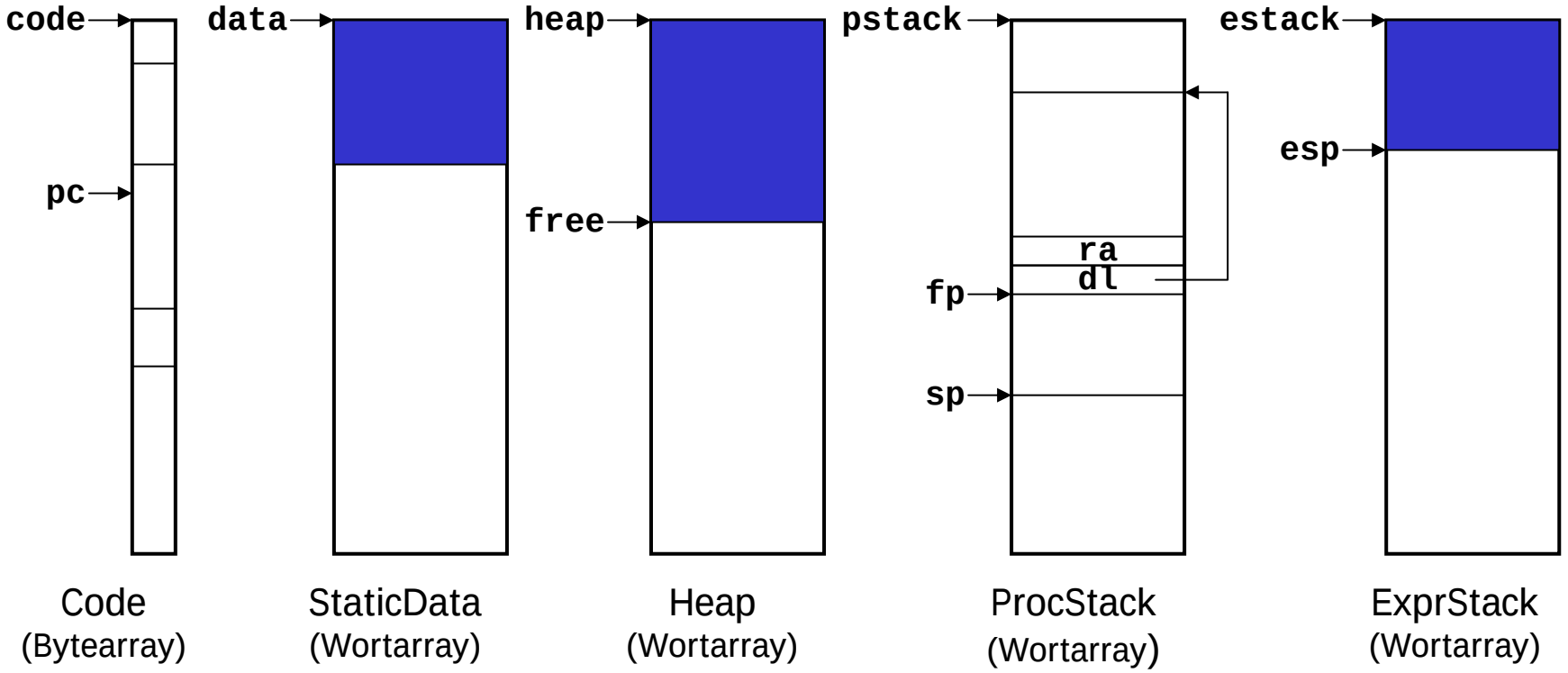
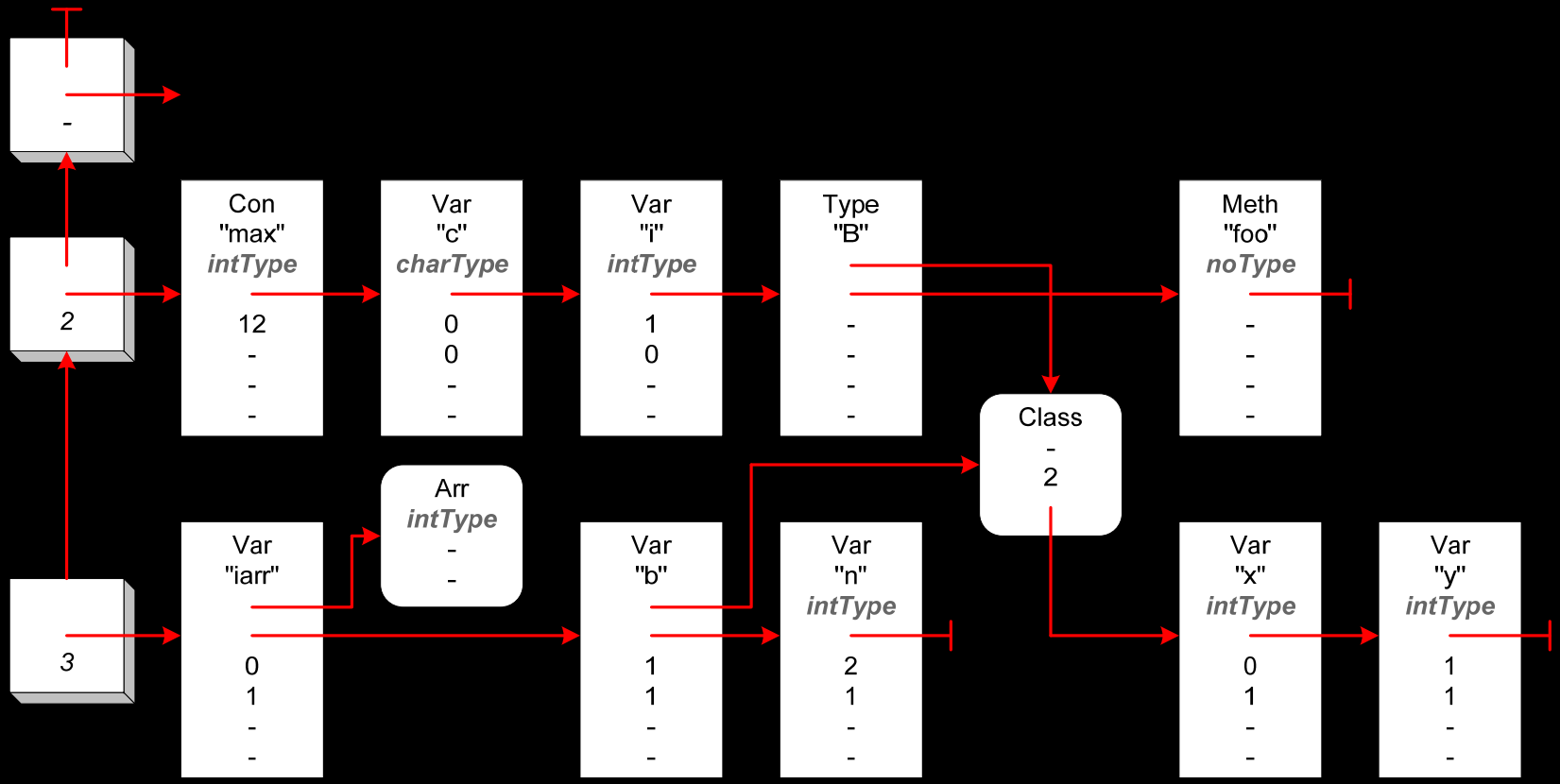
MicroJava VM: Speicher-Layout



