

# Stack allocation

- Memory from the stack:

```
u32b_t* create_array() {  
    u32b_t i, array[64];  
  
    for(i=0; i<64; i++)  
        array[i] = generate_value(i);  
  
    return array;  
}
```

- Fast allocation time.
- Size set at compile time.
- Requires careful attention to the stack.
- Automatic deallocation.

# Heap allocation

- Memory from the heap:

```
u32b_t* create_arary(u32b_t sz) {  
    u32b_t i,*array;  
  
    array = (u32b_t*) malloc(sz*sizeof(u32b_t));  
  
    for(i=0; i<sz; i++)  
        array[i] = generate_value(i);  
  
    return array;  
}
```

- Slower allocation time.
- Size set at run time.
- Requires manual deallocation.

# Malloc and Free

- `void* malloc( u32b_t size );`
- Malloc returns a pointer to the first byte in a block bytes of the requested size; 0 on failure.
- Memory comes from the heap and will not be deallocated automatically like the stack; one must `free()` the allocated memory.
- `void free(void *ptr);`

# Transition

Done with memory allocation.

Moving on to binary operators.

# Shifts

- `>>` right shift, integer divide by two.
- `<<` left shift, integer multiply by two.
- Any bits pushed off the end disappear, and any bits pulled in by the shift are set to 0.

# AND

P	Q	P&Q
0	0	0
0	1	0
1	0	0
1	1	1

- bit-wise AND operation: &

# OR

P	Q	P Q
0	0	0
0	1	1
1	0	1
1	1	1

- bit-wise OR operation: |

# NOT

P	$\sim P$
0	1
1	0

- bit-wise NOT operation:  $\sim$

# XOR

P	Q	$P \oplus Q$
0	0	0
0	1	1
1	0	1
1	1	0

- bit-wise XOR operation:  $\oplus$

# Bit Flags

```
#define F_RUNNING (1<<0) // 1
#define F_PAUSED (1<<1) // 2
#define F_STOPPED (1<<2) // 4
#define F_ERROR (1<<3) // 8
```

```
u32b_t flags;
```

- Set a field:  
`flags |= F_PAUSED;`
- Clear a field:  
`flags &= ~F_PAUSED;`
- Test a field:  
`if( flags & F_PAUSED )`

# Flipping and Swapping

```
u32b_t paused;
```

```
u32b_t x, y;
```

- Flip a flag:

```
paused ^= 1;
```

- Swap values:

```
x ^= y;
```

```
y ^= x;
```

```
x ^= y;
```

- Many more at:

```
http://graphics.stanford.edu/~seander/bithacks.html
```