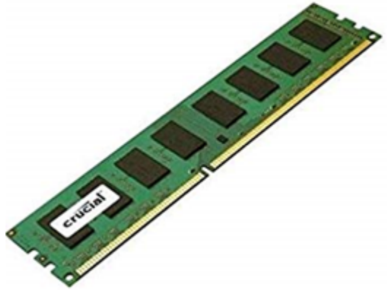
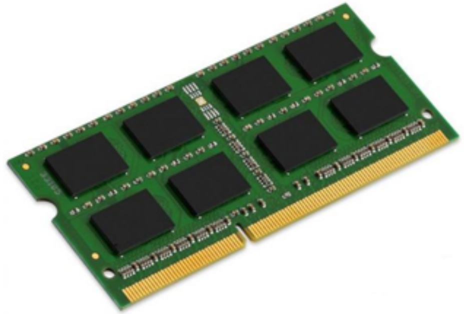
Lecture 2 a

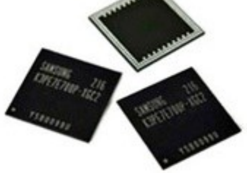
Pointers and memory

Computer’s Memory

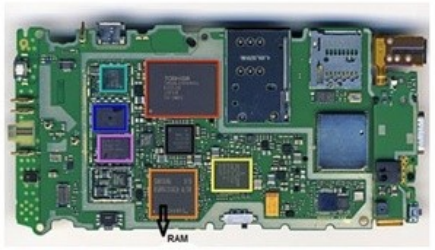




Samsung Mobile
RAM (2 GB)



RAM inside Mobile
Motherboard



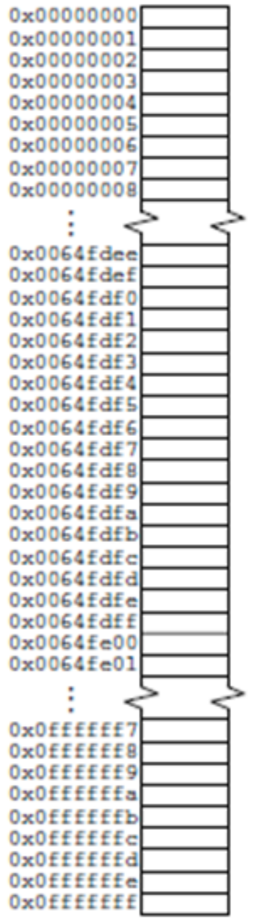
Computer’s Memory

A computer’s memory is a very large array of bytes

So, a 256 MB of RAM actually has an array of 268,435,456 (228) bytes

Ch

Each of these bytes are addressed from 0 to 268,435,455 i.e., 0x00000000 to 0xffffffff



What is a Pointer?

When we execute the following line:

```
int i = 5;
```

Compiler does the following three tasks:

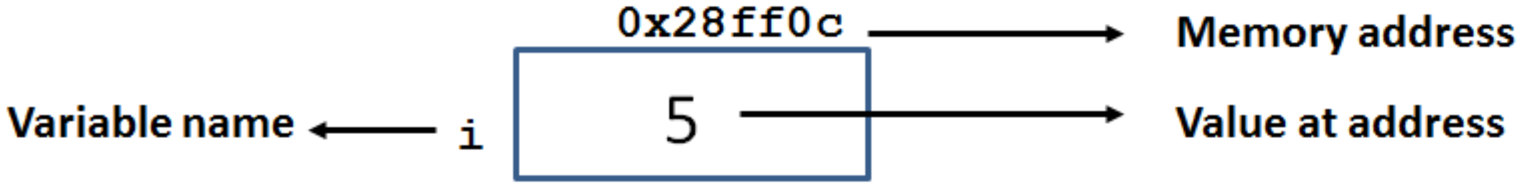
a. Reserve space in memory to hold the integer value.

b. Associate the name i with this memory location.

c. Store the value 3 at this location.

What is a Pointer?

Graphically, this is what happens:



The location number 0x28ff0c is selected by compiler and can’t be relied upon, as the memory address may change if you run the same program for another time.

