PIC 10A 2020 W

Surface $x^{\frac{2}{3}} + y^{\frac{2}{3}} + z^{\frac{2}{3}} = 1$

Normal vector to the surface

$$\left\langle \tfrac{2}{3}x^{-\frac{1}{3}}, \tfrac{2}{3}y^{-\frac{1}{3}}, \tfrac{2}{3}z^{-\frac{1}{3}} \right\rangle$$

Planes tagent to the surface

$$\tfrac{2}{3}x_0^{-\frac{1}{3}}(x-x_0) + \tfrac{2}{3}y_0^{-\frac{1}{3}}(y-y_0) + \tfrac{2}{3}z_0^{-\frac{1}{3}}(z-z_0) = 0$$

$$\Rightarrow \quad \tfrac{2}{3}x_0^{-\frac{1}{3}}x - \tfrac{2}{3}x_0^{\frac{2}{3}} + \tfrac{2}{3}y_0^{-\frac{1}{3}}y - \tfrac{2}{3}y_0^{\frac{2}{3}} + \tfrac{2}{3}z_0^{-\frac{1}{3}}z - \tfrac{2}{3}z_0^{\frac{2}{3}} = 0$$

$$x_0^{-\frac{1}{3}}x + y_0^{-\frac{1}{3}}y + z_0^{-\frac{1}{3}}z = x_0^{\frac{2}{3}} + y_0^{\frac{2}{3}} + z_0^{\frac{2}{3}} = 1$$

$$\downarrow$$

$$x_0^{-\frac{1}{3}}x + y_0^{-\frac{1}{3}}y + z_0^{-\frac{1}{3}}z = 1$$

x-intercept: $y=0 \quad z=0$

$$x_0^{-\frac{1}{3}}x = 1 \quad \Rightarrow \quad x = x_0^{\frac{1}{3}}$$

y-intercept: $x=0 \quad z=0$

$$y_0^{-\frac{1}{3}}y = 1 \quad \Rightarrow \quad y = y_0^{\frac{1}{3}}$$

z-intercept: $x=0 \quad y=0$

$$z_0^{-\frac{1}{3}}z = 1 \quad \Rightarrow \quad z = z_0^{\frac{1}{3}}$$

The sum of squares:

$$\left(x_0^{\frac{1}{3}}\right)^2 + \left(y_0^{\frac{1}{3}}\right)^2 + \left(z_0^{\frac{1}{3}}\right)^2 = 1$$

