

Slides

Development > Programming Languages > C++

The C++ 20 Masterclass : From Fundamentals to Advanced

Learn and Master Modern C++ From Beginning to Advanced in Plain English : C++11, C++14, C++17, C++20 and More!

4.7 ★★★★★

Created by [Daniel Gakwaya](#)

Section 3 : Diving In

C++ Project template

```
#include <iostream>

consteval int get_value(){
    return 3;
}

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```

Slide intentionally left empty

Your First C++ Program





```
#include <iostream>

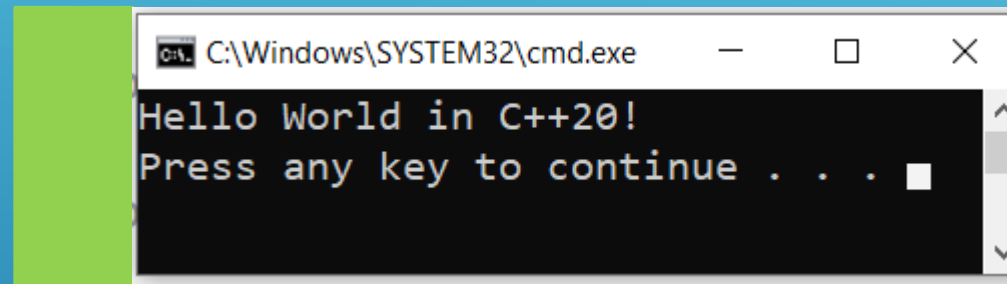
constexpr int get_value(){
    return 3;
}

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```

Build output : Your Program

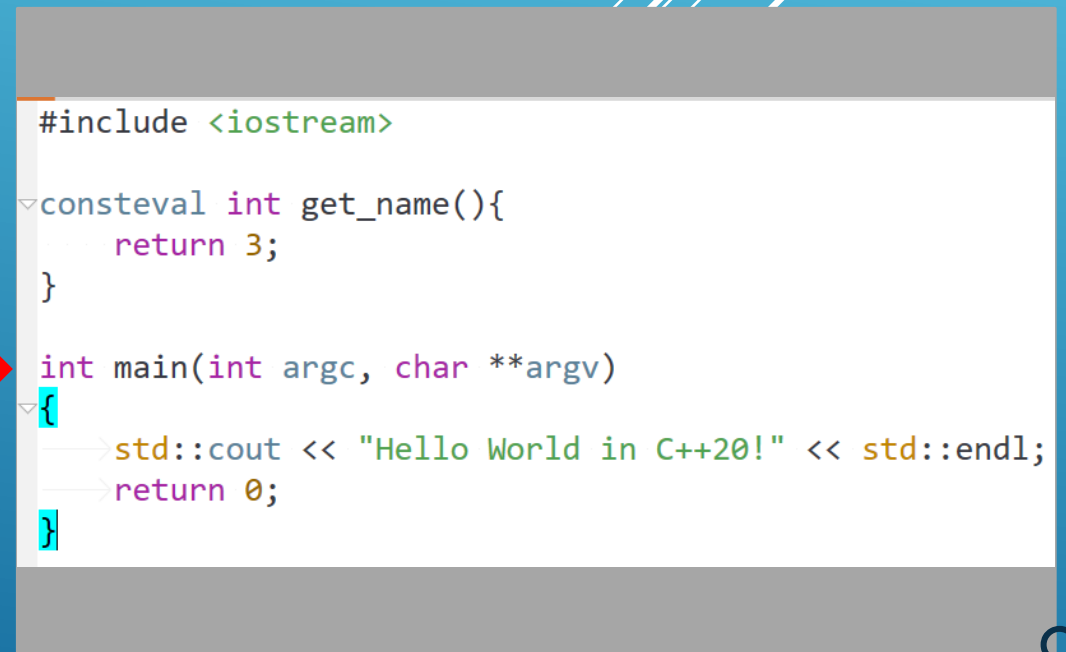
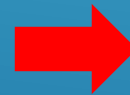
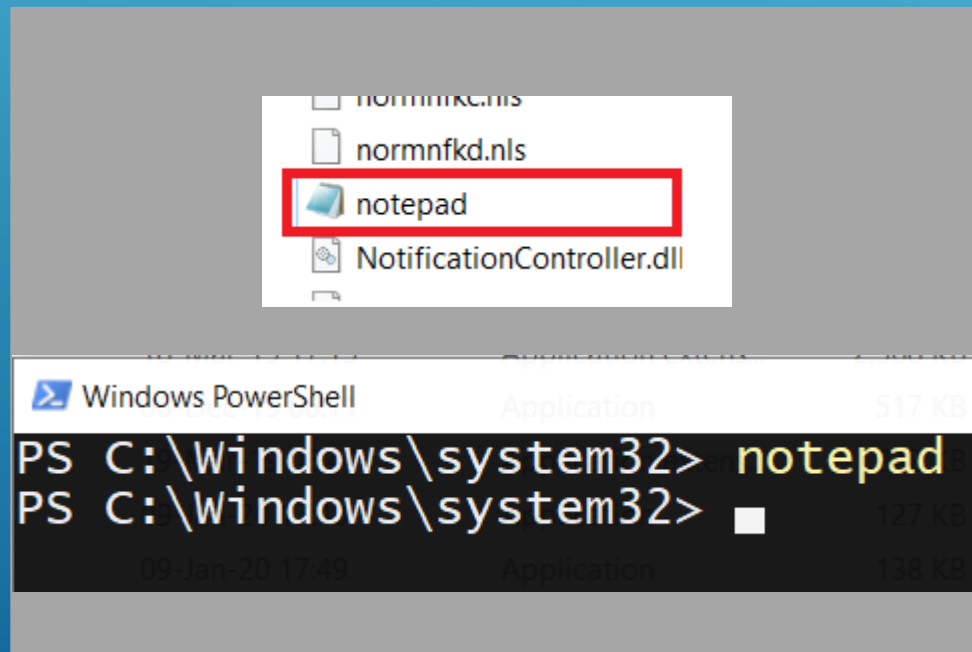
Local Disk (D:) > OnlineCourses > 9.CppMasterClass > DemoCodeV2 > 3.FirstSteps > 3-1FirstCppProgram > Debug