Note that the location of i is a hexadecimal number

Dereferencing a pointer

```
int x = 12
```

```
int *y = &x
```

0x23ff12

x $\boxed{12}$

0xabcde

y $\boxed{\texttt{0x23ff12}}$

```
cout << x << endl//12

cout<< &x << endl//0x23ff12
```

```
cout << y << endl//0x23ff12

cout << *y << endl//12

cout << &y << endl//0xabcde
```

Pass by value

```
void passByValue(int a){
      a = 596;
}

int main(){
      int x = 3;
      passByValue(x);
      cout << ”x is “ << x << endl;
      return 0;
}
```

Pass by reference

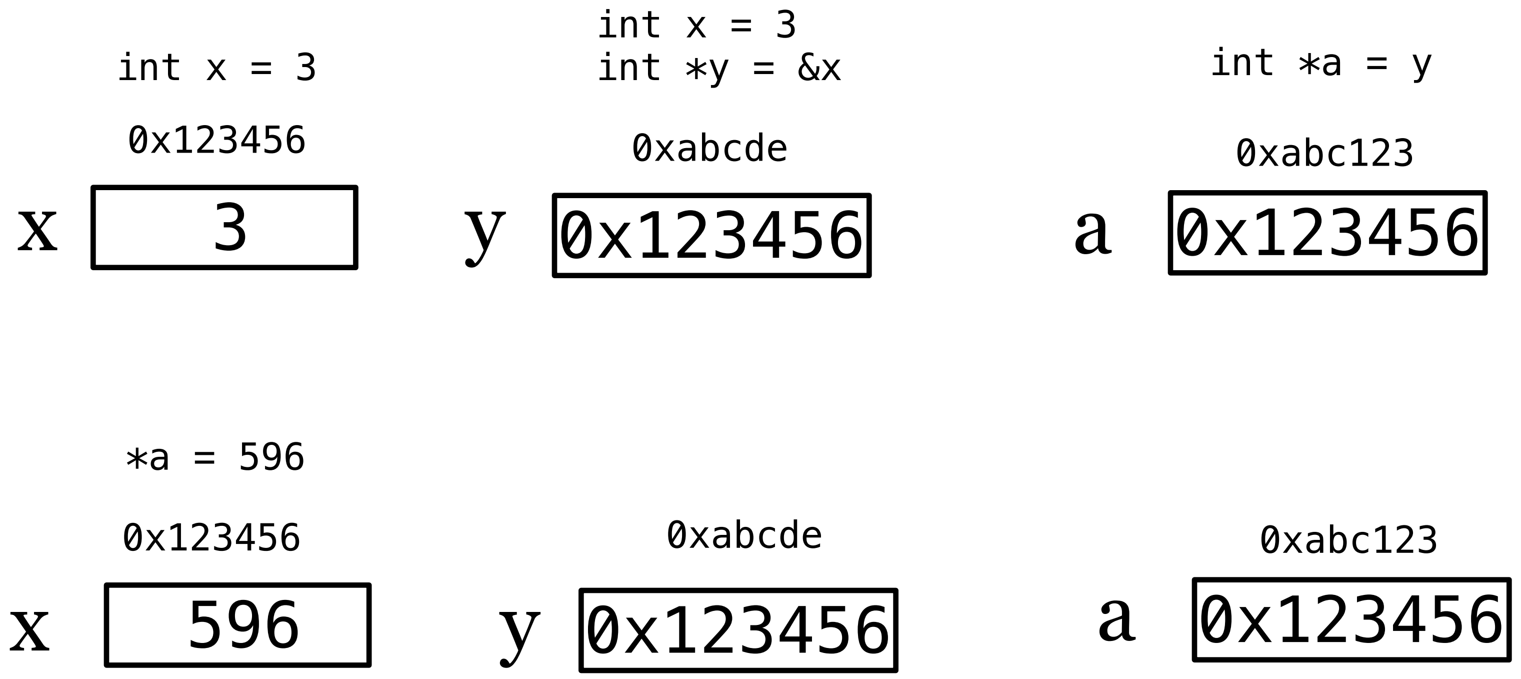
```
void passByReference(int &a){
      a = 596;
}

int main(){
      int x = 3;
      passByReference(x);
      cout << ”x is “ << x << endl;
      return 0;
}
```

