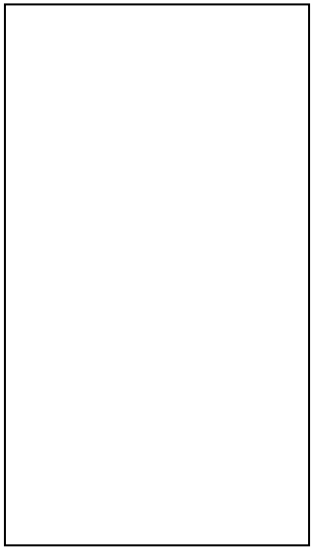
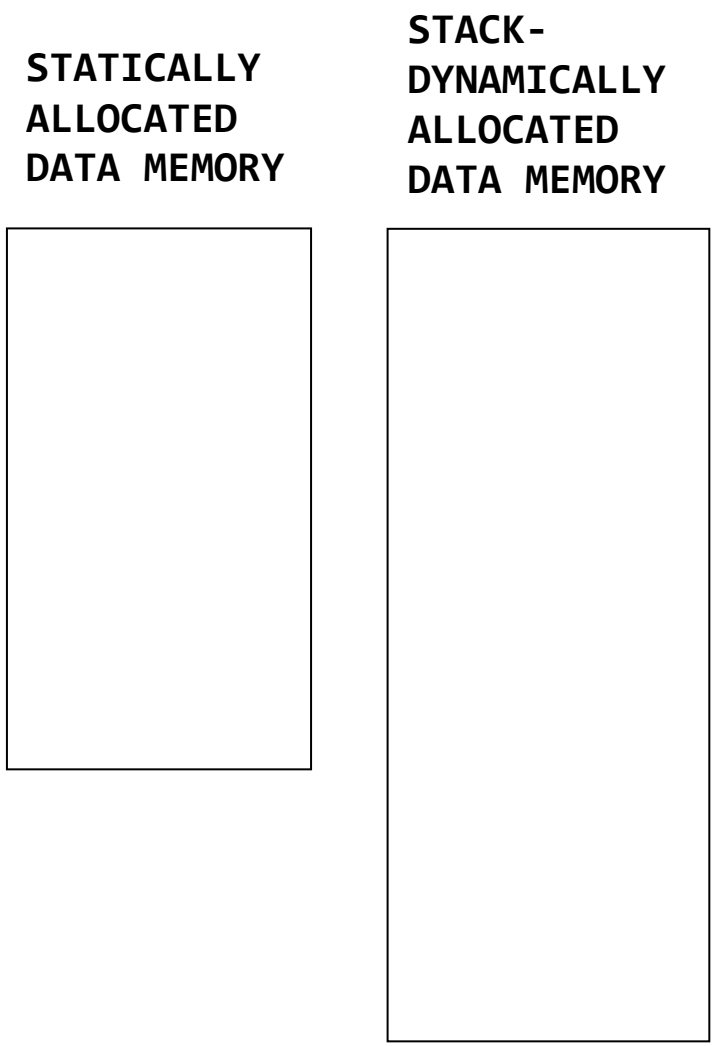
The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)

The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)