Name	Date modified	Type	Size
 .d	17-Aug-20 22:34	D File	1 KB
 3-1FirstCppProgram	17-Aug-20 22:34	Application	63 KB
 main.cpp.o	17-Aug-20 22:34	O File	16 KB
 main.cpp.o.d	17-Aug-20 22:34	D File	1 KB



```
C:\Windows\SYSTEM32\cmd.exe
Hello World in C++20!
Press any key to continue . . .
```


Entry Point



Slide intentionally left empty

Comments

```
//Entry point main function
int main(int argc, char **argv)
{
    //One line comment

    /*
    Multi-line block comment
    Another line
    Oh! And another one !
    */

    //Print out some text
    → std::cout << "Hello World in C++20!" << std::endl;
    → return 0;
}
```

// Comments out a single line

/* ... */ Block Comments out a block of text

/* ... */ Block comments can't be nested

Use comments to document your code. Don't overdo it though.

Slide intentionally left empty

Errors and Warnings

15



Compile Time Errors

Runtime Errors

Warnings


```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```



Compiler



Executable binary file

```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```

Compiler



```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```

```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl
    return 0;
}
```

Compiler



```
mingw32-make[1]: Entering directory 'D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors'
C:/mingw32/bin/g++.exe -c "D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors/main.cpp" -g -O0 -Wall
D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors/main.cpp: In function 'int main(int, char**)':
D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors/main.cpp:13:51: error: expected ';' before 'return'
13 | std::cout << "Hello World in C++20!" << std::endl
```

Executable binary file

Runtime



C:\Windows\SYSTEM32\cmd.exe

```
Hello World in C++20!  
Press any key to continue . . .
```

Executable binary file

Runtime



C:\Windows\SYSTEM32\cmd.exe

```
Hello World in C++20!  
Press any key to continue . . .
```

Crash

21

```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "Hello World in C++20!" << std::endl;
    return 0;
}
```



Compiler

```
D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors/main.cpp:24:7: warning: division by zero [-Wdiv-by-zero]
```

```
24 |     20/0;  
   |     ~~~
```

```
D:/OnlineCourses/9.CppMasterClass/DemoCodeV2/3.FirstSteps/3-3Errors/main.cpp:24:7: warning: statement has no effect [-Wunused-value]
```

22

Slide intentionally left empty

Statements and Functions

- A statement is a basic unit of computation in a C++ program
- Every C++ program is a collection of statements organized in a certain way to achieve some goal
- Statements end with a semicolon in C++ (;)

```
int main(int argc, char **argv)
{
    int firstNumber = 12;
    int secondNumber = 9;

    int sum = firstNumber + secondNumber;

    std::cout << "The sum of the two numbers is : " << sum << std::endl;

    return 0;
}
```

- Statements are executed in order from top to bottom when the program is run
- Execution keeps going until there is a statement causing the program to terminate, or run another sequence of statements

```
int firstNumber = 12;  
int secondNumber = 9;  
  
int sum = firstNumber + secondNumber;
```

first_number



second_number



sum

```
int addNumbers(int first_number, int second_number){  
    int sum = first_number + second_number;  
    return sum;  
}
```

return type

```
int addNumbers(int first_number, int second_number){  
    int sum = first_number + second_number;  
    return sum;  
}
```

function name

```
int addNumbers(int first_number, int second_number){  
    int sum = first_number + second_number;  
    return sum;  
}
```


parameters

```
int addNumbers(int first_number, int second_number){  
    int sum = first_number + second_number;  
    return sum;  
}
```



A function must be defined before it's use

```

int addNumbers(int first_number, int second_number){
    int sum = first_number + second_number;
    return sum;
}

int main(int argc, char **argv)
{
    // ...