Pass by address

```
void passByAddress(int *a){
      *a = 596;
}
int main(){
      int x = 3;
      int *y = &x;
      passByAddress(y);
      cout<<x<<endl;
      return 0;
}
```

Pass by address



```
int x = 3
```

```
int x = 3
int *y = &x
```

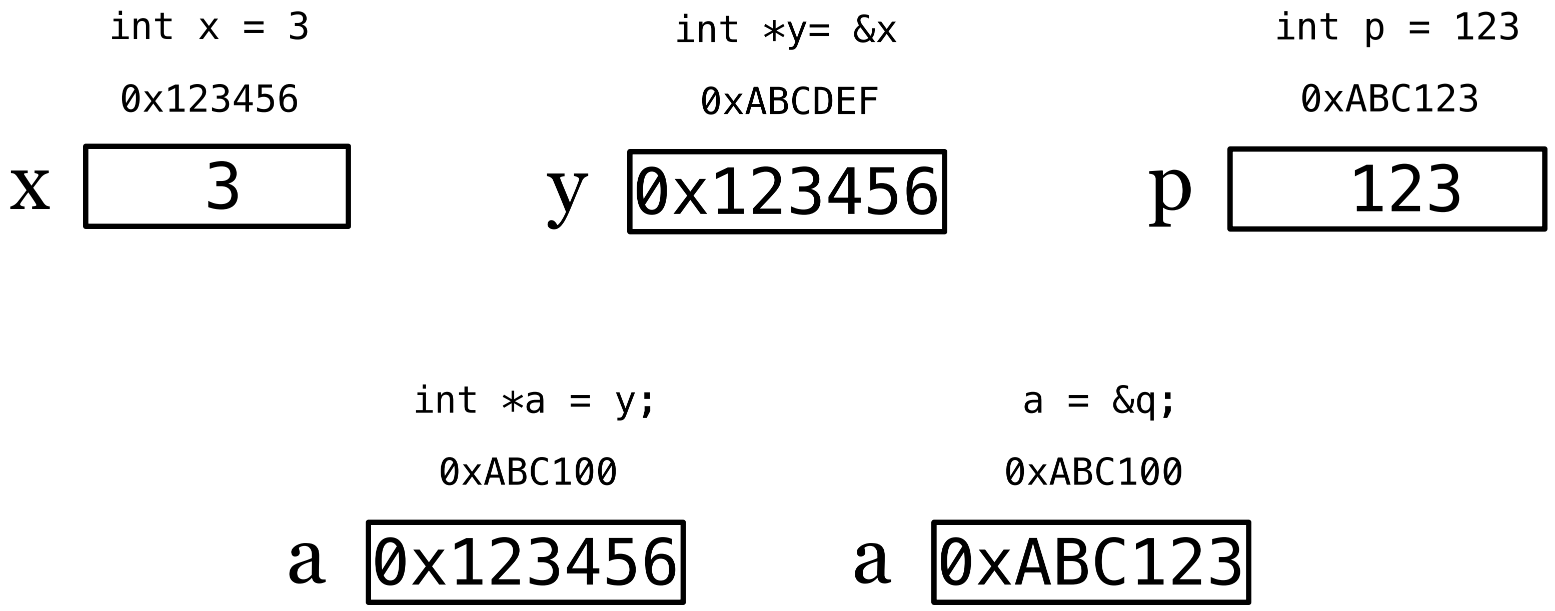
```
int *a = y
```

```
*a = 596
```

Pass pointers by reference(Wrong)

```
void passPointerByReference(int* a, int &q){
       a = &q;
}
int main(){
       int x = 3;
       int *y= &x;
       int p = 123;
       cout << "y is: " << y << " *y is: " << *y << endl;
       passPointerByReference(y,p);
       cout << "y is: " << "y" << " *y is: " << *y <<endl;
       return 0;
}
```