$$!(a \,||\, (b \,\&\&\, !a) \,\&\&\, !c)$$

$$= !(a \,||\, ((b \,\&\&\, !a) \,\&\&\, !c))$$

$$= !(T \,||\, ((T \,\&\&\, F) \,\&\&\, F))$$

$$= !(T \,||\, (F \,\&\&\, F))$$

$$= !(T \,||\, F)$$

$$= F$$

$$!(a \,||\, ((b \,\&\&\, !a) \,\&\&\, !c))$$

$$= !a \,\&\&\, !((b \,\&\&\, !a) \,\&\&\, !c)$$

$$= !a \,\&\&\, (!(b \,\&\&\, !a) \,||\, c)$$

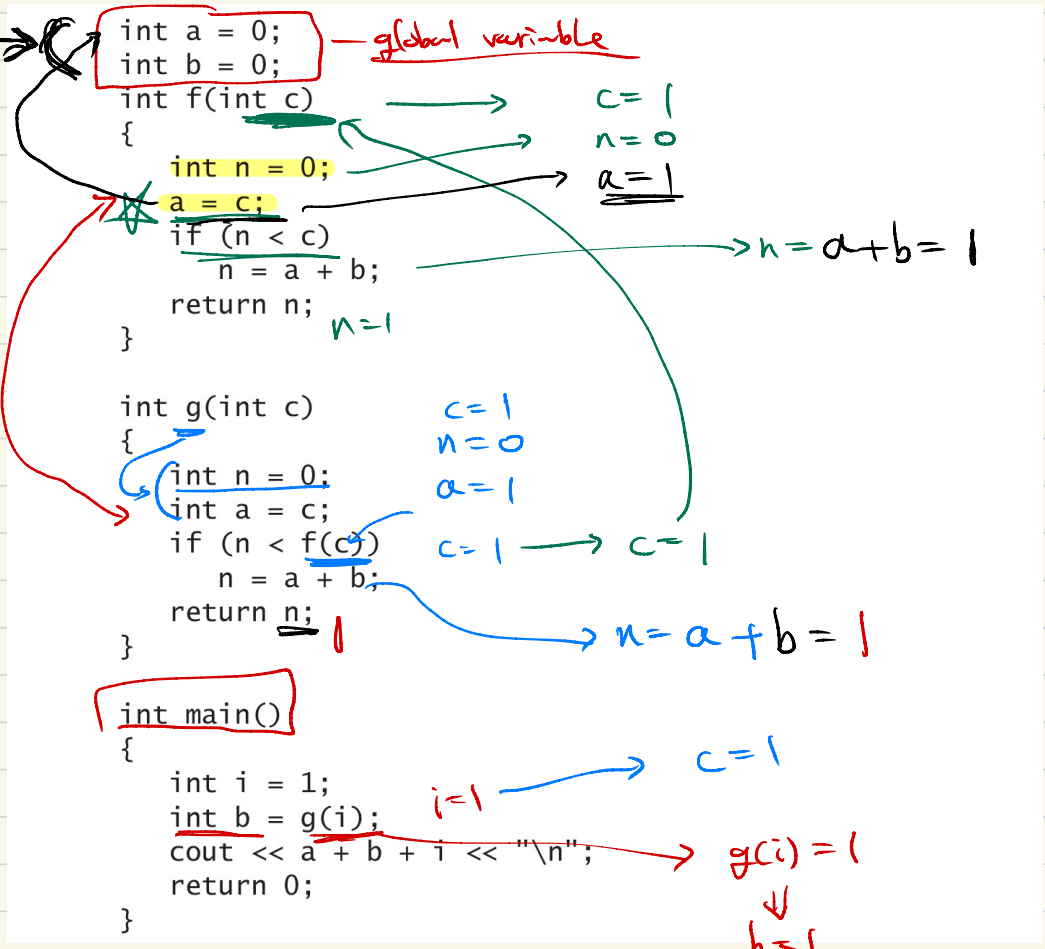
$$= !a \,\&\&\, ((!b \,||\, a) \,\&\&\, !c)$$

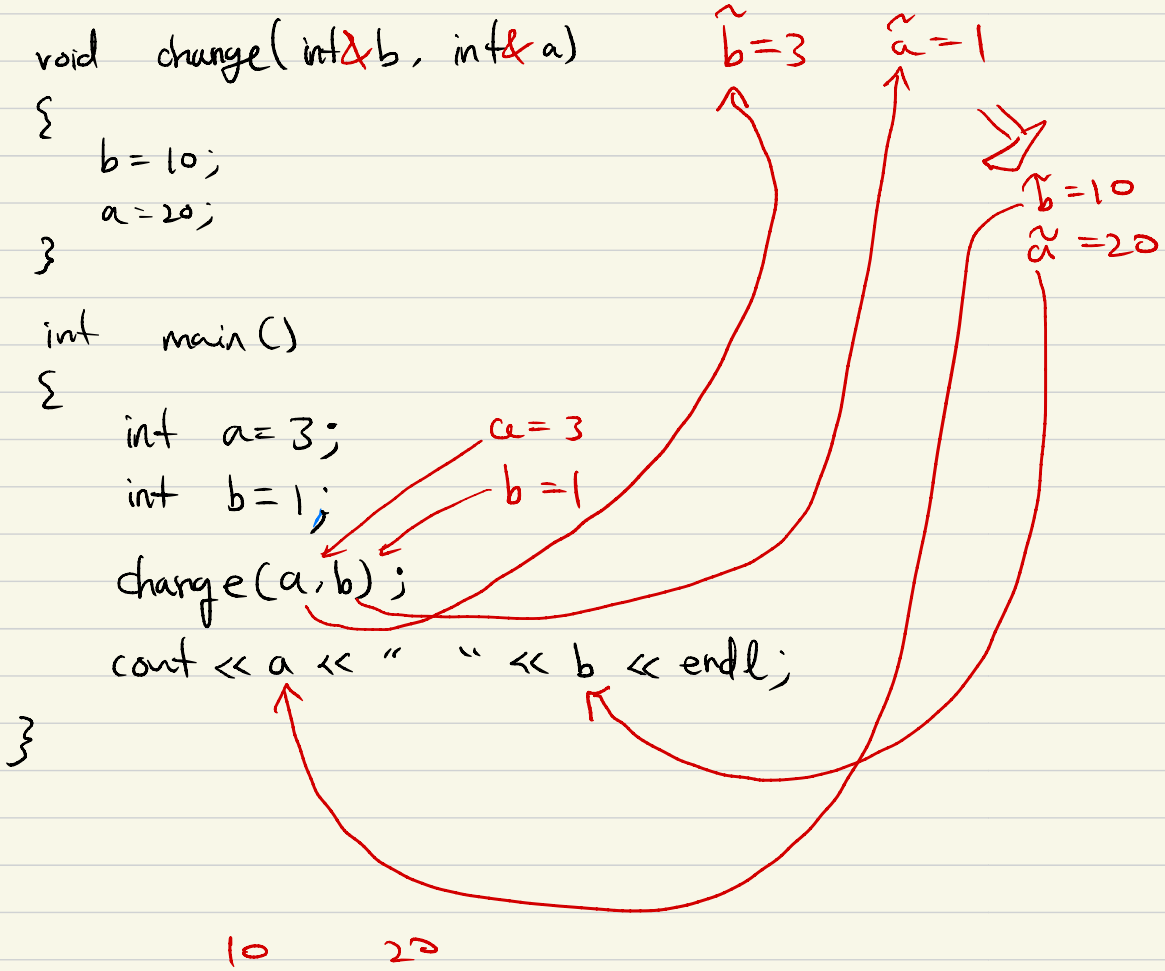
$$= F \,\&\&\, ((F \,||\, T) \,\&\&\, F)$$