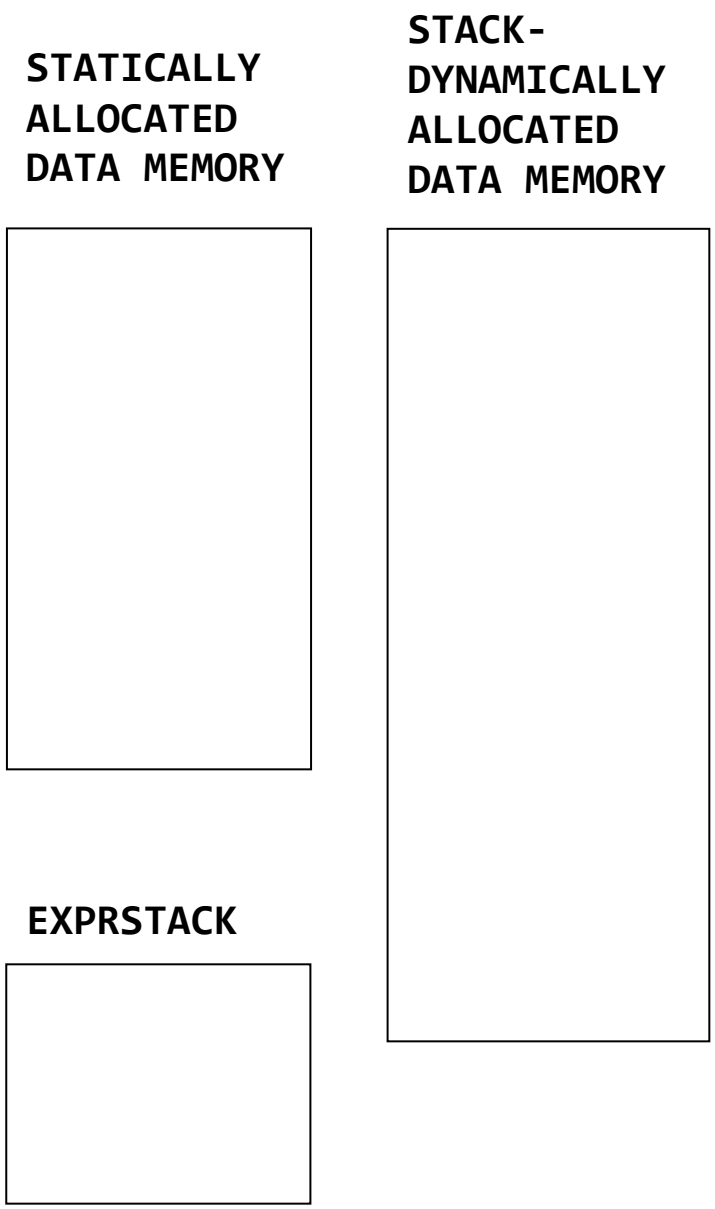
STATICALLY
ALLOCATED
DATA MEMORY



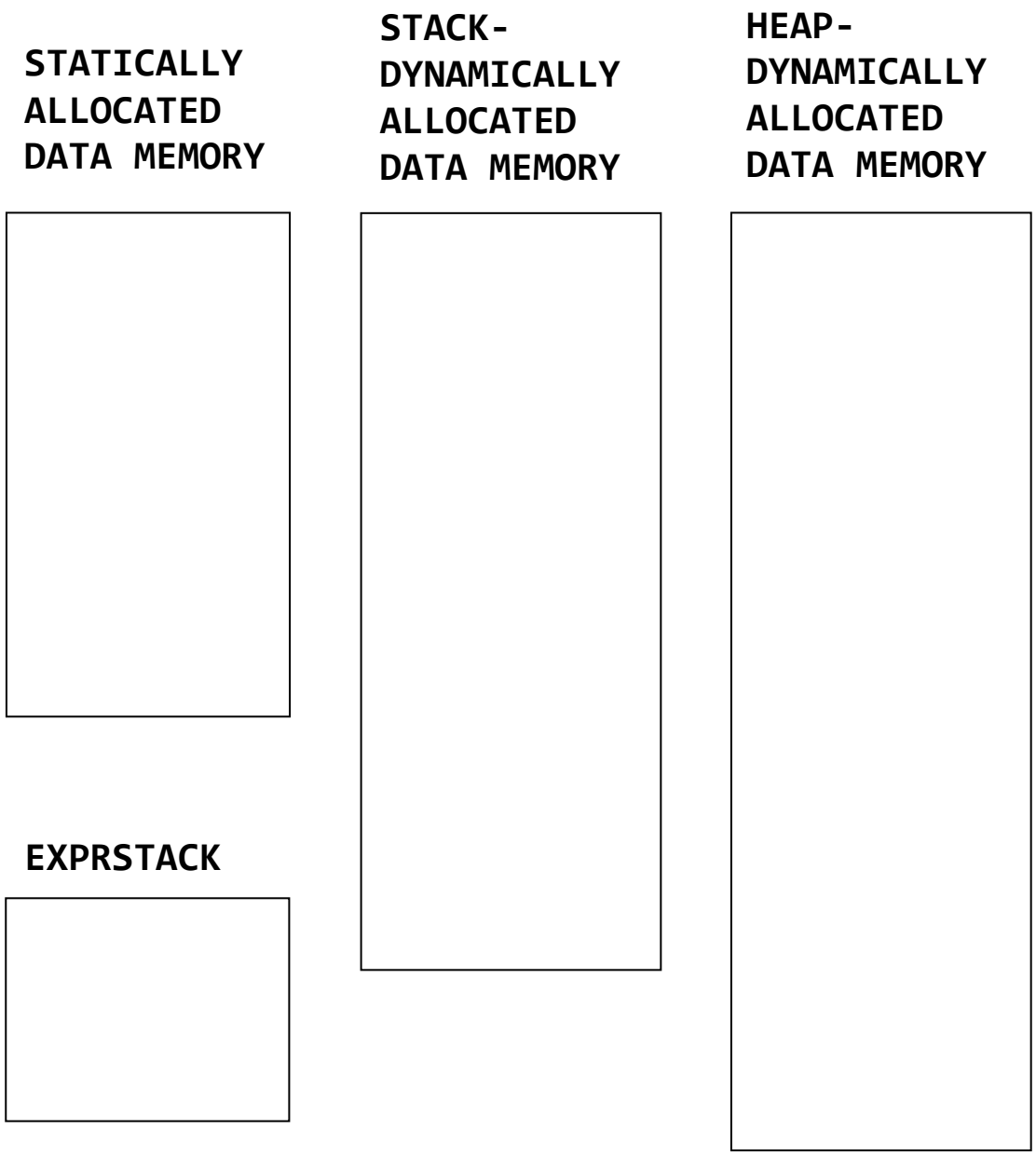
The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)



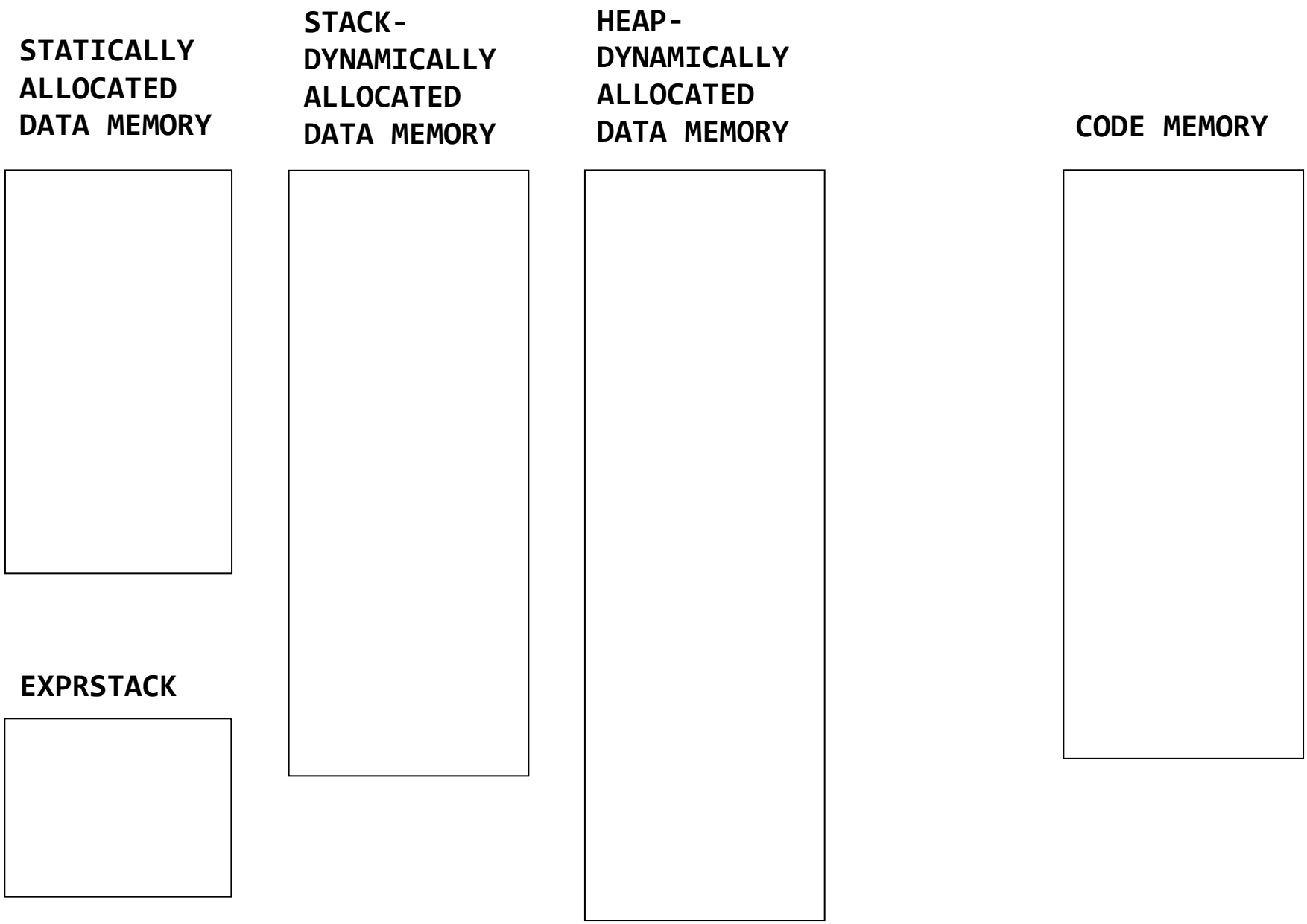
The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)



The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)



The TinyJ Virtual Machine’s Memory (VM Registers are Not Shown)