Pass pointers by reference(Wrong)



```
int x = 3
int *y= &x
int p = 123
int *a = y;
a = &q;
```

Pass pointers by reference(Right)

```
void passPointerByReference(int*&a, int &p){
       a = &p;
}
int main(){
       int x = 3;
       int *y= &x;
       int p = 123;
       cout << "y is: " << y << " *y is: " << *y << endl;
       passPointerByReference(y,p);
       cout<< "y is: " << y << " *y is: " << *y << endl;
       return 0;
}
```

Pass pointers by reference(Right)

`int x = 3`

`0x123456`

x $\boxed{\texttt{3}}$

`int *y= &x`

`0xABCDEF`

y $\boxed{\texttt{0x123456}}$

`int p = 123`

`0xABC123`

p $\boxed{\texttt{123}}$

`int* &a = y`

`0xABC100`

a,y $\boxed{\texttt{0x123456}}$

`a = &q;`

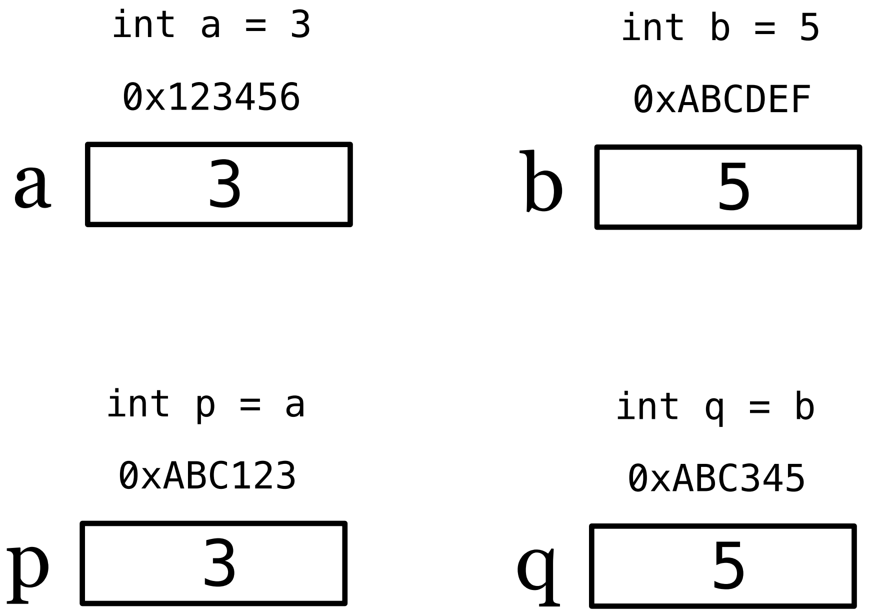
`0xABC100`

a,y $\boxed{\texttt{0xABC123}}$

Swapping values(Wrong way)

```
void mySwap(int p, int q){
        int temp = p;
        p = q;
        q = temp;
}
int main(){
        int a = 3; int b = 5;
        cout << "a: " << a << "b: " << b << endl;
        mySwap(a,b);
        cout<< "a: " << a << "b: " << b <<endl;
        return 0;
}
```

Swapping values(Wrong way)