    int firstNumber = 12;
    int secondNumber = 9;

    int sum = firstNumber + secondNumber;

    // ...

    sum = addNumbers(firstNumber, secondNumber);

    sum = addNumbers(34, 7);

    std::cout << "The sum of the two numbers is : " << sum << std::endl;
    std::cout << "The sum of the two numbers is : " << addNumbers(23, 8) << std::endl;
    return 0;
}

```

Slide intentionally left empty

Input Output

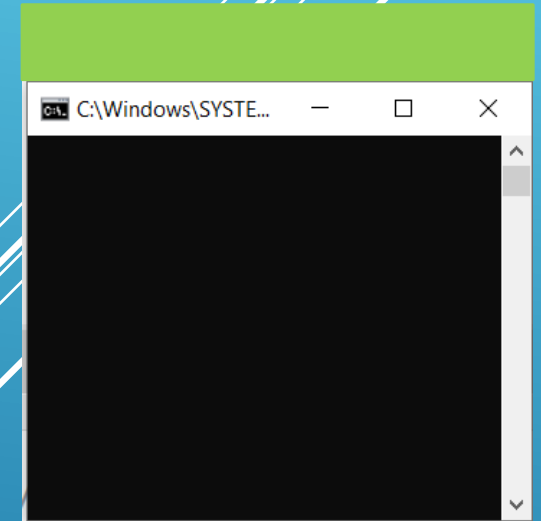
```
int main(int argc, char **argv)
{
    //Compiler syntax error : missing semicolon
    std::cout << "Hello World in C++20!" << std::endl;

    int a {4};
    int b {4};

    //Runtime error
    int c = 10/ (a -b);
    std::cout << "The value of c is : " << c << std::endl;

    //Warnings
    20/0; // This throws a warning on gcc10.
    return 0;
}
```

std::cout



stream	Purpose
std::cout	Printing data to the console (terminal)
std::cin	Reading data from the terminal
std::cerr	Printing errors to the console
std::clog	Printing log messages to the console

Printing data

```
//std::cout : Printing stuff to the console
std::cout << "Hello World!" << std::endl;

std::cout << "The number is : " << 12 << std::endl;

int age {21};
std::cout << "The age is : " << age << std::endl;

//Error
std::cerr << "std::cerr output : Something went wrong" << std::endl;

//Log message
std::clog << " std::clog output : This is a log message" << std::endl;
```


Reading data in

```
int age;
std::string name;

std::cout << "Please type in your Last Name : " << std::endl;
std::cin >> name;

std::cout << "Please type in your age : " << std::endl;
std::cin >> age;

std::cout << "Hello " << name << "! You are " << age << " years old" << std::endl;
```

Chaining std::cin

```
int age;  
std::string name;  
  
std::cout << "Please type in your Last name and age, separated by spaces : " << std::endl;  
  
std::cin >> name >> age ;//Input name and age  
  
std::cout << "Hello " << name << "! You are " << age << " years old." << std::endl;
```

Reading data with spaces

```
int age;
std::string full_name;

std::cout << "Please type in your full name : " << std::endl;
std::getline(std::cin, full_name);

std::cout << "Type in your age : " << std::endl;
std::cin >> age;
std::cout << "Hello " << full_name << "! You are " << age << " years old." << std::endl;
```

Slide intentionally left empty

C++ Program Execution Model & Memory Model

45

```

#include <iostream>

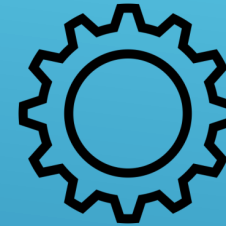
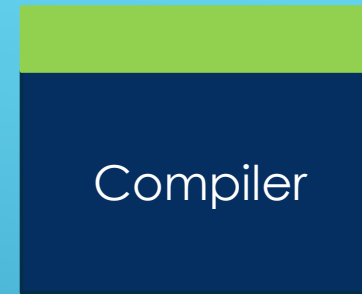
int add_numbers(int a, int b)
{
    return a + b;
}

int main()
{
    int a = 10;
    int b = 5;
    int c;

    std::cout << "Statement1" << std::endl;
    std::cout << "Statement2" << std::endl;
    c = add_numbers(a, b);
    std::cout << "Statement3" << std::endl;
    std::cout << "Statement4" << std::endl;