An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```

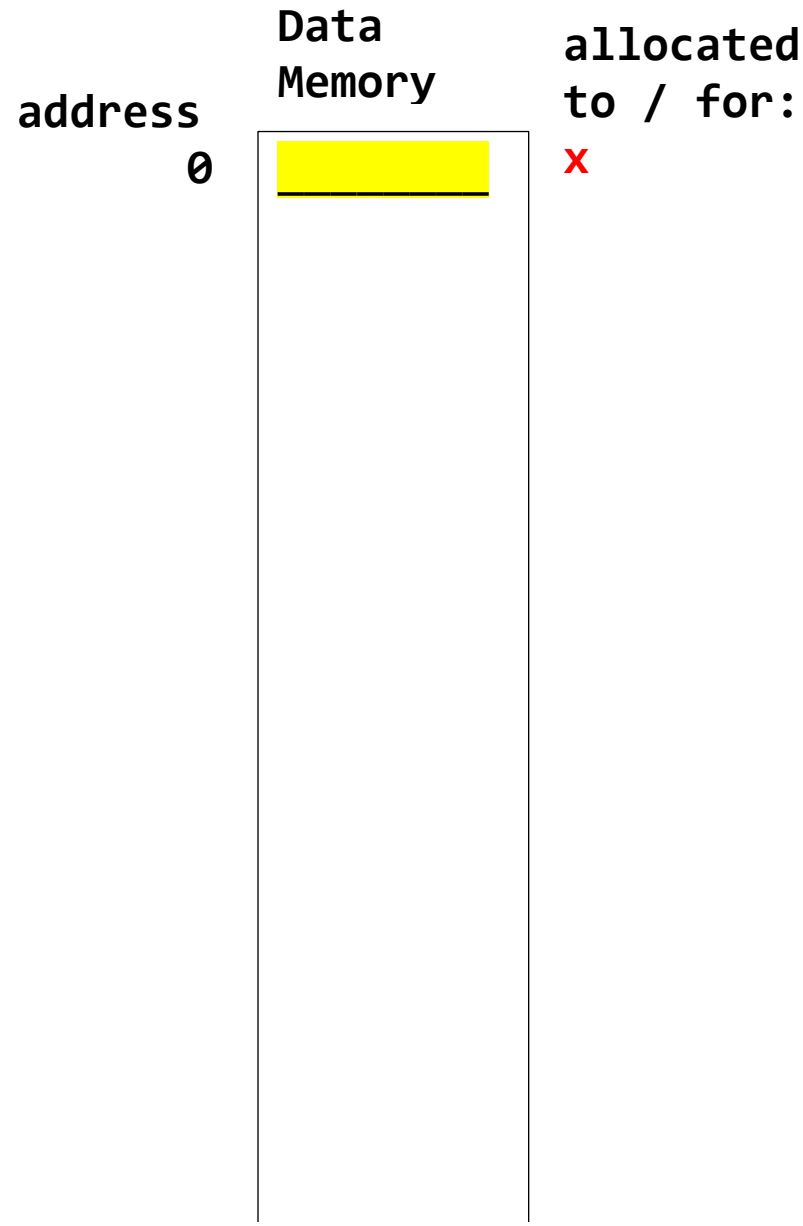
An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
          = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```

address	Data Memory	allocated to / for:

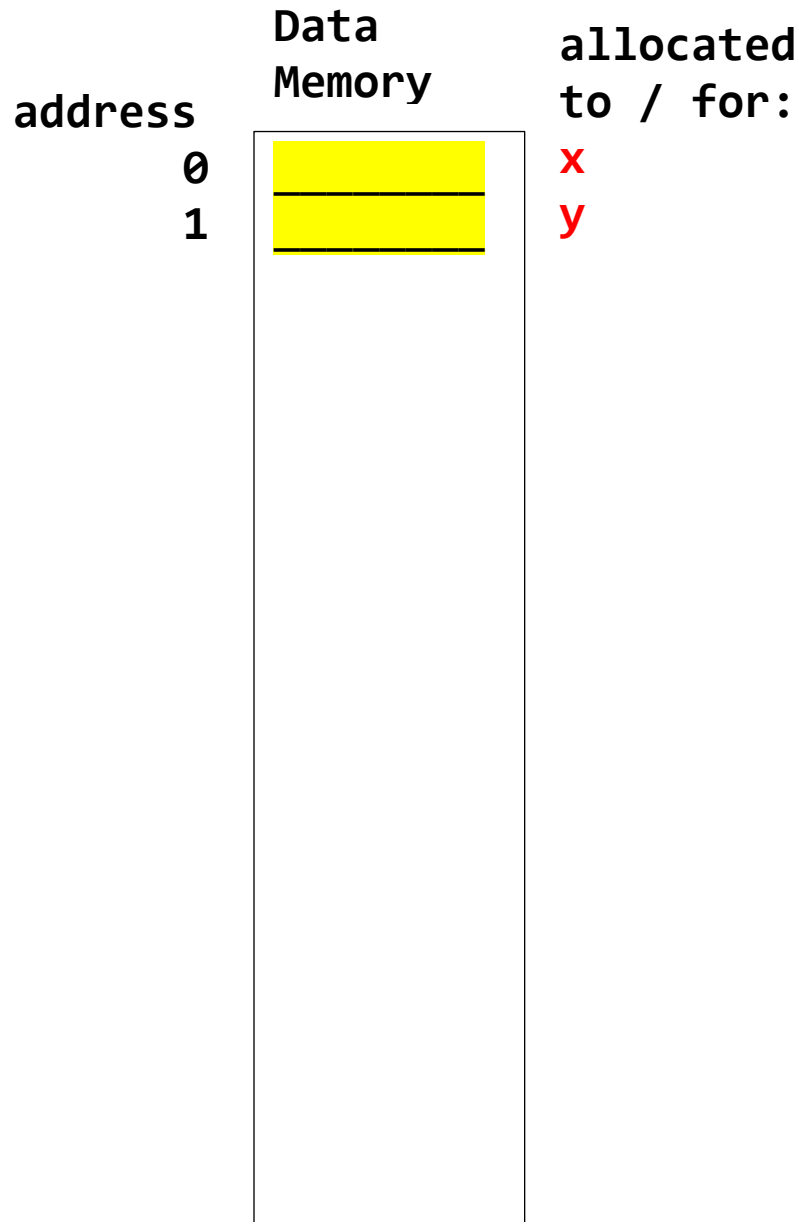
An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```