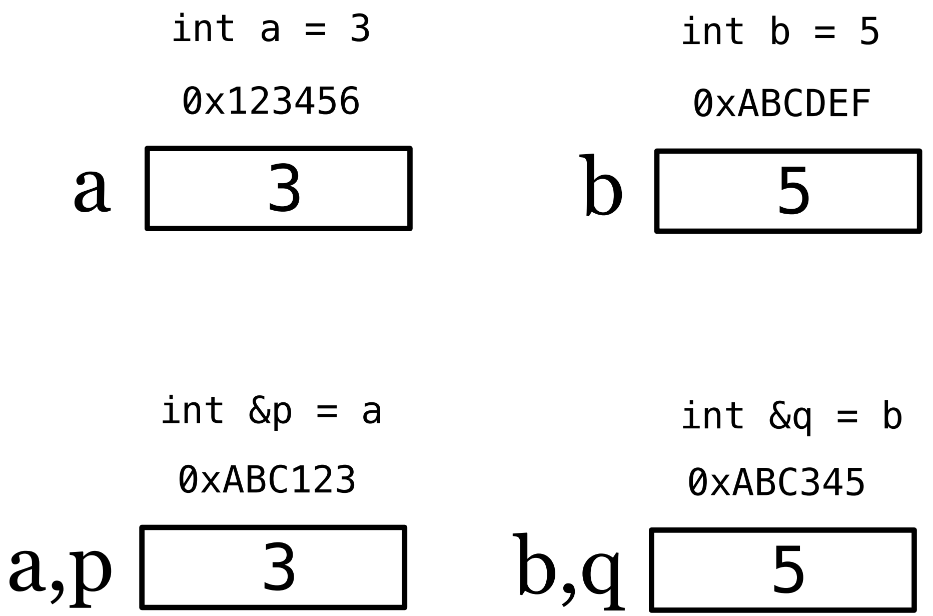


```
int a = 3
int b = 5
int p = a
int q = b
```

Swapping values(Right way)

```
void mySwap(int &p, int &q){
        int temp = p;
        p = q;
        q = temp;
}
int main(){
        int a = 3; int b = 5;
        cout << "a: " << a << " b: " << b <<endl;
        mySwap(a,b);
        cout<< "a: " << a << " b: " << b <<endl;
        return 0;
}
```

Swapping values(Right way)



```
int a = 3
0x123456
```

```
int b = 5
0xABCDEF
```

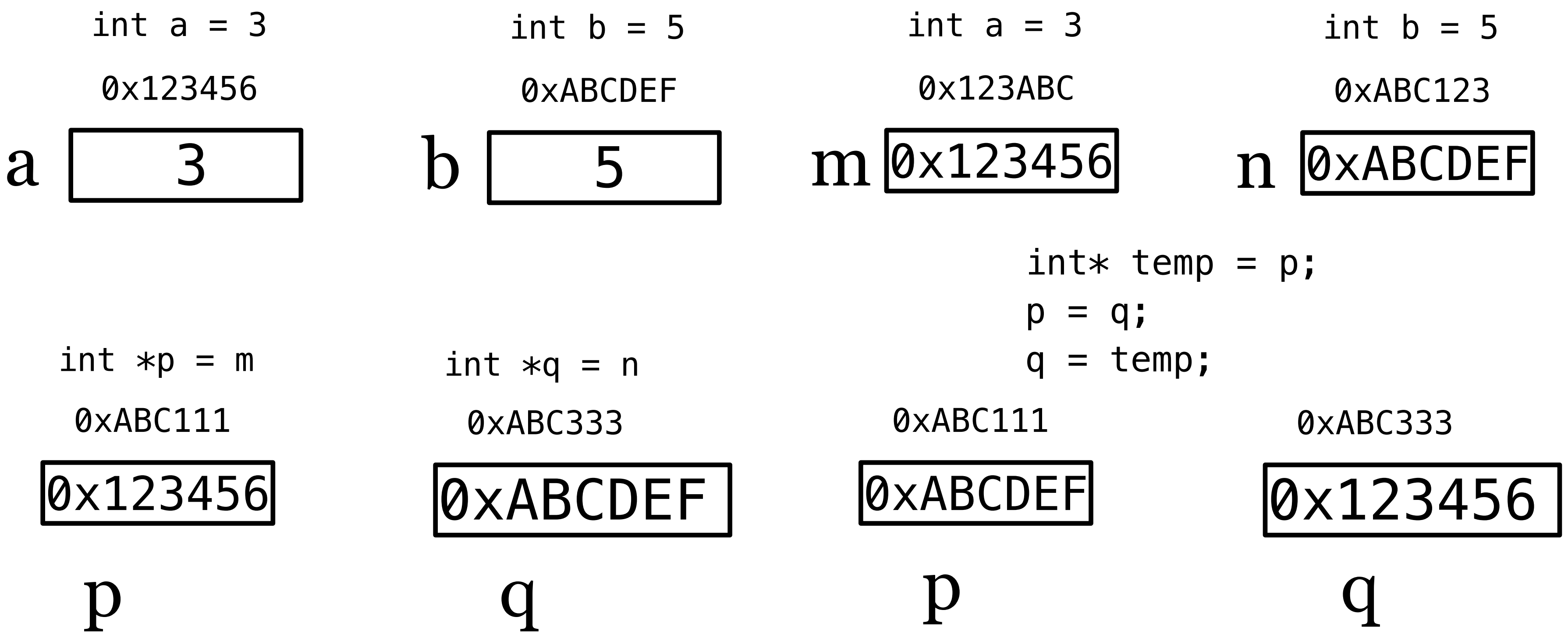
```
int &p = a
0xABC123
```

```
int &q = b
0xABC345
```

