BTH001
Object Oriented Programming
Lesson 04
Dynamic memory allocation

Different parts of memory

The primary memory is divided into two main parts:

- One part for which memory is allocated **statically**. This is done by the compiler . Memory is reserved for variables and functions.
- One part that is used for allocation of memory during execution, e.g. used for **dynamic** memory allocation.

Memory allocation

Static part of memory

101	…	
102	…	
103	12	value
104	nullptr	nrs
105	0	arr
106	0	
107	0	
108		
109		
110		
111		
112		

Dynamic part of memory

501	
502	
503	
504	
505	
506	
507	
508	
509	
510	
511	
512	

513	
514	
515	
516	
517	
518	
519	
520	
521	
522	
523	
524	

```
int main()
{
        int value = 12;
        int *nrs = nullptr;
        int arr[3] = {0, 0, 0};

        //….

        return 0;
}
```

Memory allocation

Static part of memory

101	…
102	…
103	12
104	501
105	0
106	0
107	0
108	
109	
110	
111	
112	

value

nrs

arr

Dynamic part of memory

501	0
502	0
503	0
504	0
505	
506	
507	
508	
509	
510	
511	
512	

513	
514	
515	
516	
517	
518	
519	
520	
521	
522	
523	
524	

```
int main()
{
        int value = 12;
        int *nrs = nullptr;
        int arr[3] = {0, 0, 0};
        nrs = new int[4]{0};


        return 0;
}
```

Memory allocation

Static part of memory

101	…
102	…
103	12
104	501
105	0
106	0
107	0
108	
109	
110	
111	
112	

value

nrs

arr

Dynamic part of memory

501	0
502	33
503	0
504	0
505	
506	
507	
508	
509	
510	
511	
512	

513	
514	
515	
516	
517	
518	
519	
520	
521	
522	
523	
524	

```
int main()
{
        int value = 12;
        int *nrs = nullptr;
        int arr[3] = {0, 0, 0};
        nrs = new int[4]{0};
        nrs[1] = 33;

        return 0;
}
```

Memory deallocation

Static part of memory

101	…
102	…
103	12
104	501
105	0
106	0
107	0
108	
109	
110	
111	
112	

value

nrs

arr

Dynamic part of memory

501	0
502	33
503	0
504	0
505	
506	
507	
508	
509	
510	
511	
512	

513	
514	
515	
516	
517	
518	
519	
520	
521	
522	
523	
524	

```
int main()
{
        int value = 12;
        int *nrs = nullptr;
        int arr[3] = {0, 0, 0};
        nrs = new int[4]{0};
        nrs[1] = 33;
        delete [] nrs;
        return 0;
}
```