    return 0;
}

```



```

a = 10      (int)
b = 5       (int)
c           (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end

```

Program
area

0001

0002

0003

0004

0005

0006

0007

0008

0009

0010

...

...

0020

...

...

0030

...

CPU

0001

0002

0003

0004

0005

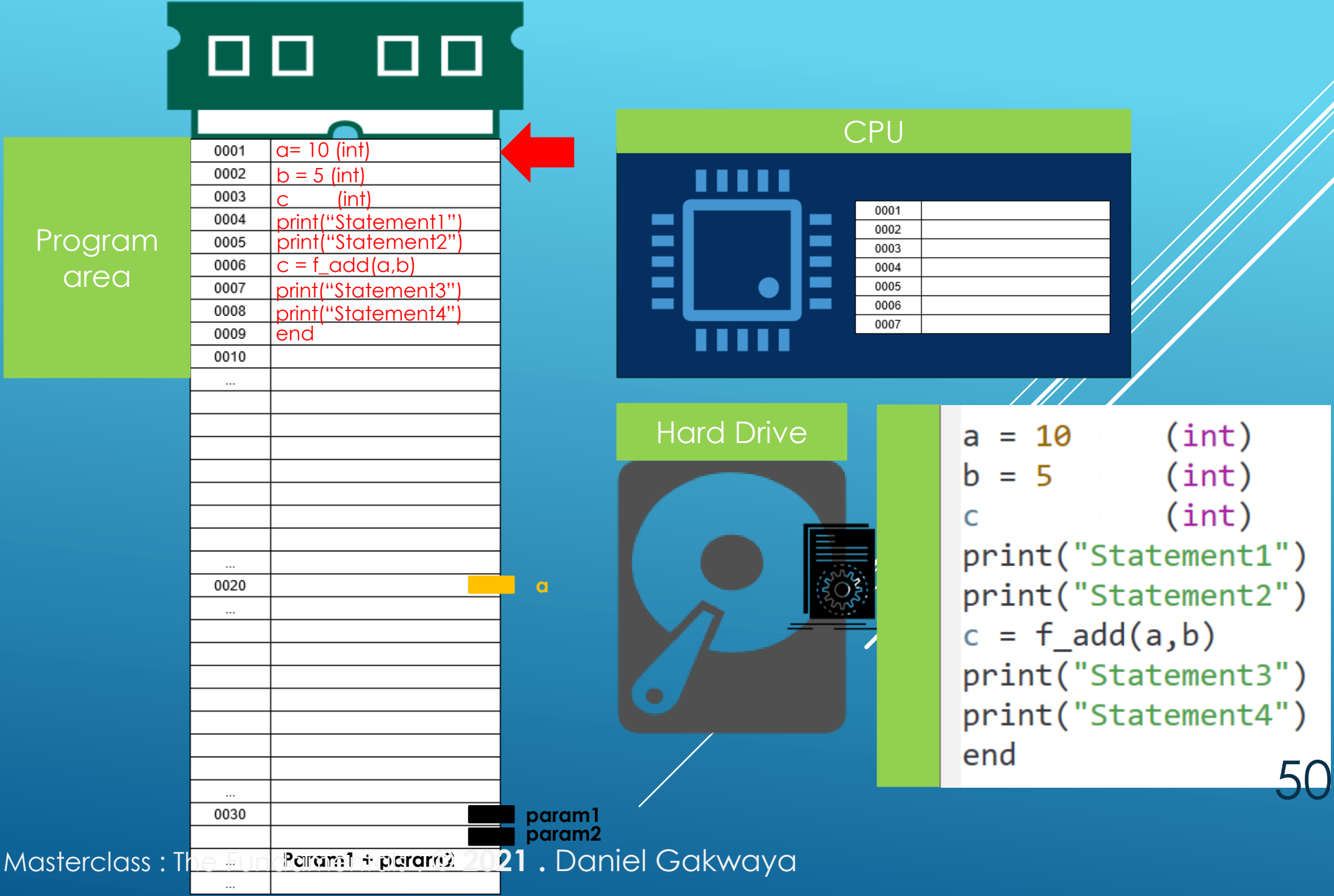
0006

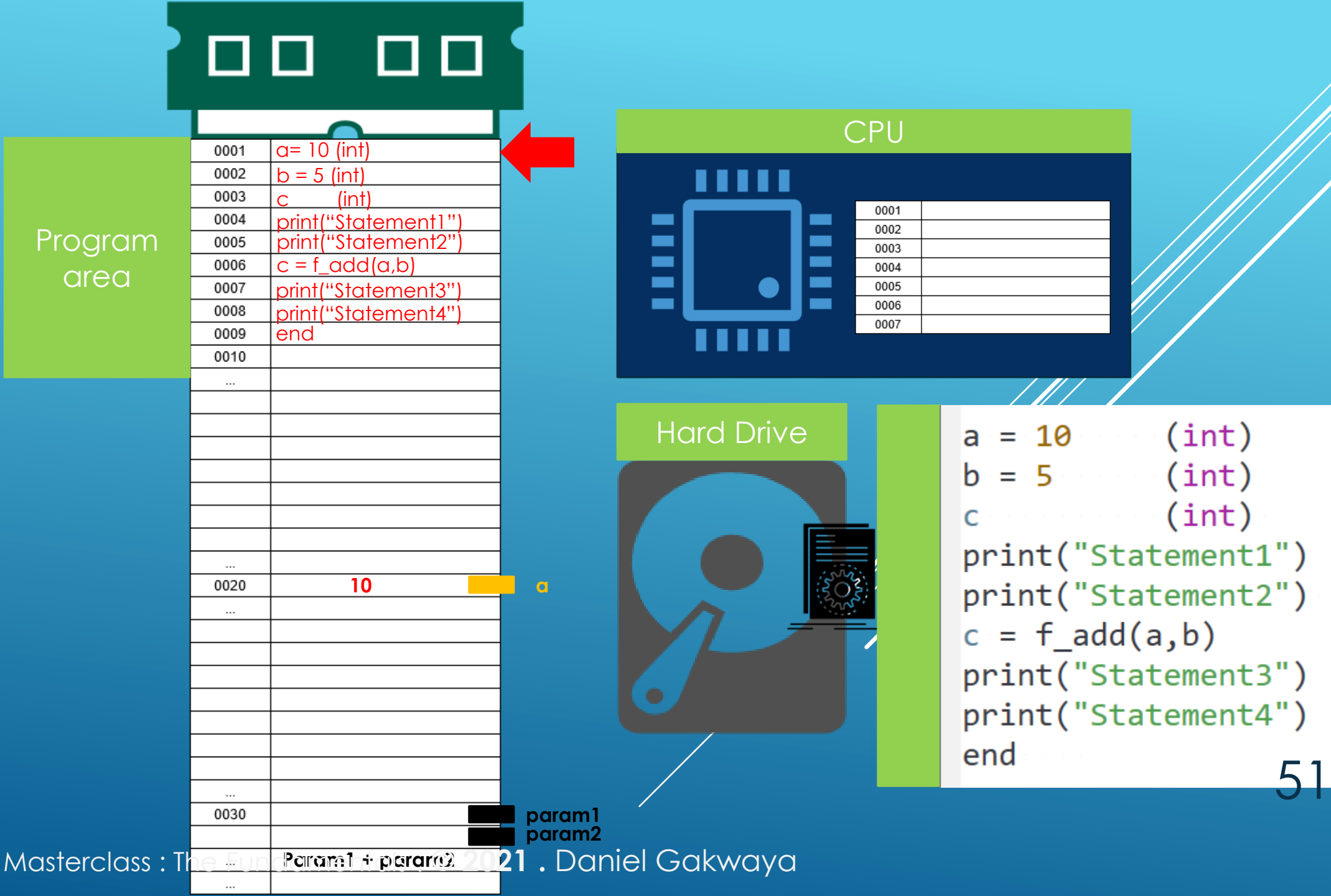
0007

Hard Drive