Swapping addresses(Wrong way)

```
void mySwap(int* p, int* q){
        int* temp = p;
        p = q;
        q = temp;
}
int main(){
    int a = 3; int b = 5;
    int* m = &a; int* n = &b;
    cout << "m: " << m << "n: " << n << endl;
    mySwap(m,n);
    cout<< "m: " << m << "n: " << n << endl;
    return 0;
}
```

Swapping addresses(Wrong way)

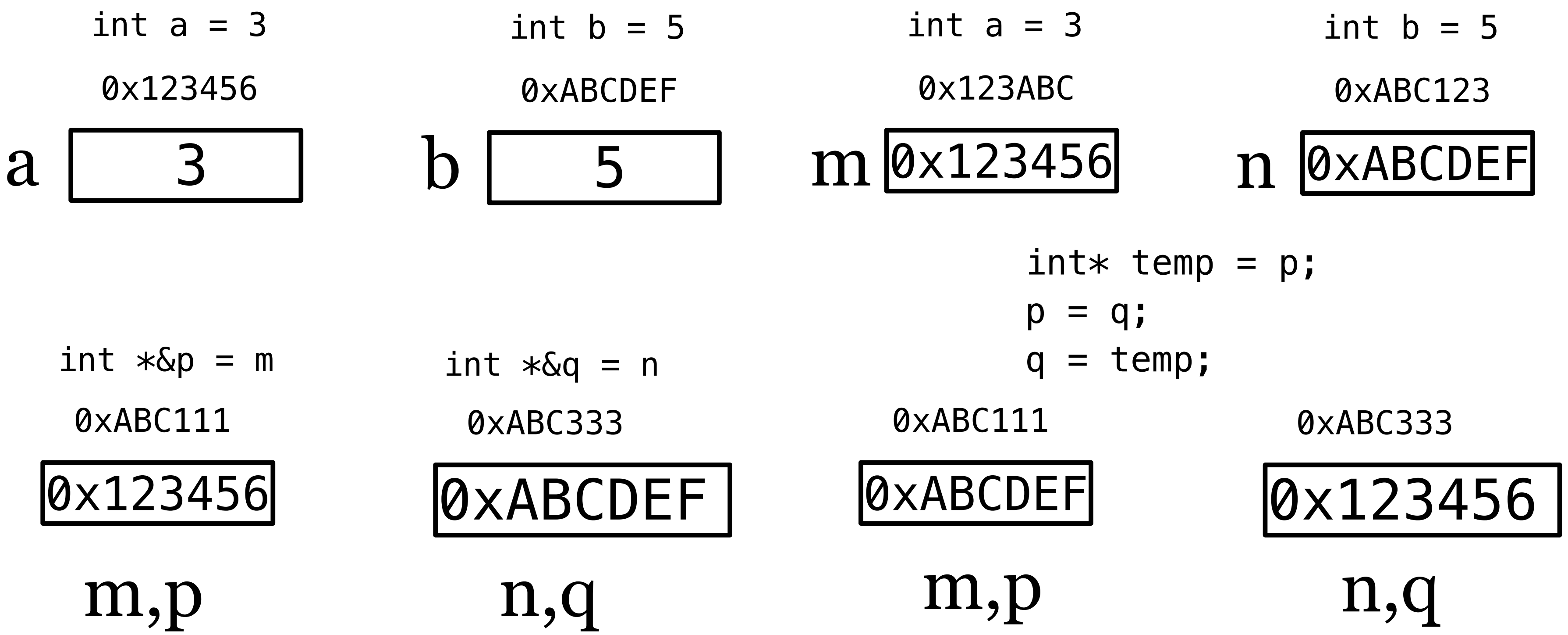


```
int* temp = p;
p = q;
q = temp;
```