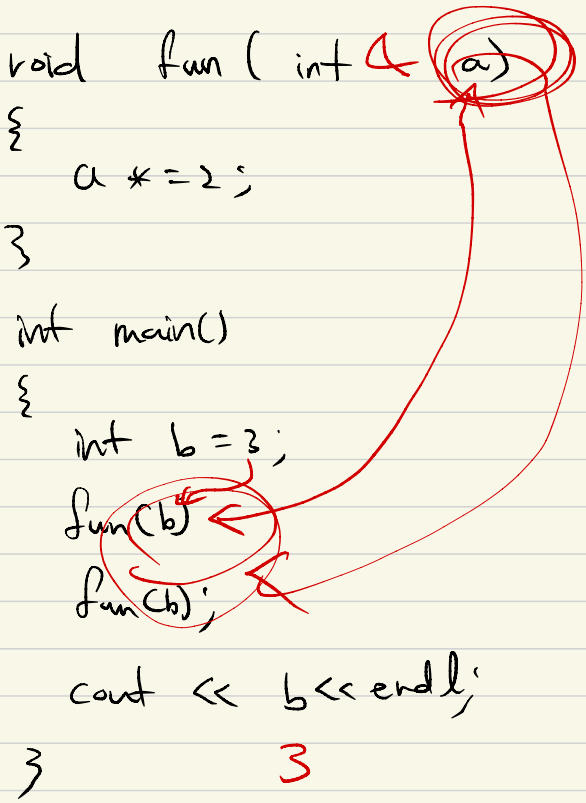
$$= F \,\&\&\, (T \,\&\&\, F)$$

$$= F \,\&\&\, F$$

$$= F$$


int a = 0;
int b = 0;
global variable
int f(int c)
{
int n = 0;
a = c;
if (n < c)
n = a + b;
return n;
}
int g(int c)
{
int n = 0;
int a = c;
if (n < f(c))
n = a + b;
return n;
}
int main()
{
int i = 1;
int b = g(i);
cout << a + b + i << "\n";
return 0;
}
c = 1
n = 0
a = 1
n = a + b = 1
n = 1
c = 1
n = 0
a = 1
c = 1
c = 1
n = a + b = 1
1
c = 1
i = 1
g(i) = 1
b = 1
a + b + i = 1 + 1 + 1 = 3


void change(int& b, int& a)
{
b = 10;
a = 20;
}
int main()
{
int a = 3;
int b = 1;
change(a, b);
cout << a << " " << b << endl;
}
b = 3
a = 1
b = 10
a = 20
a = 3
b = 1
10
20



```
void fun ( int a)
{
   a *= 2;
}
int main()
{
   int b = 3;
   fun(b);
   fun(b);
   cout << b<<endl;
}
```