```
a = 10      (int)
b = 5       (int)
c           (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end
```

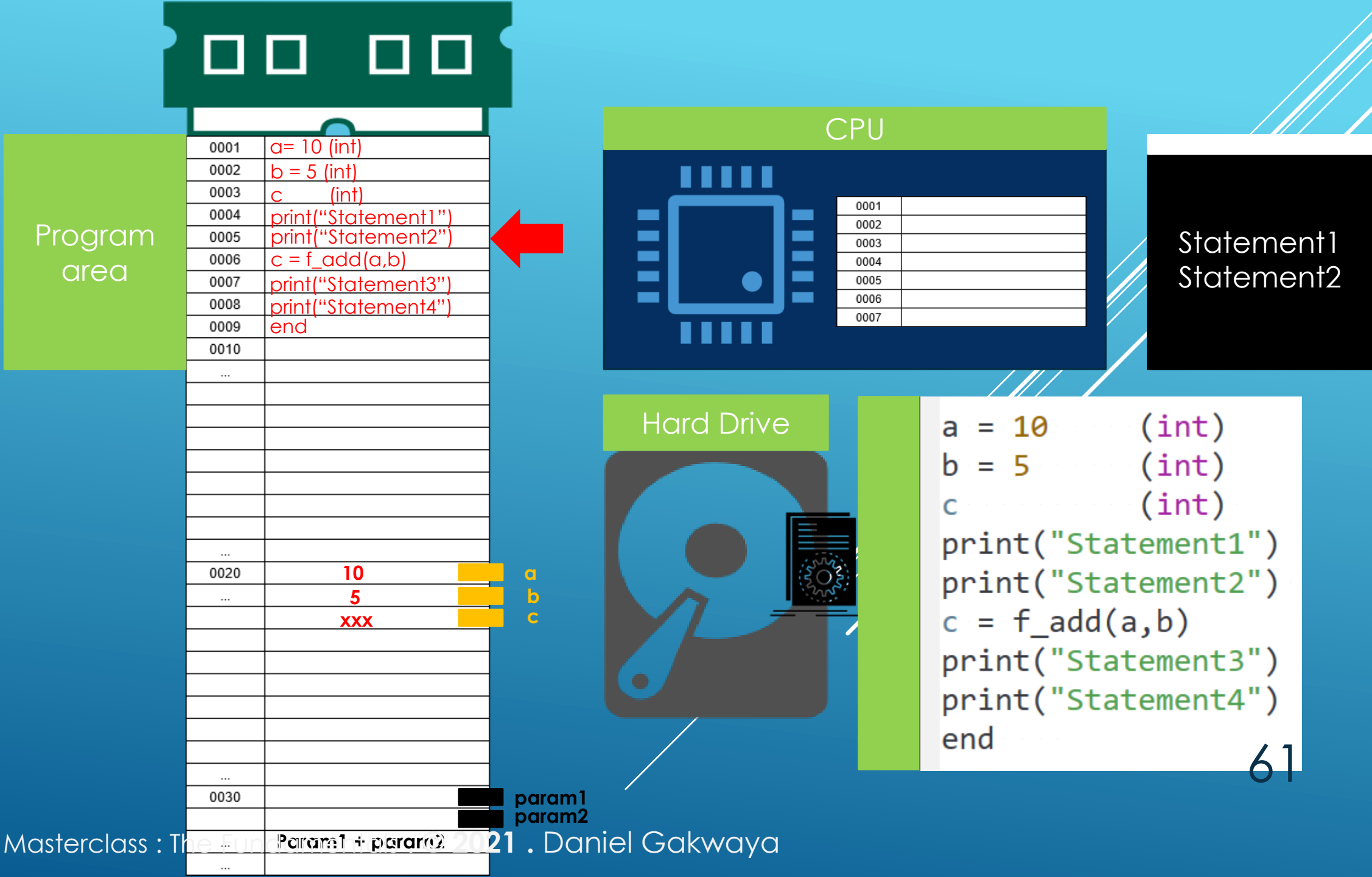
47

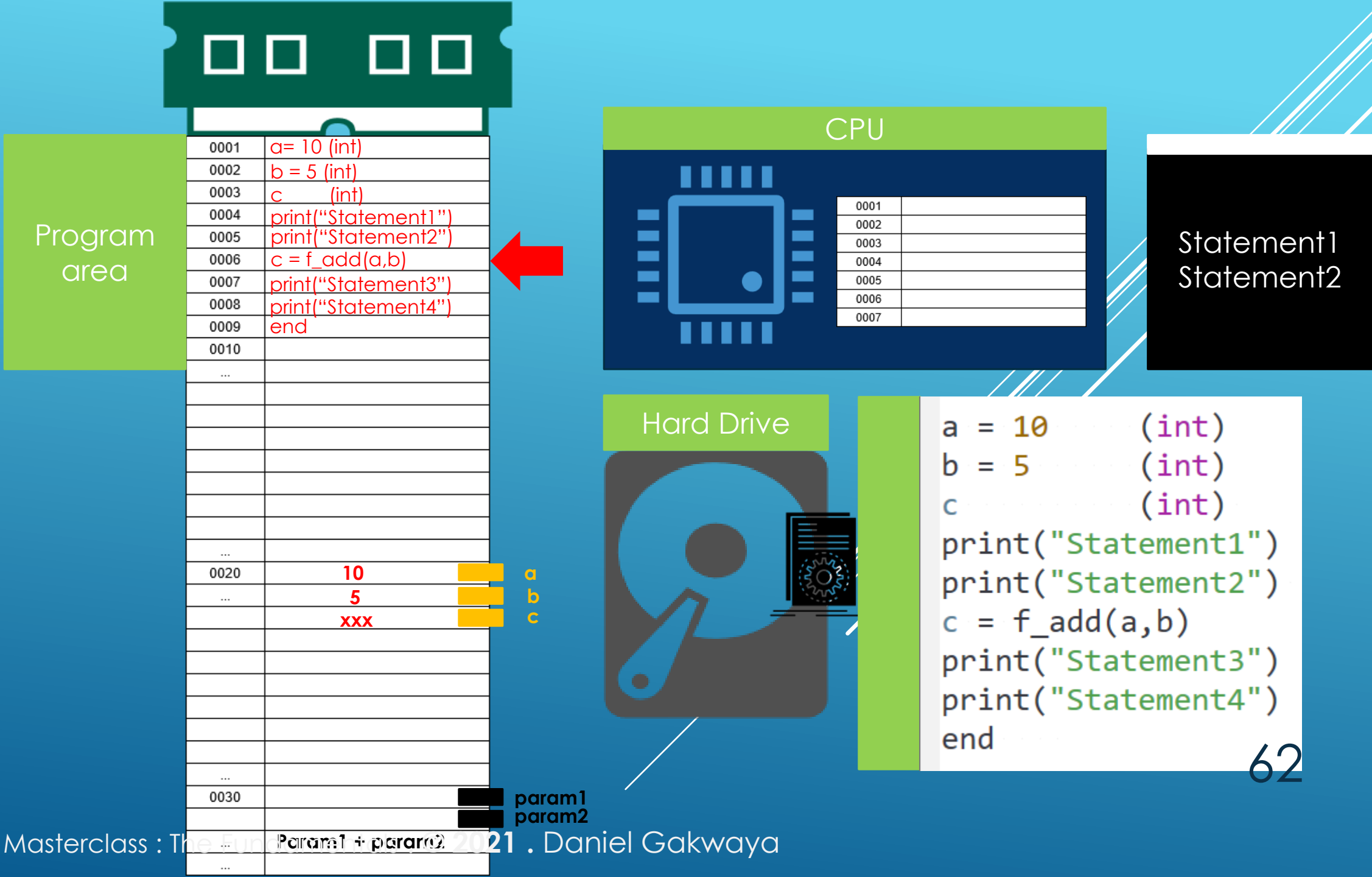


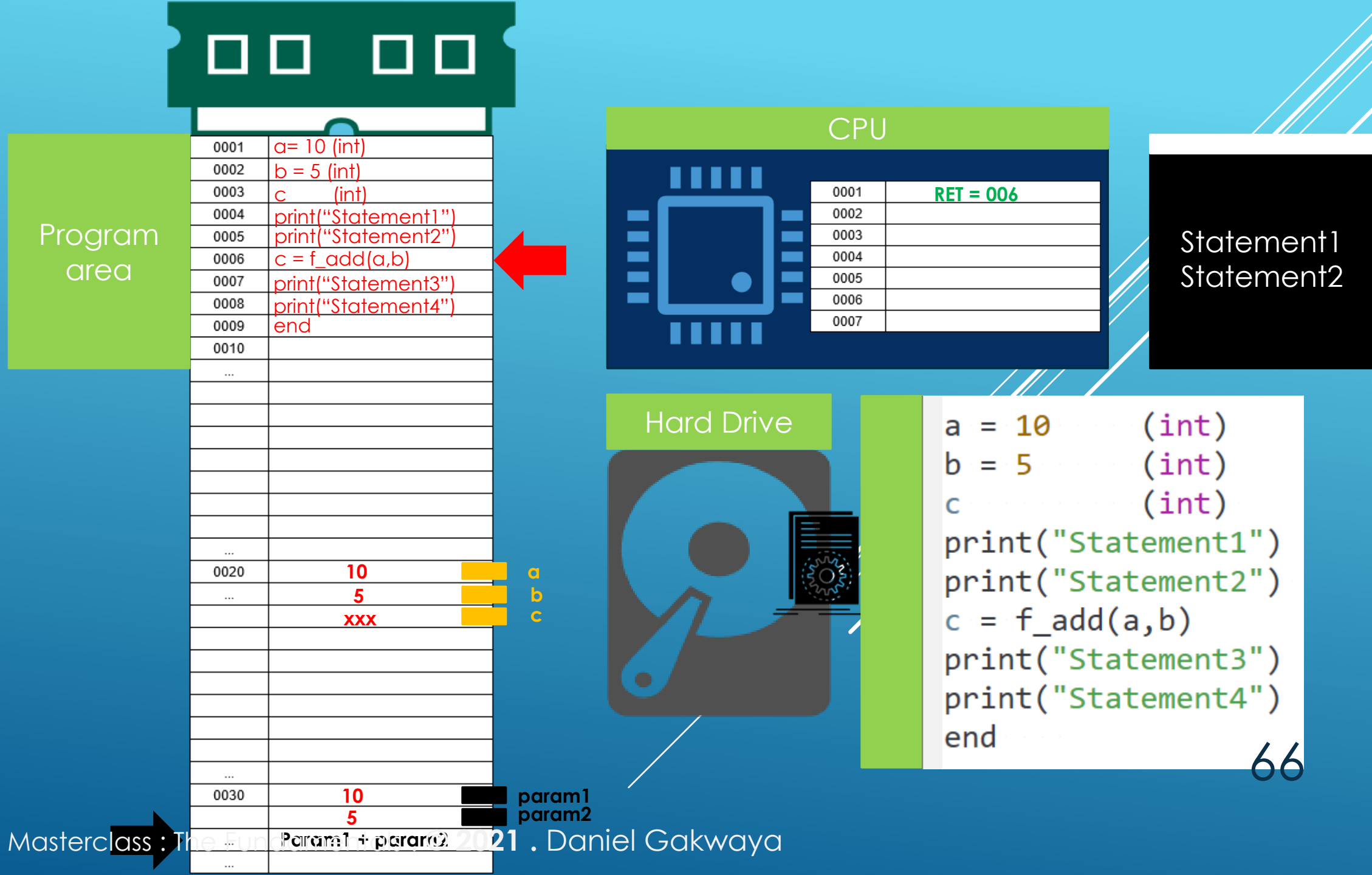