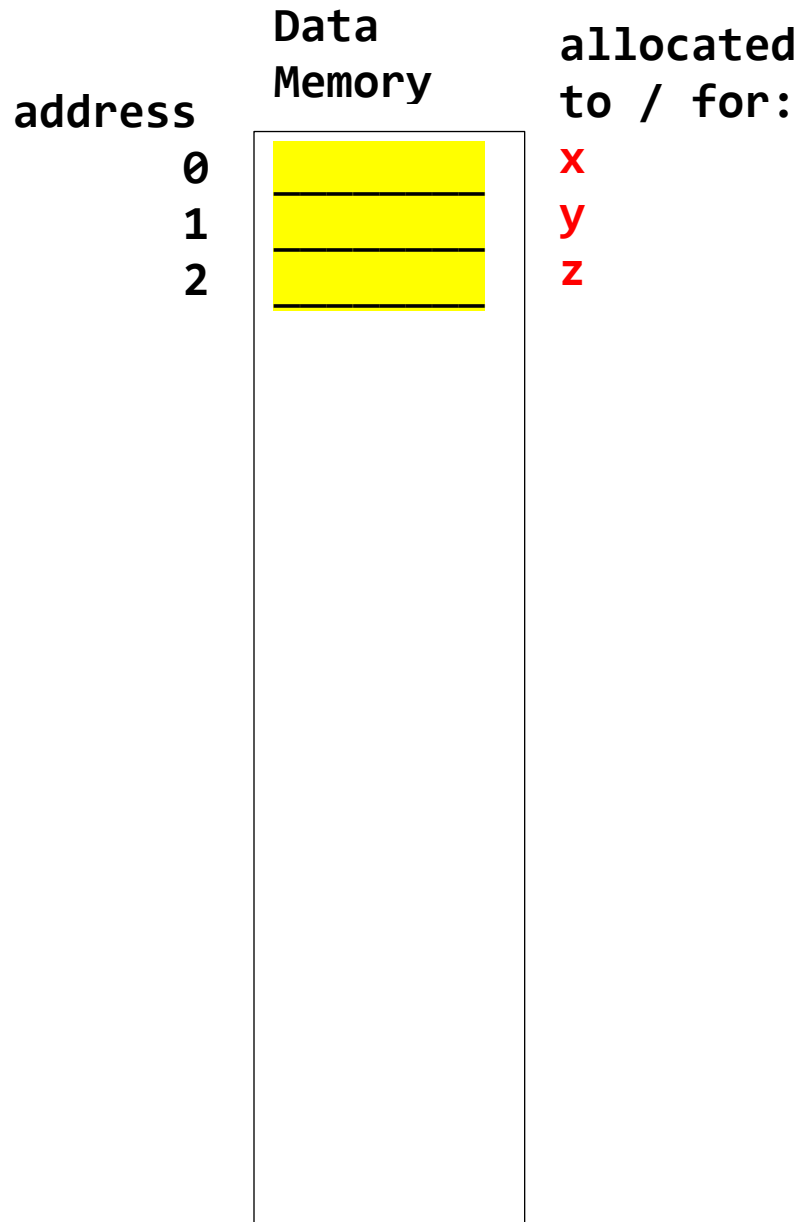
An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```



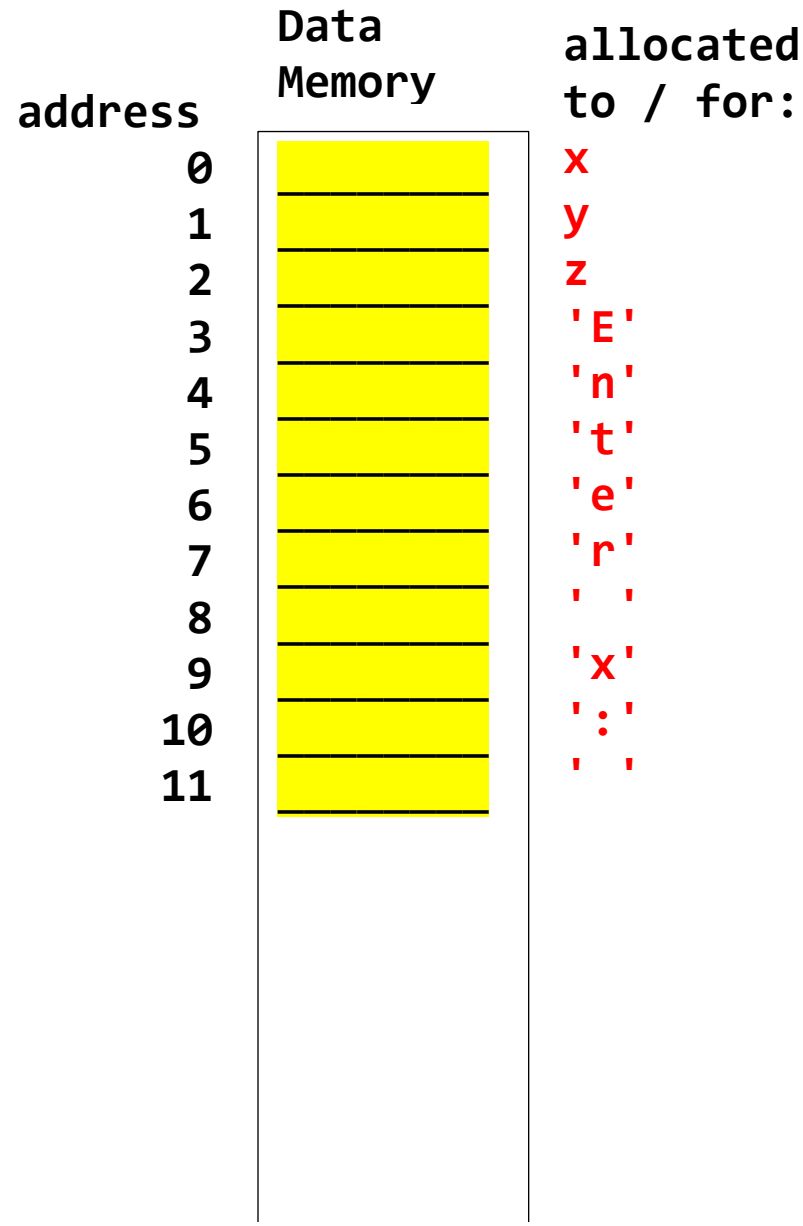
An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```



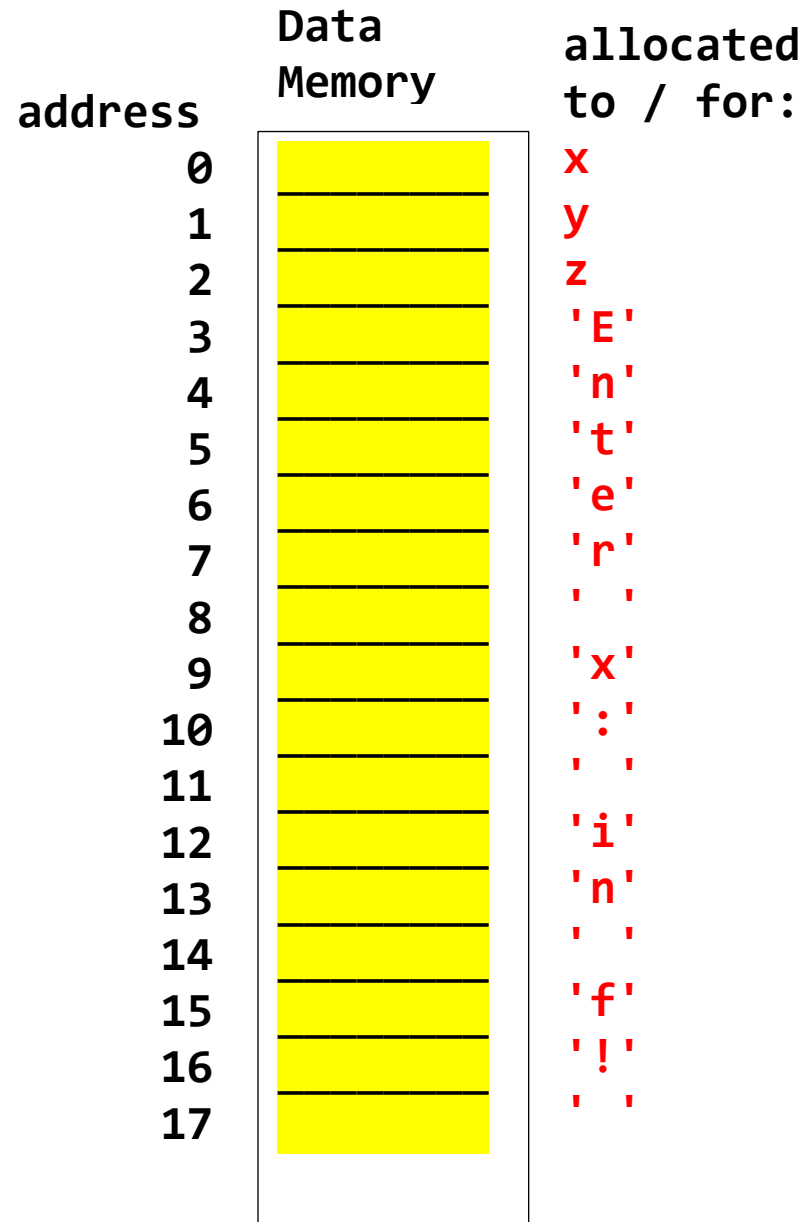
An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```