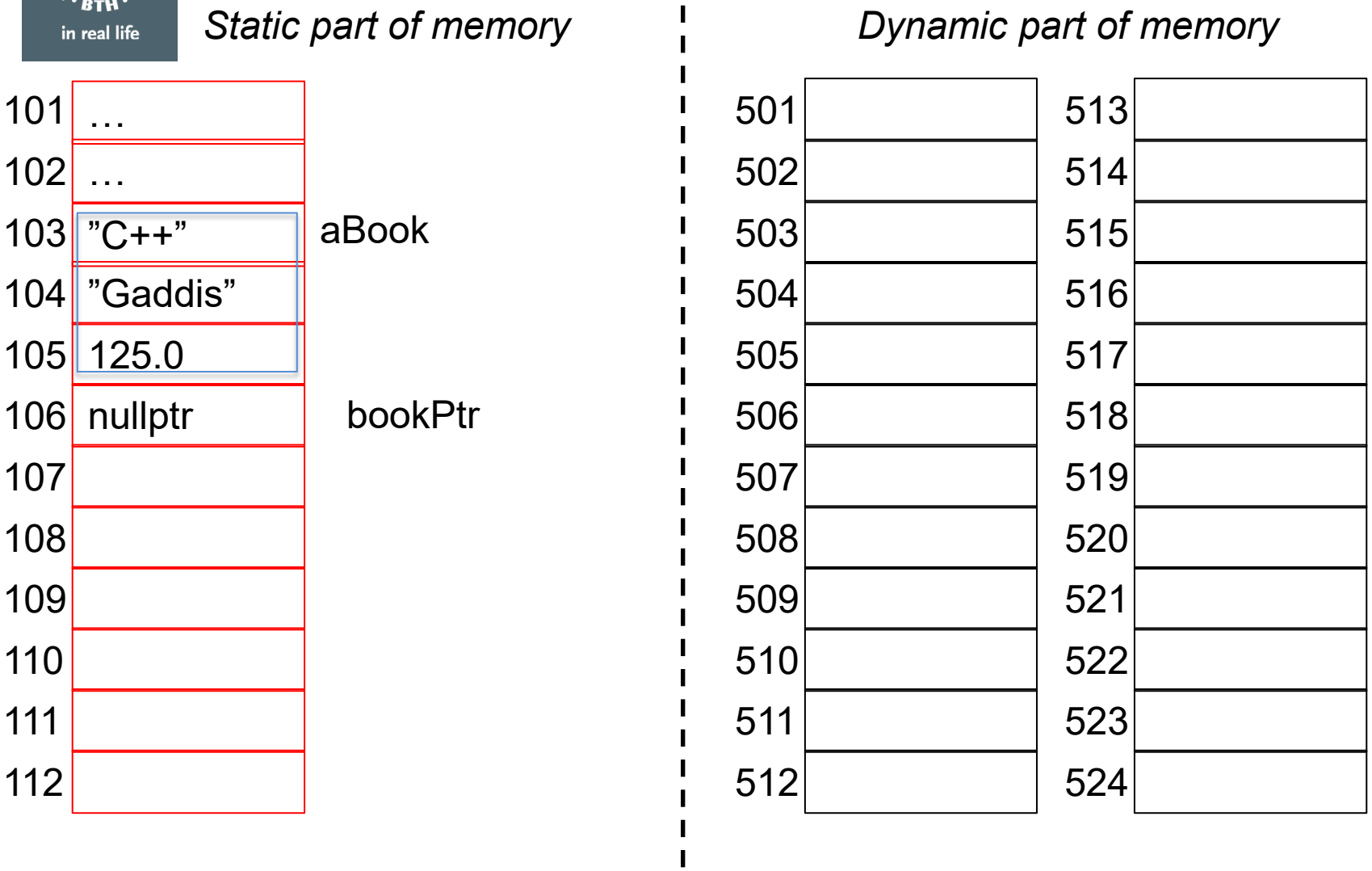
More static allocation

Assume the class type Book

Book aBook(”C++”, ”Gaddis”, 125.0);

Book *bookPtr = nullptr;