Symboltabelle

Deklaration: **program A**

```
    final int max = 12;          // Konstante
    char c; int i;               // globale Variablen
    class B { int x, y; }        // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```



Bsp 1: **n = 3;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

const_3 = **2** byte

store_2

Bsp 2: **i = 10;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
const   10                                  = 8 byte
putstatic   1
```

Bsp 3: **n = 3 + i;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
const_3                                   = 6 byte
getstatic   1
add
store_2
```

Bsp 4: **n = 3 + i * max - n;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
const_3                                   = 14 byte
getstatic   1
const   12
mul
add
load_2
sub
store_2
```

Bsp 5: **iarr[5] = 10;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
load_0                                   = 8 byte
const_5
const   10
astore
```

Bsp 6: **b.y = iarr[5] * 3;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
load_1
load_0
const_5
aload
const_3
mul
putfield   1
```

= **9** byte

Bsp 7: **n--;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
inc   2   -1
```

= **3** byte

Bsp 8: **i--;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
getstatic   1
const_m1
add
putstatic   1
```

= **8** byte

Bsp 9: **b.y--;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
load_1
dup
getfield   1
const_m1
add
putfield   1
```

= **10** byte

Bsp 10: **iarr[0]--;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
load_0
const_0
dup2
aload
const_m1
add
astore
```

= **7** byte

Bsp 11: **if (i <= n)** n=0**;**

Deklaration:

```
program A
    final int max = 12;          // Konstante
    char c; int i;               // globale Variablen
    class B { int x, y; }        // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:   getstatic 1
13:   load_2
14:   jgt 5        (--> 19)
17:   const_0
18:   store_2
19:   ...
```

Bsp 12: **if (i <= n && n < 0)** n=0**;**

Deklaration:

```
program A
    final int max = 12;          // Konstante
    char c; int i;               // globale Variablen
    class B { int x, y; }        // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:   getstatic 1
13:   load_2
14:   jgt 10              (--> 24)
17:   load_2
18:   const_0
19:   jge 5               (--> 24)
22:   const_0
23:   store_2
24:   ...
```

Bsp 13: **if (i <= n || n < 0)** n=0**;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:   getstatic 1
13:   load_2
14:   jle 8              (--> 22)
17:   load_2
18:   const_0
19:   jge 5              (--> 24)
22:   const_0
23:   store_2
24:   ...
```

Bsp 14: **if (i<=n || n<0 && i>0)** n=0;

Deklaration:

```
program A
    final int max = 12;          // Konstante
    char c; int i;               // globale Variablen
    class B { int x, y; }        // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:   getstatic 1
13:   load_2
14:   jle 15              (--> 29)
17:   load_2
18:   const_0
19:   jge 12              (--> 31)
22:   getstatic 1
25:   const_0
26:   jle 5               (--> 31)
29:   const_0
30:   store_2
31:   ...
```

Bsp 15: **while (i<=n)** n++**;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:  getstatic 1
13:  load_2
14:  jgt 9              (--> 23)
17:  inc 2 1
20:  jmp -10            (--> 10)
23:  ...
```

Bsp 16: **if (i <= n)** n=0 **else** n=1**;**

Deklaration:

```
program A
    final int max = 12;        // Konstante
    char c; int i;             // globale Variablen
    class B { int x, y; }      // innere Klasse mit Feldern
{   void foo ()   int[] iarr; B b; int n;   {...}    }
```

```
10:   getstatic 1
13:   load_2
14:   jgt 8        (--> 22)
17:   const_0
18:   store_2
19:   jmp 5        (--> 24)
22:   const_1
23:   store_2
24:   ...
```