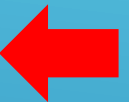






Program
area

0001	a= 10 (int)
0002	b = 5 (int)
0003	c (int)
0004	print("Statement1")
0005	print("Statement2")
0006	c = f_add(a,b)
0007	print("Statement3")
0008	print("Statement4")
0009	end
0010	
...	
...	
0020	10
...	5
	15
...	
0030	10
	5
...	



CPU

0001	RET = 006
0002	
0003	
0004	
0005	
0006	
0007	

Statement1
Statement2

Hard Drive



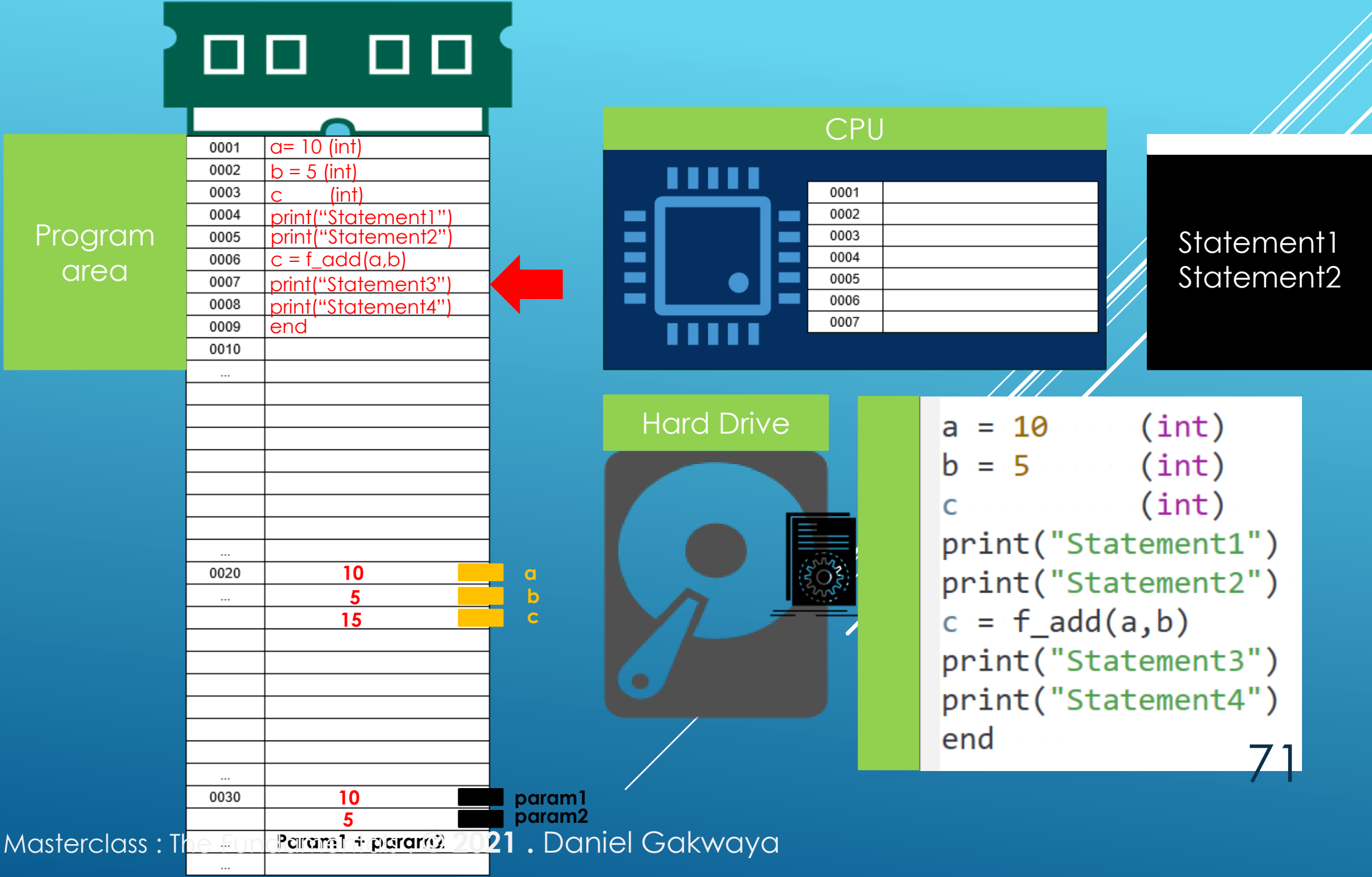
```

