function with same name but different signature

~~void add(int,int);~~

void add(double,double);

void add(int,double);

void add(int,int,int);

int add(double,int);

void add(int);

int add();

If function parameters have default value we call it as
Default Argument Function

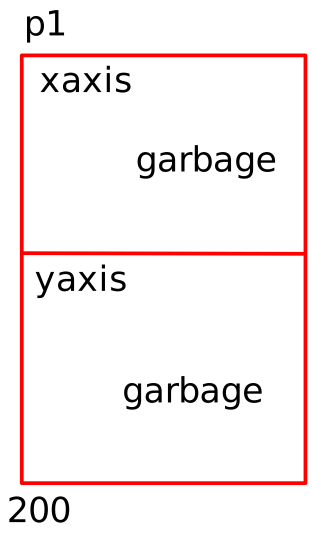
```
void add(int n1 ,int n2=0,int n3=0){
}
```

Types of member functions
1. Constructor
2. destructor
3. Mutators
4. Inspectors
5. Facilitators

Stack

main(){
p1
}

p1

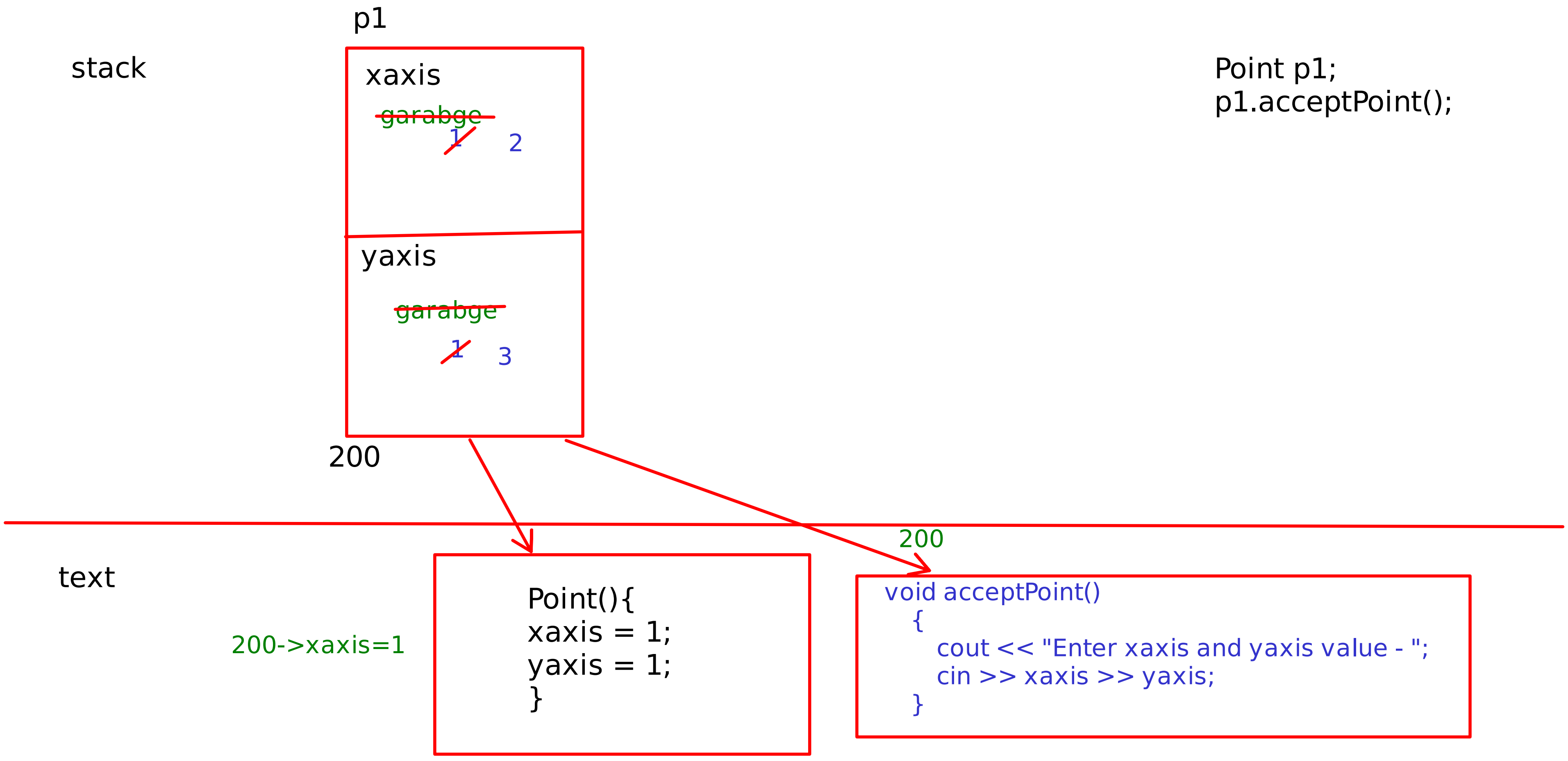


Constructor is special member function of a class.
1. name od ctor is same as that of class name
2. It do not have any return type
3. it gets automaically called when object is created

Types of Constructor
