, second, third, fourth, fifth, seventh and
20 eighth program components.

[Claim 12]

The computer-implemented method according to claim 10,
if a fifth one of the seventh program component and the eighth program
25 component makes the executed one nonexecuted, wherein a sixth one of the seventh

program component and the eighth program component makes the nonexecuted one nonexecuted,

further executing, by the at least one processor, the fifth one of the seventh program component and the eighth program component to make the executed one

5 nonexecuted.

[Claim 13]

A non-transitory computer-readable medium that stores a computer-executable software component including first, second, third, fourth, fifth, seventh and eighth program

10 components, when executed by one or more proceeding devices, to cause the one or more proceeding devices to perform at least:

executing the first program component to generate a ninth program component;

executing the second program component to generate a tenth program component;

15 executing the third program component to cause the ninth program component to operate a target program component in a first type of a bit process in bit number;

executing the fourth program component to cause the tenth program component to operate the target program component in a second type of the bit process in bit number, wherein the second type of the bit process in bit number is different in the bit process in bit number which is executed on a first platform from the first type of the bit process in
20 bit number;

executing the fifth program component to identify the type of the bit process in bit number of the target program component depending on the first platform in which the target program component is executed;

25 executing the sixth program component to determine whether the type of the bit

process in bit number of the target program component identified matches the type of the bit process in bit number of an executed one of the ninth program component and the tenth program component, wherein the executed one is being executed;

executing the seventh program component to make the ninth program component
5 nonexecuted;

executing the eighth program component to make the tenth program component nonexecuted; and

if the sixth program component determined that the type of the bit process in bit number of the target program component does not match the type of the bit process in bit
10 number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program component matches the type of the bit process in bit number of the nonexecuted one of the ninth program component and the tenth program component, wherein the nonexecuted one is being nonexecuted, and wherein a first one of the third program
15 component and the fourth program component executes the executed one, and wherein a second one of the third program component and the fourth program component executes the nonexecuted one, wherein a third one of the seventh program component and the eighth program component makes the executed one nonexecuted, wherein a fourth one of the seventh program component and the eighth program component makes the
20 nonexecuted one nonexecuted,

executing the second one of the third program component and the fourth program component to execute the nonexecuted one; and

executing the third one of the seventh program component and the eighth program component to make the executed one nonexecuted.

[Claim 14]

A non-transitory computer-readable medium that stores a computer-executable software component including first, second, third, fourth, fifth, seventh and eighth program components, when executed by one or more proceeding devices, to cause the one or more
 5 proceeding devices to perform at least:

executing the first program component to generate a ninth program component;

executing the second program component to generate a tenth program
 component;

executing the third program component to cause the ninth program component to
 10 operate a target program component in a first type of the bit process in bit number;

executing the fourth program component to cause the tenth program component
 to operate the target program component in a second type of the bit process in bit number,
 wherein the second type of the bit process in bit number is different in the bit process in
 bit number which is executed on a first platform from the first type of the bit process in
 15 bit number;

executing the fifth program component to identify the type of the bit process in
 bit number of the target program component depending on the first platform in which the
 target program component is executed;

executing the sixth program component to determine whether the type of the bit
 20 process in bit number of the target program component identified matches the type of the
 bit process in bit number of an executed one of the ninth program component and the
 tenth program component, wherein the executed one is being executed;

executing the seventh program component to make the ninth program component
 nonexecuted;

25 executing the eighth program component to make the tenth program component

nonexecuted;and

if the sixth program component determined that the type of the bit process in bit number of the target program component does not match the type of the bit process in bit number of the executed one of the ninth program component and the tenth program component and that the type of the bit process in bit number of the target program component matches the type of the bit process in bit number of the nonexecuted one of the ninth program component and the tenth program component, wherein the nonexecuted one is being nonexecuted, and wherein a first one of the first program component and the second program component generates the executed one, and wherein a second one of the first program component and the second program component generates the nonexecuted one, and wherein a third one of the third program component and the fourth program component executes the executed one, and wherein a fourth one of the third program component and the fourth program component executes the nonexecuted one,

executing the second one of the first program component and the second program component to generate the nonexecuted one; and
 executing the fourth one of the third program component and the fourth program component to execute the nonexecuted one.

[Claim 15]

The non-transitory computer-readable medium according to claim 14,

if a fifth one of the seventh program component and the eighth program component makes the executed one nonexecuted, wherein a sixth one of the seventh program component and the eighth program component makes the nonexecuted one nonexecuted,

further executing the fifth one of the seventh program component and the eighth program component to make the executed one nonexecuted.