

Memory Allocation

```
public class Point {
    public Point(int x, int y) {
        px = x;
        py = y;
    }
    public void move(int dx,
                    int dy) {
        px += dx;
        py += dy;
    }
    /* instance variables */
    private int px;
    private int py;
}
```

```
public class MyProgram
    extends ConsoleProgram {
    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

Memory Allocation

↓ heap

stack ↑

```
public class MyProgram
    extends ConsoleProgram {

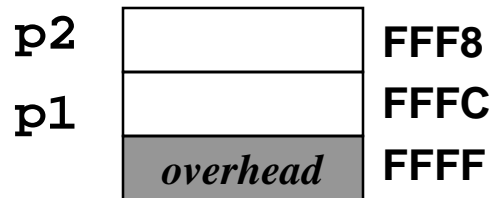
    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

Memory Allocation

↓ heap

stack ↑

```
public class MyProgram
    extends ConsoleProgram {
    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```



Memory Allocation

↓ heap

	<i>overhead</i>	1000
px	2	1004
py	3	1008

stack ↑

p2		FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

Memory Allocation

↓ heap

	<i>overhead</i>	1000
pX	2	1004
pY	3	1008
	<i>overhead</i>	100C
pX	4	1010
pY	5	1014

stack ↑

p2	100C	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

Memory Allocation

↓ heap

	<i>overhead</i>	1000
px	2	1004
py	3	1008
	<i>overhead</i>	100C
px	4	1010
py	5	1014

stack ↑

	dy	11	FFE8
	dx	10	FFEC
this		1000	FFF0
	<i>overhead</i>		FFF4
	p2	100C	FFF8
	p1	1000	FFFC
	<i>overhead</i>		FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

```
public void move(int dx,
                 int dy) {

    px += dx;
    py += dy;
}
```

Memory Allocation

↓ heap

	<i>overhead</i>	1000
px	12	1004
py	3	1008
	<i>overhead</i>	100C
px	4	1010
py	5	1014

stack ↑

	dy	11	FFE8
	dx	10	FFEC
this		1000	FFF0
	<i>overhead</i>		FFF4
	p2	100C	FFF8
	p1	1000	FFFC
	<i>overhead</i>		FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

```
public void move(int dx,
                 int dy) {
    px += dx;
    py += dy;
}
```

Memory Allocation

↓ heap

stack ↑

	<i>overhead</i>	1000			
px	12	1004			
py	14	1008			
	<i>overhead</i>	100C			
px	4	1010			
py	5	1014	dy	11	FFE8
			dx	10	FFEC
			this	1000	FFF0
				<i>overhead</i>	FFF4
			p2	100C	FFF8
			p1	1000	FFFC
				<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

```
public void move(int dx,
                 int dy) {

    px += dx;
    py += dy;
}
```

Memory Allocation

↓ heap

stack ↑

	<i>overhead</i>	1000
px	12	1004
py	14	1008
	<i>overhead</i>	100C
px	4	1010
py	5	1014

dy	11	FFE8
dx	10	FFEC
this	1000	FFF0
	<i>overhead</i>	FFF4
p2	100C	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

reclaimed when
method is done

"popped off stack"

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
    }
}
```

```
public void move(int dx,
                 int dy) {

    px += dx;
    py += dy;
}
```

Memory Allocation

↓ heap

	<i>overhead</i>	1000
pX	12	1004
pY	14	1008
	<i>overhead</i>	100C
pX	4	1010
pY	5	1014