An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

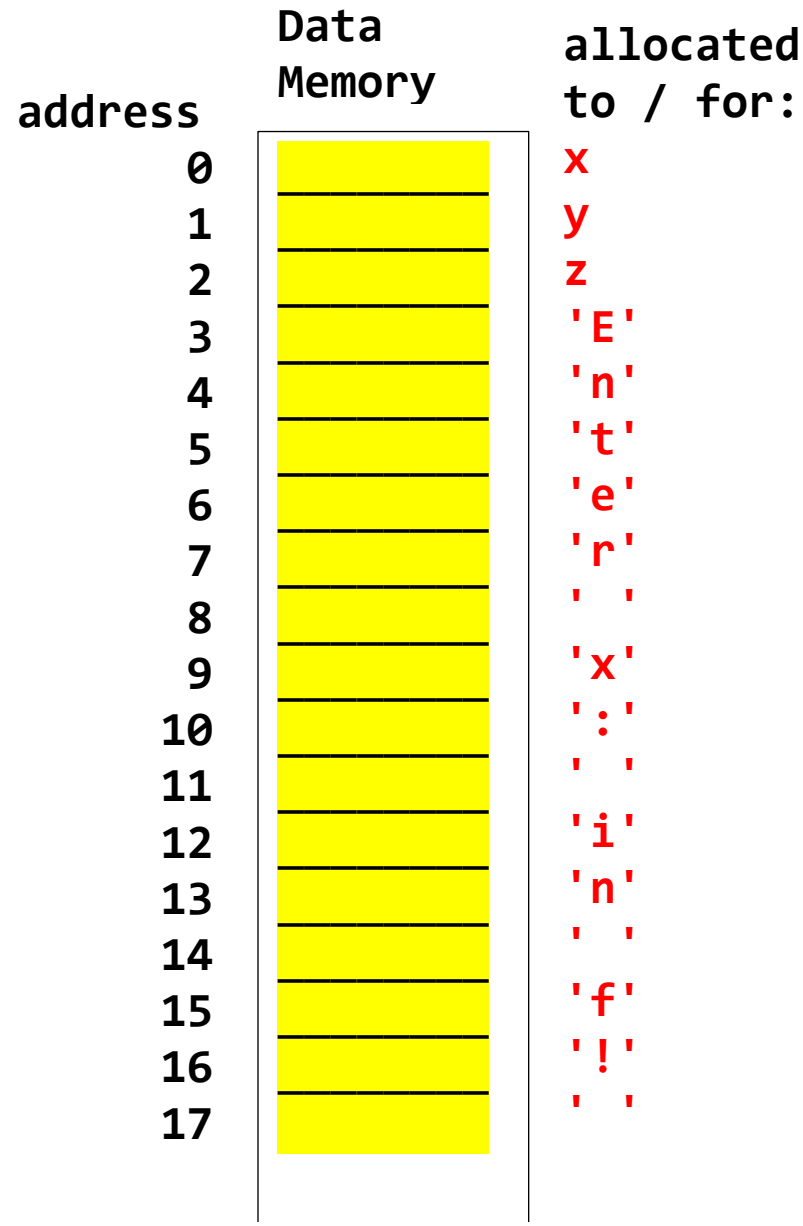
```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```



An Example: Which memory locations are statically allocated for the following TinyJ program? What variables / data are they allocated to / for?

```
import java.util.Scanner;

class Simple {

 static Scanner input
           = new Scanner(System.in);

 static int x, y = 10, z;

 public static void main(String args[])
 {
   System.out.print("Enter x: ");
   x = input.nextInt();
   f(17, y, x-y);
   System.out.println(y + f(21,22,23));
 }