a = 10 (int)
b = 5 (int)
c (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end

```

a
b
c

param1
param2



Slide intentionally left empty

C++ core language Vs Standard library Vs STL



Core features

Standard library

STL

First Steps

80



Project template

First C++ Program



Comments



Errors

Statements and functions

```
#include <iostream>

int addNumbers(int first_number, int second_number){
    int sum = first_number + second_number;
    return sum;
}

int main(int argc, char **argv)
{
    int firstNumber = 12;
    int secondNumber = 9;

    int sum = firstNumber + secondNumber;

    sum = addNumbers(firstNumber,secondNumber);

    sum = addNumbers(34,7);

    std::cout << "The sum of the two numbers is : " << sum << std::endl;
    std::cout << "The sum of the two numbers is : " << addNumbers(23,8) << std::endl;
    return 0;
}
```



Data input and output



Execution Model

Core language Vs Standard Library Vs STL

Core features

The diagram consists of three rectangular boxes arranged horizontally. The first box on the left is light blue and contains the text 'Core features'. The second box in the middle is a darker blue and contains the text 'Standard library'. The third box on the right is the darkest blue and contains the text 'STL'. The boxes are separated by small gaps. The background is a gradient of blue with some white diagonal lines in the top right corner.

Standard library

STL

Slide intentionally left empty