stack ↑

p2	100C	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
        ...
    }
}
```

Pointer Viewpoint

↓ heap

	<i>overhead</i>	1000
pX	12	1004
pY	14	1008
	<i>overhead</i>	100C
pX	4	1010
pY	5	1014

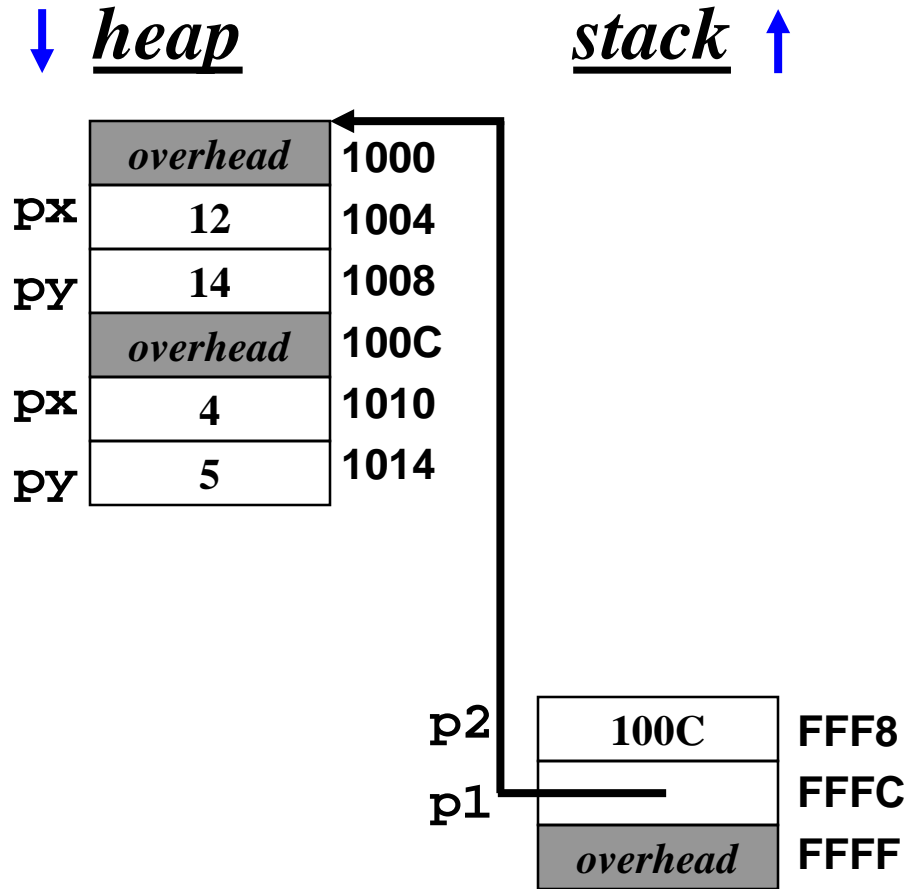
stack ↑

p2	100C	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
        ...
    }
}
```

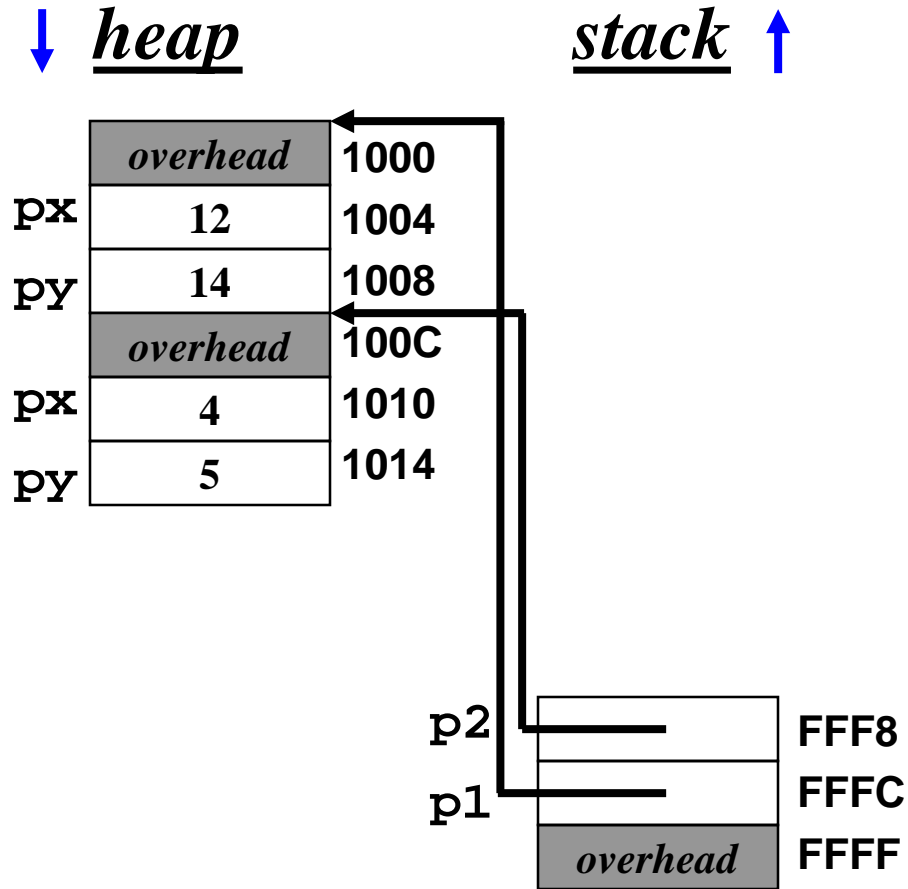
Pointer Viewpoint



```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
        ...
    }
}
```