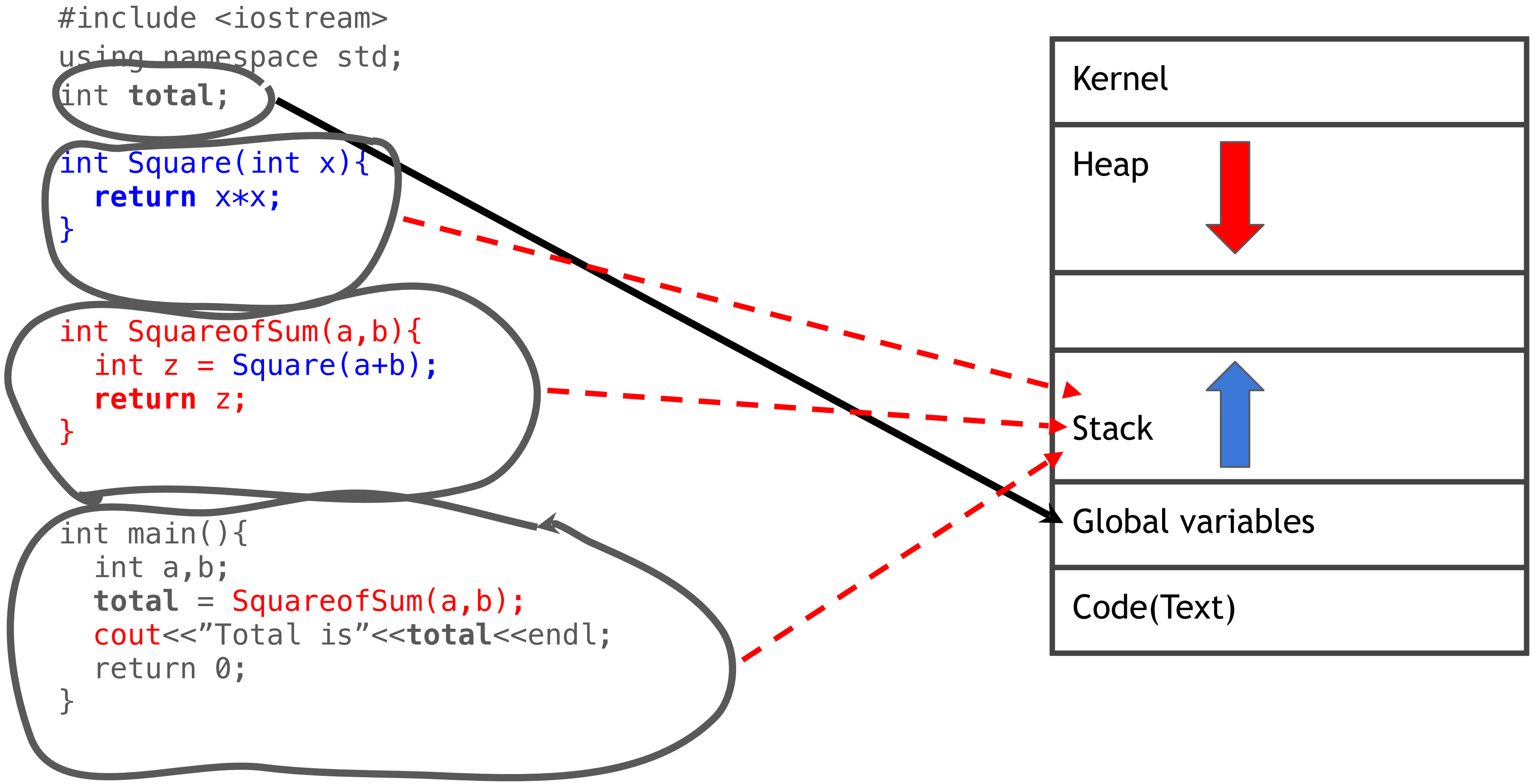
Swapping addresses

```
void mySwap(int* &p, int* &q){
        int* temp = p;
        p = q;
        q = temp;
}
int main(){
    int a = 3; int b = 5;
    int* m = &a; int* n = &b;
    cout << "m: "<< m << "n: " << n << endl;
    mySwap(m,n);
    cout<< "m: "<< m << "n: " << n << endl;
    return 0;
}
```