 static int f (int a, int b, int c)
 {
   int v[], w, u = x;
   System.out.print("in f! ");
   return y - a % u;
 }
}
```

Answer: 18 locations are statically allocated as shown above.



An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)   ANSWER:
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)   ANSWER: 11
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

See pp. 3-4 of:
https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

offset | **Stackframe of any call of my_func** | allocated to

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset		allocated to
-1		return addr
0		dynamic link

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-2	z
-1	return addr
0	dynamic link

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
      ...
  }
  else {
    int e, f;
      ...
    int g;
      ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-3	y
-2	z
-1	return addr
0	dynamic link

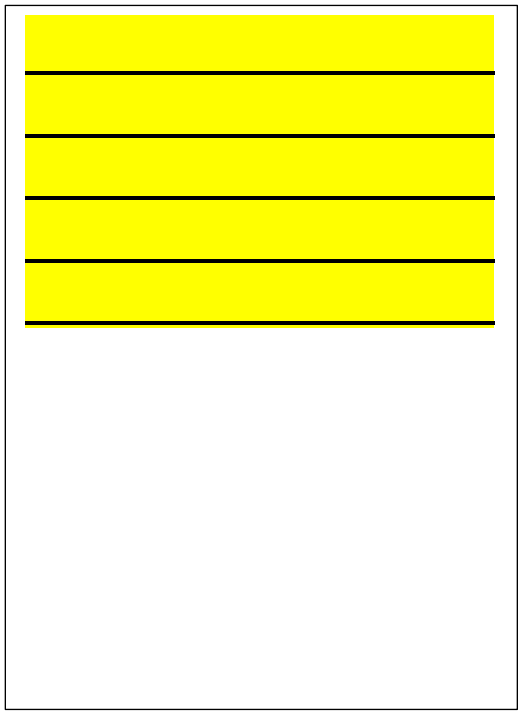
An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link



An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b

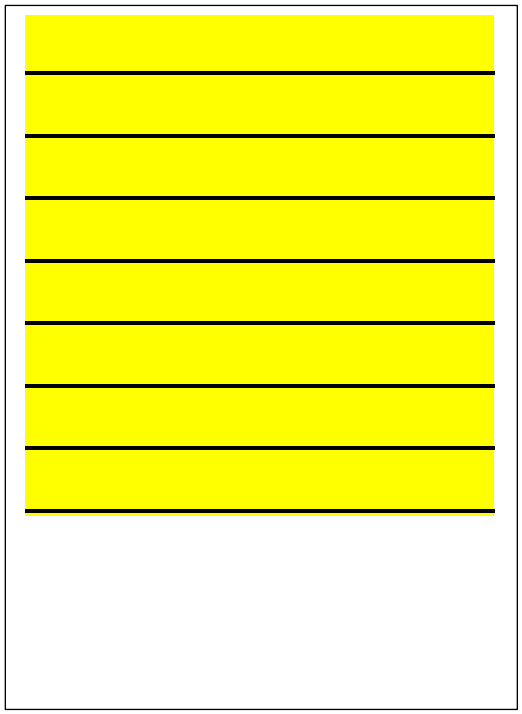
An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c



An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c
+4	d

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e
+4	d

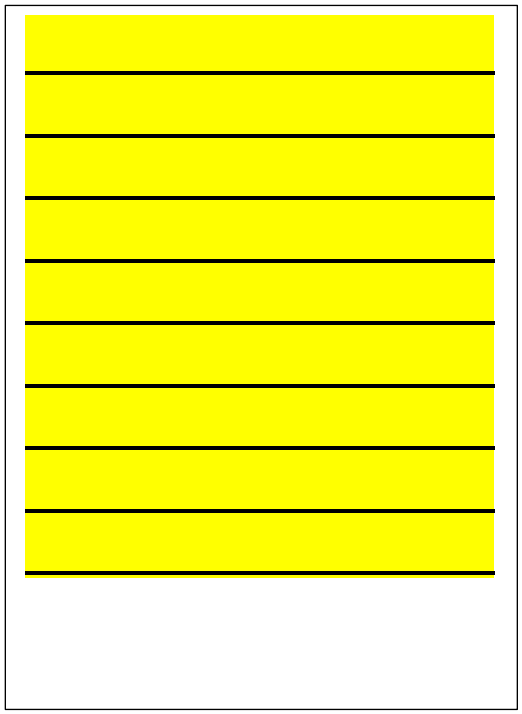
An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e
+4	d, f



An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e
+4	d, f
+5	g

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

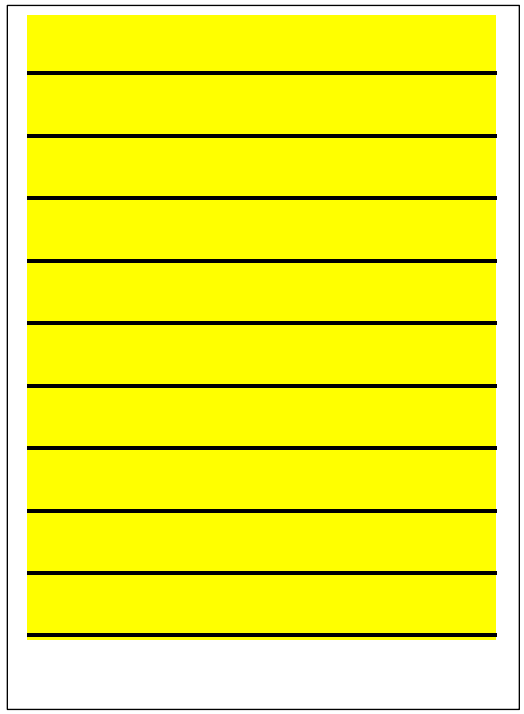
offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e, h
+4	d, f
+5	g

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func



offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e, h
+4	d, f, i
+5	g

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e, h
+4	d, f, i
+5	g, j

An Example: How many locations are allocated for each stackframe of the following function, and what stackframe offset is given to each formal parameter and each local variable declared in the body?

```
int my_func(int x, int[] y, int z)
{
  int a, b[];
  if ( ... ) {
    int c, d[];
       ...
  }
  else {
    int e, f;
       ...
    int g;
       ...
  }
   ...
  int h, i, j, k;
   ...
}
```

ANSWER: 11

Stackframe of any call of my_func

offset	allocated to
-4	x
-3	y
-2	z
-1	return addr
0	dynamic link
+1	a
+2	b
+3	c, e, h
+4	d, f, i
+5	g, j
+6	k

Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

How are stackframes allocated and deallocated during execution of the following sequence of calls and returns?

Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

How are stackframes allocated and deallocated during execution of the following sequence of calls and returns?

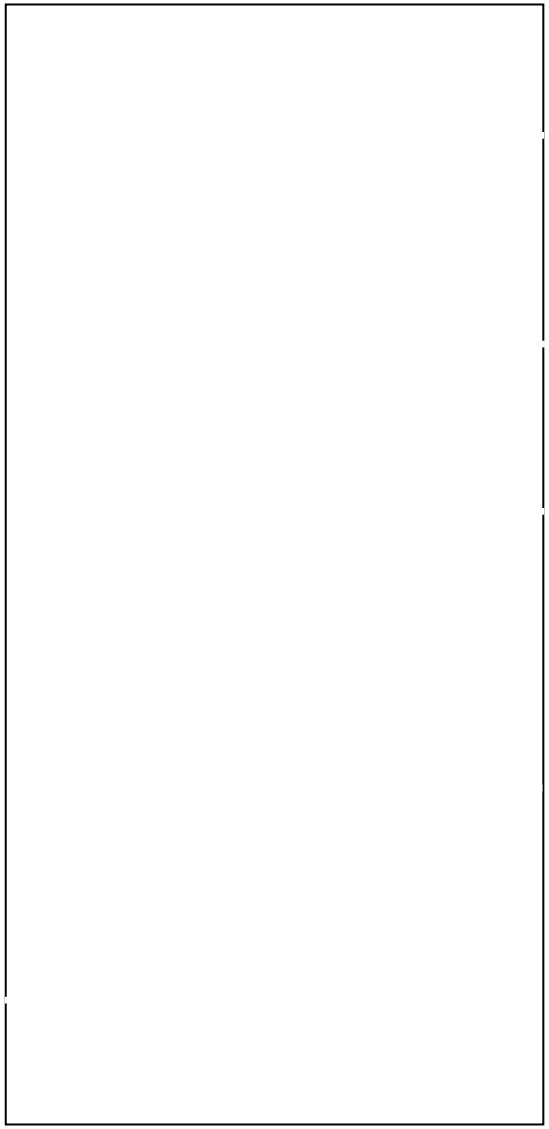
```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory

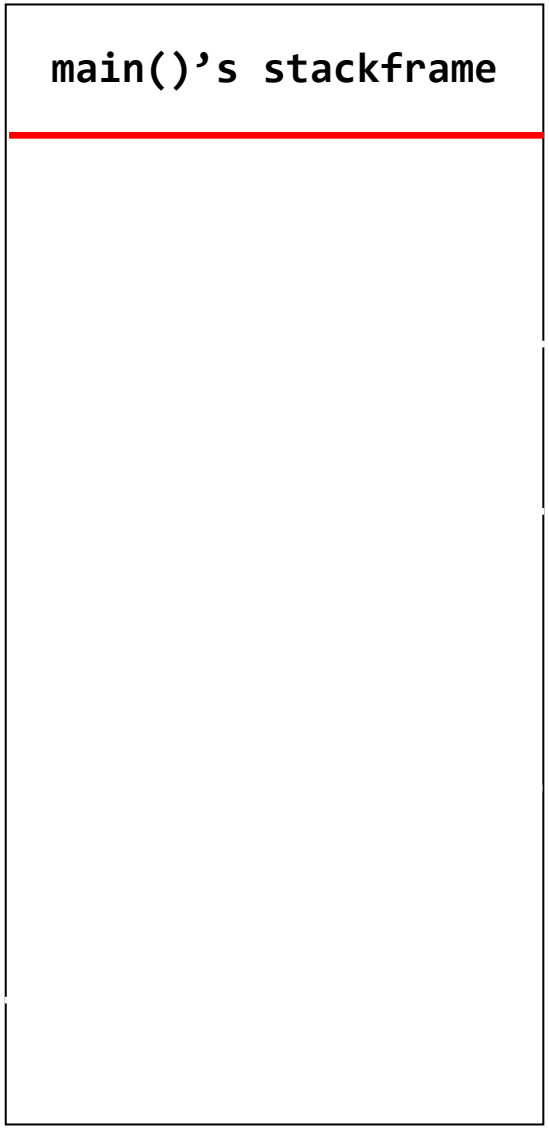


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory

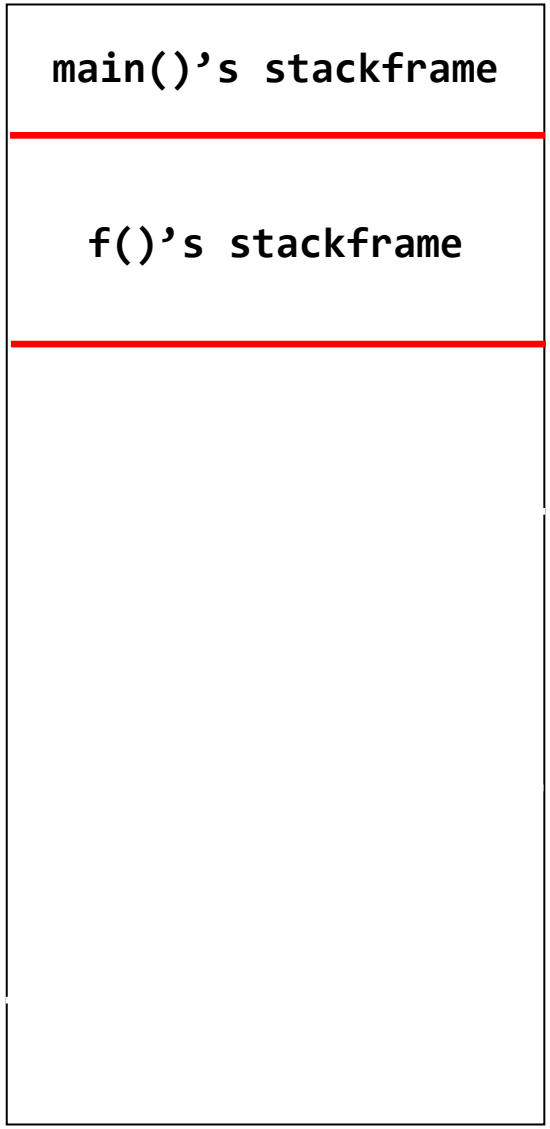


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

(1) main() is called
(2) main() calls f()
(3) f() calls g()
(4) g() calls h()
(5) h() calls f()
(6) f() returns control to h()
(7) h() returns control to g()
(8) g() calls f()

Stack-Dynamically Allocated Data Memory

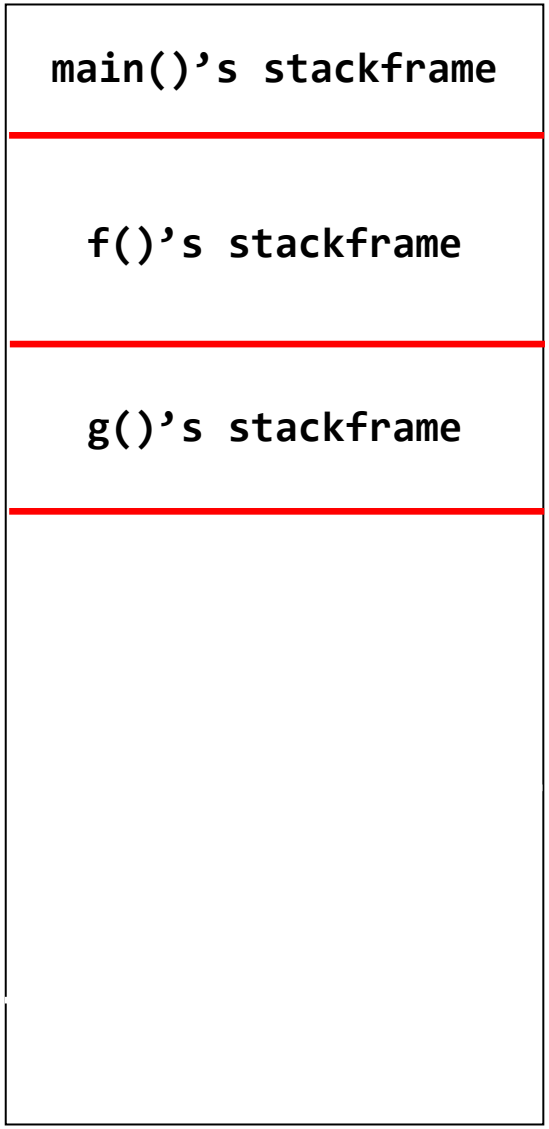


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

(1) main() is called
(2) main() calls f()
(3) f() calls g()
(4) g() calls h()
(5) h() calls f()
(6) f() returns control to h()
(7) h() returns control to g()
(8) g() calls f()