Memory allocation



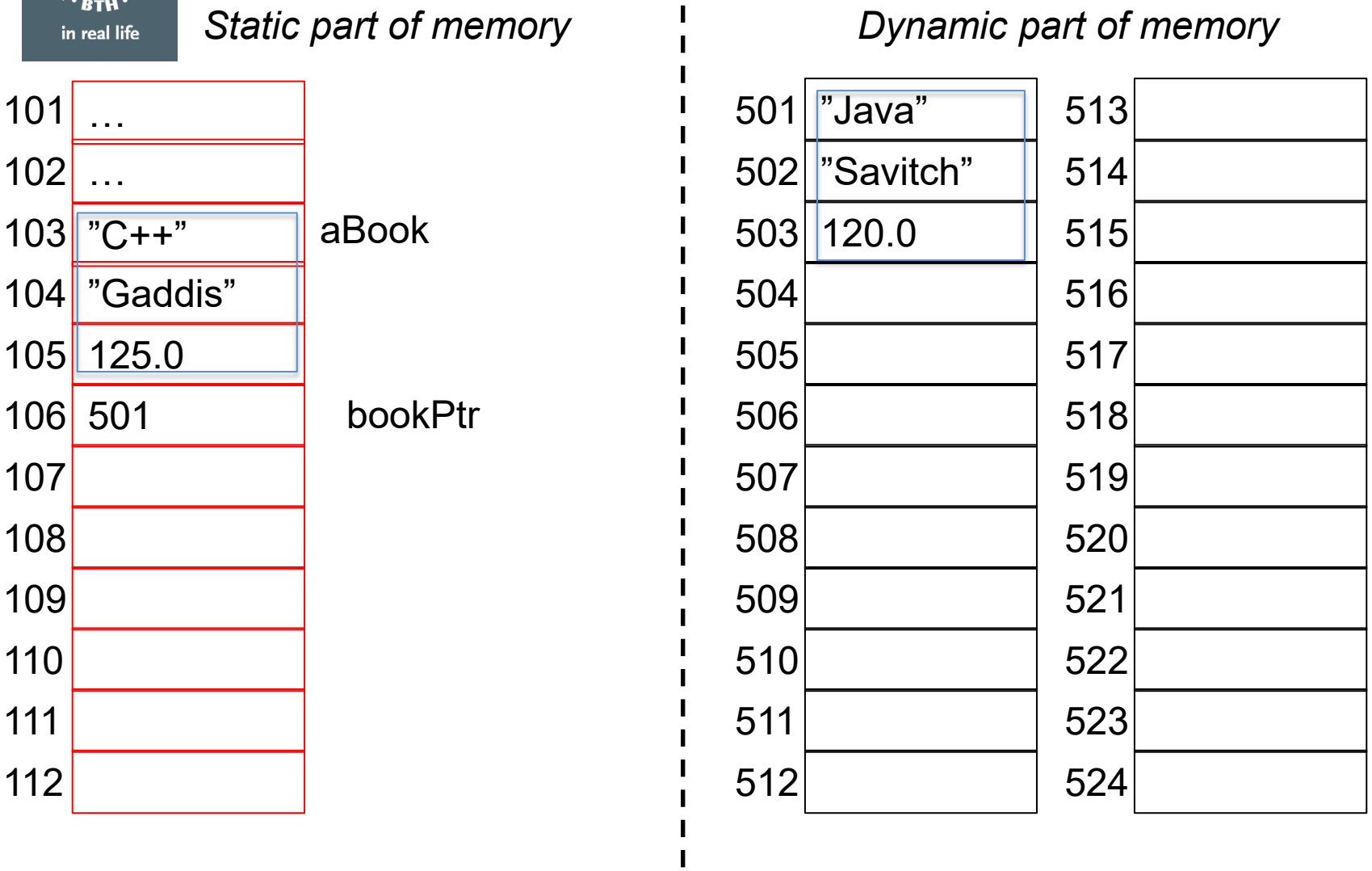
Dynamic allocation

Book aBook(”C++”, ”Gaddis”, 125.0);

*Book *bookPtr = nullptr;*

bookPtr = new Book(”Java”, ”Savitch”, 120.0);

Memory allocation



dot or arrow

Book aBook(”C++”, ”Gaddis”, 125.0);