1. Parameterless constructor
2. Parameterized constructor
3. default constructor

```
class Empty{

Empty(){

}
}

int main(){
Empty e1;
}
```

default Constructor
. It will not initialie your class data members
.
.
.



```
Point(){
xaxis = 1;
yaxis = 1;
}
```

```
void acceptPoint()
   {
      cout << "Enter xaxis and yaxis value - ";
      cin >> xaxis >> yaxis;
   }
```

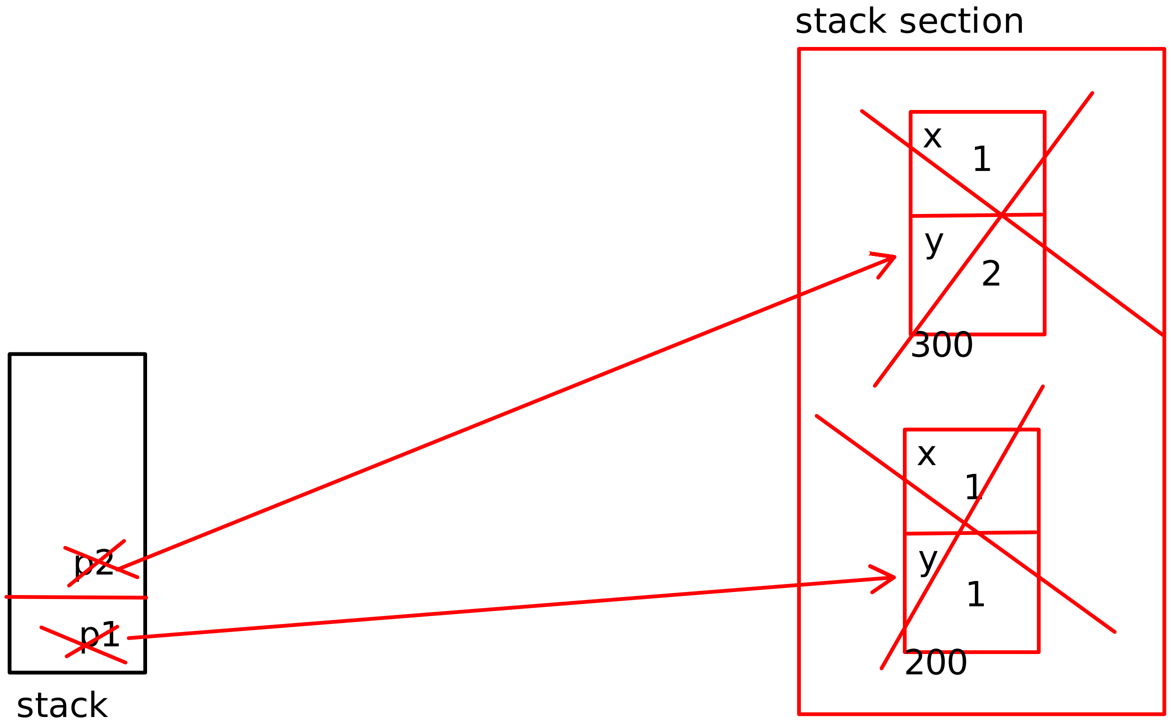
Point p1;
p1.acceptPoint();

```
          2              3
Point(int xaxis, int yaxis)
 {
   xaxis = xaxis; 2
   yaxis = yaxis; 3
 }
```

```
                2          3
Point(int xaxis, int yaxis)
 {
   this->xaxis = xaxis;        200->xaxis = 2;
   this->yaxis = yaxis;        200->yaxis = 3;
 }
```