Swapping addresses(Right way)



```
int* temp = p;
p = q;
q = temp;
```



```
#include <iostream>
using namespace std;
int total;
int Square(int x){
  return x*x;
}
int SquareofSum(a,b){
  int z = Square(a+b);
  return z;
}
int main(){
  int a,b;
  total = SquareofSum(a,b);
  cout<<”Total is”<<total<<endl;
  return 0;
}
```

Limit of Static Memory

```
int main(){
    const int N = 10000000;
    int arr[N];
    for(int i=0; i<N; ++i){
        arr[i] = i;
    }
    cout<<(N*4)<<" B "<<endl;

    return 0;
}
```

Crashes at 4000 KB in linux

Limit of Dynamic Memory

```
int main(){

    int N = 10000000;
    int *arr =new int[N];
    for(int i=0; i<N; ++i){
        arr[i] = i;
    }
    cout<<(N*4)/1000<<" KB "<<endl;
    delete[] arr;
    return 0;
}
```

Static memory 1d array

```
const int N = 100;
int oddNumbers[N];

//Initialization
for (int i=0; i<N; ++i){
        oddNumbers[i] = (2*i + 1);
}
//Display
for (int i=0; i<N; ++i){
        cout<<oddNumbers[i]<<endl;
}
```

Address of elements of array

```
int oddNumbers[3] = {1,3,5};

//Print address of 1st odd number
cout<<&oddNumbers[0]<<endl;
//0x7ffdc1594290

//Print address of 2nd odd number
cout<<&oddNumbers[1]<<endl;
//0x7ffdc1594294

//Print address of 3rd odd number
cout<<&oddNumbers[2]<<endl;
//0x7fff9db88de8
```

Pointer arithmetic Static Memory

```
int oddNumbers[3] = {1,3,5};
int *p = oddNumber;
p = &oddNumbers[0];
cout<<*p<<endl;
cout<<p<<endl;

p++;
cout<<*p<<endl;
cout<<p<<endl;

p++;// Try cout<<(*p)++<<endl;
cout<<*p<<endl;
cout<<p<<endl;
```

Pointer arithmetic Dynamic Memory

```
int *oddNumbers = new int[4]{1,3,5};

cout<<*oddNumbers<<endl;
cout<<oddNumbers<<endl;

oddNumbers++;
cout<<*oddNumbers<<endl;
cout<<oddNumbers<<endl;

oddNumbers++;
// Try cout<<(*oddNumbers)++<<endl;
cout<<*oddNumbers<<endl;
cout<<oddNumbers<<endl;
```

Dynamic memory 1d array

```
int num;
cout << ”Please enter a number: ”;
cin >> num;

int *evenNumber = new int[num];

for(int i = 0; i< num; ++i){

          evenNumber[i] = 2*i;

}


for(int i = 0; i< num; ++i){

          cout<<evenNumber[i]<<" ";

}

delete[] evenNumber;
```

Dynamic memory 1d array

```
for (int n=0; n<num; n++) {
    *(evenNumber+n) = (2*n+2);
    //evenNumber[n] = (2*n+2);
}

for (int n=0; n<num; n++) {
    cout<<*(evenNumber+n) << ", ";
    //cout<<evenNumber[n];
}
delete[] evenNumber;
```