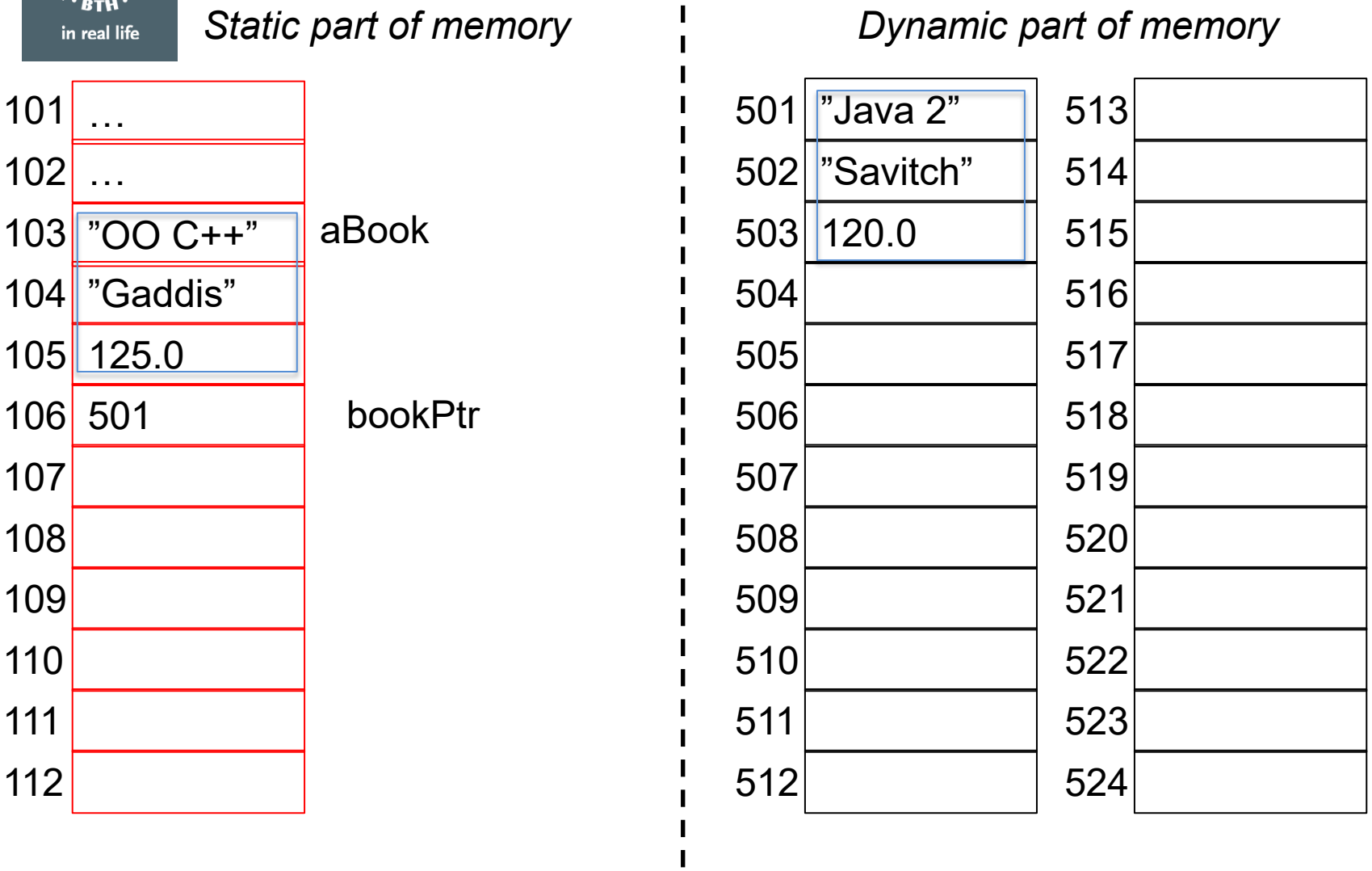
*Book *bookPtr = nullptr;*

bookPtr = new Book(”Java”, ”Savitch”, 120.0);

aBook.setTitle(”OO C++”); // use .

bookPtr->setTitle(”Java 2”); // use ->

Memory allocation



Dereferencing pointer

Book aBook(”C++”, ”Gaddis”, 125.0);

*Book *bookPtr = nullptr;*

bookPtr = new Book(”Java”, ”Savitch”, 120.0);

aBook.setTitle(”OO C++”); // use .

bookPtr->setTitle(”Java 2”); // use ->

(*bookPtr).changePrice(10); // dereferencing

Memory allocation

Static part of memory

101	…	
102	…	
103	”OO C++”	aBook
104	”Gaddis”	
105	125.0	
106	501	bookPtr
107		
108		
109		
110		
111		
112		

Dynamic part of memory

501	”Java 2”	513	
502	”Savitch”	514	
503	132.0	515	
504		516	
505		517	
506		518	
507		519	
508		520	
509		521	
510		522	
511		523	
512		524	

deallocating

Book aBook(”C++”, ”Gaddis”, 125.0);

*Book *bookPtr = nullptr;*

bookPtr = new Book(”Java”, ”Savitch”, 120.0);

aBook.setTitle(”OO C++”); // use .

bookPtr->setTitle(”Java 2”); // use ->

*(*bookPtr).changePrice(10); // dereferencing*

delete bookPtr; // call of destructor

Memory allocation