Stack-Dynamically Allocated Data Memory

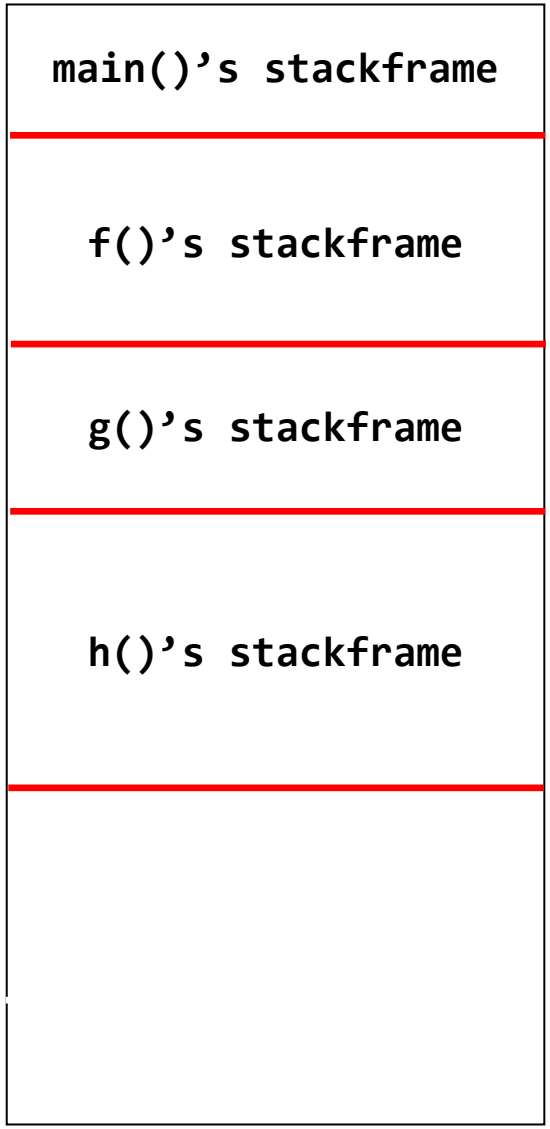


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory

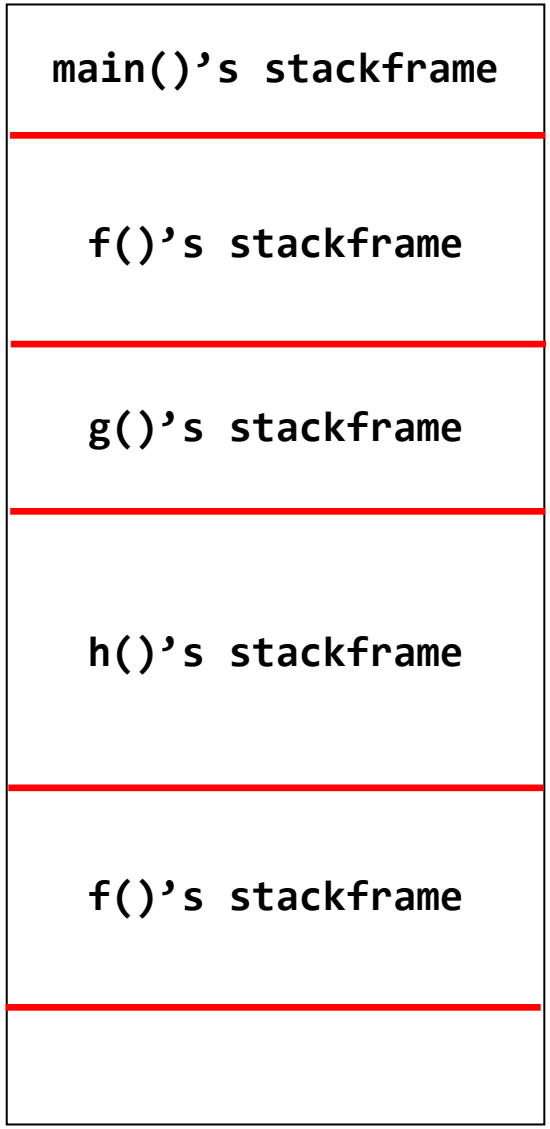


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory

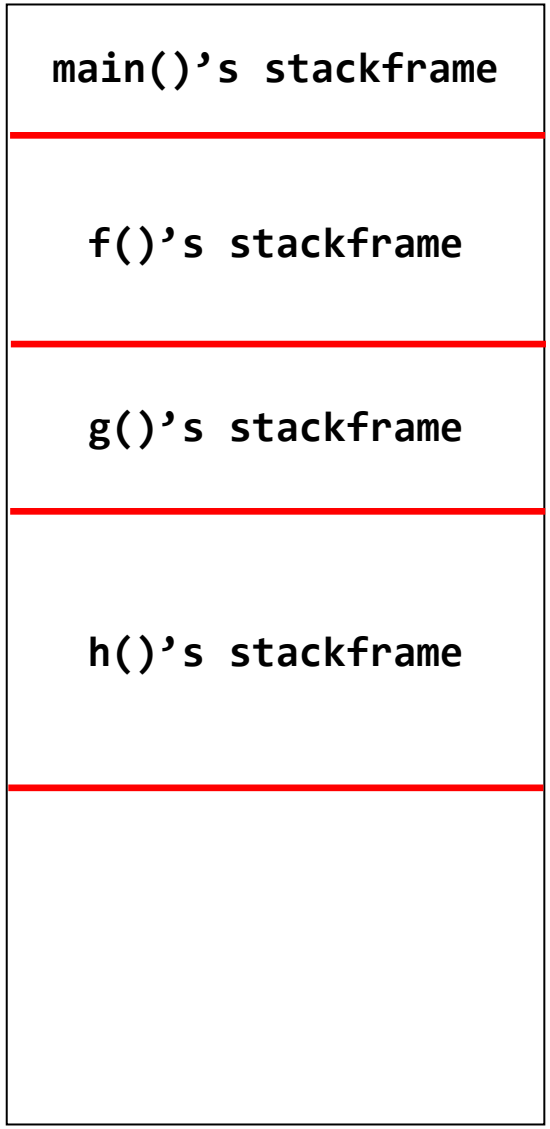


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory

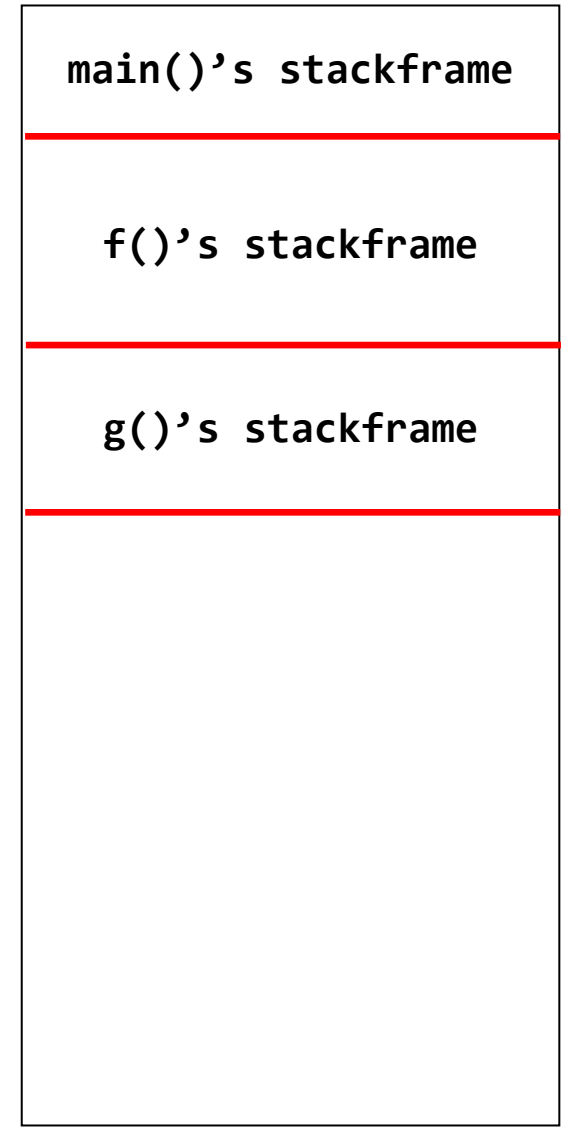


Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory



Example of Stack-Dynamic Allocation

See p. 4 of: https://euclid.cs.qc.cuny.edu/316/Memory-allocation-VM-instruction-set-and-hints-for-asn-2.pdf

```
(1) main() is called
(2)  main() calls f()
(3)   f() calls g()
(4)    g() calls h()
(5)     h() calls f()
(6)      f() returns control to h()
(7)     h() returns control to g()
(8)    g() calls f()
```

Stack-Dynamically Allocated Data Memory