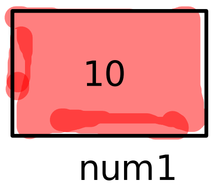
```
vod f1(){
Point p1;

Point p2(1,2);

}


void f2(){

Point P3(2,3);
Point p4(4,5);

}
```

```
main(){
f1();
f2();
}
```

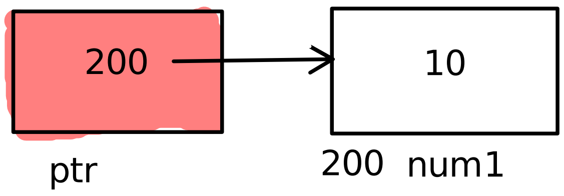


Constant

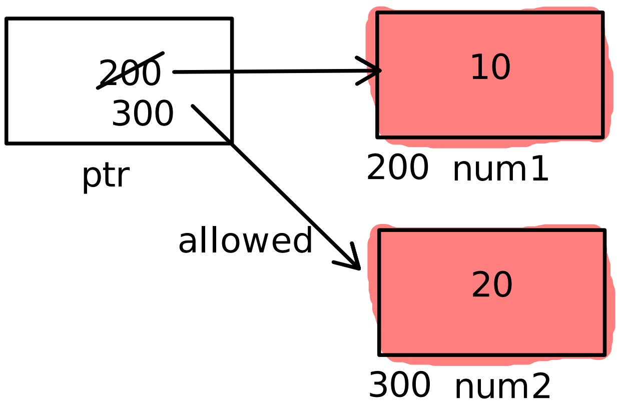
```
// const int num1 = 10;

int const num1 = 10;
```



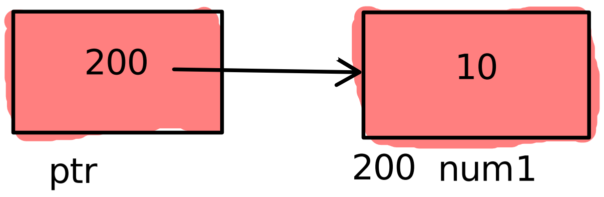
```
int num1 = 10;
int num2 = 20;
int *const ptr =&num1;

ptr = &num2; // Not Allowed
```



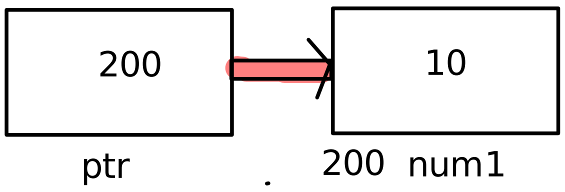
```
const int num1 = 10;
const int num2 =  20;
const int *ptr = &num1;
*ptr = 100; // Not Allowed
ptr = &num2; // OK
```



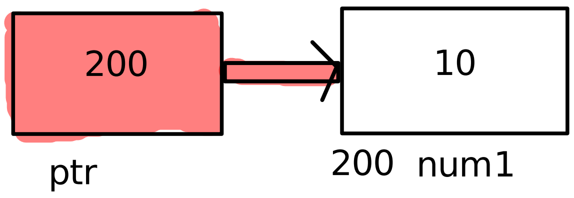
```
const int num1 = 10;
const int num2 = 20;
const int *const ptr = & num1;
*ptr = 100; //Not Allowed
ptr = &num2; //Not Allowed
```



```
int num1 = 10;
int num2 = 20;

const int *ptr = &num1;
```



```
int num1= 10;
const int *const ptr = &num1;
```



```
clasname *const this;
```

```
void acceptTime()// Time *const this = 200
{
       Time t;
       this = &t;// NOT ALLOWED

}

  Time t1;
  t1.acceptTime(); // 200-> this
```