Pointer Viewpoint



```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(2, 3);
        Point p2 = new Point(4, 5);
        p1.move(10, 11);
        ...
    }
}
```

Another Example

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

stack ↑

```
public class MyProgram
    extends ConsoleProgram {

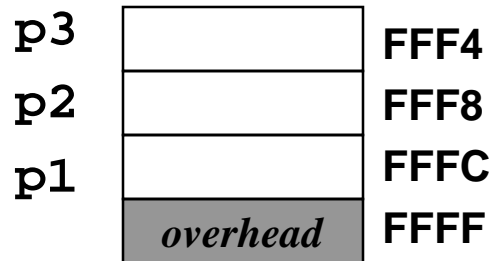
    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

stack ↑



```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	1	1004
pY	2	1008

stack ↑

p3		FFF4
p2		FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	1	1004
pY	2	1008

stack ↑

p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
px	1	1004
py	2	1008

stack ↑

dy	4	FFE4
dx	3	FFE8
this	1000	FFEC
	<i>overhead</i>	FFF0
p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

```
public void move(int dx,
                 int dy) {

    px += dx;
    py += dy;
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
px	4	1004
py	2	1008

stack ↑

dy	4	FFE4
dx	3	FFE8
this	1000	FFEC
	<i>overhead</i>	FFF0
p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

```
public void move(int dx,
                 int dy) {
    px += dx;
    py += dy;
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
px	4	1004
py	6	1008

stack ↑

dy	4	FFE4
dx	3	FFE8
this	1000	FFEC
	<i>overhead</i>	FFF0
p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

```
public void move(int dx,
                 int dy) {
    px += dx;
    py += dy;
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
px	4	1004
py	6	1008

stack ↑

dy	4	FFE4
dx	3	FFE8
this	1000	FFEC
	<i>overhead</i>	FFF0
p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

reclaimed when
method is done

"popped off stack"

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

```
public void move(int dx,
                 int dy) {

    px += dx;
    py += dy;
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	4	1004
pY	6	1008

stack ↑

p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	4	1004
pY	6	1008

stack ↑

p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	4	1004
pY	6	1008

stack ↑

p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

Another Example

↓ heap

	<i>overhead</i>	1000
pX	4	1004
pY	6	1008

stack ↑

p3		FFF4
p2	1000	FFF8
p1	1000	FFFC
	<i>overhead</i>	FFFF

```
public class MyProgram
    extends ConsoleProgram {

    public void run() {
        Point p1 = new Point(1, 2);
        Point p2 = p1;
        p2.move(3, 4);

        Point p3;
        p3.move(1, 1);
    }
}
```

ERROR!

**p3 is not pointing to a
valid object!**

(null dereference)