3

```
void swap ( int& a , int
```

Exercise P4.2. Write a procedure `void sort2(int& a, int& b)` that swaps the values of `a` and `b` if `a` is greater than `b` and otherwise leaves `a` and `b` unchanged. For example,

```
int u = 2;
int v = 3;
int w = 4;
int x = 1;
sort2(u, v); // u is still 2, v is still 3
sort2(w, x); // w is now 1, x is now 4
```

```
void sort2(int& a, int& b)
{
   //TODO
   if(a > b)
   {
      int temp = a;
      a = b;
      b = temp;
   }
}
void sort3(int& a, int& b, int& c)
{
   //TODO
   sort2(b,c);
   sort2(a,c);
   sort2(a,b);
}

int main()
{
   int u=3; int w = 1;
   int v=2;
   cout << u << v << w << endl;
   sort3(u,v,w);
   cout << u << v << w << endl;
```

a=3, b=2, c=1

b=2 c=1 ⟹ b=1 c=2

a=3 c=2 ⟹ a=2 c=3

a=2 b=1 ⟹ a=1 b=2

a=1 b=2 c=3

3 2 1

1 2 3

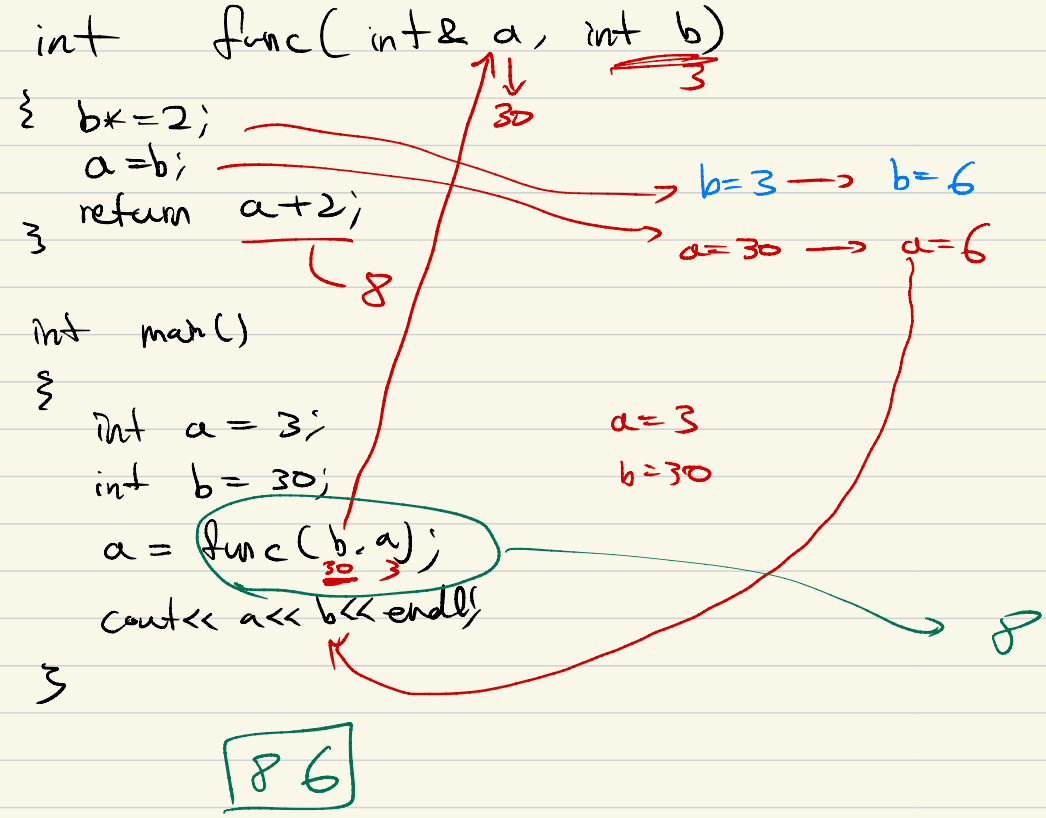
```
void sorts( int& a, int& b, int& c)
{
    int tema=a;
    int temb=b;
    int temc=c;

    if( a>b && b>c)
    {
        a=tema;
        b=temb;
        c=temc;
    }
    else if(a>c && c>b)
    {
        a=tema;
        b=temc;
        c=temb;
    }
    else if(b>a && a>c)
    {
        a=temb;
        b=tema;
        c=temc;
```

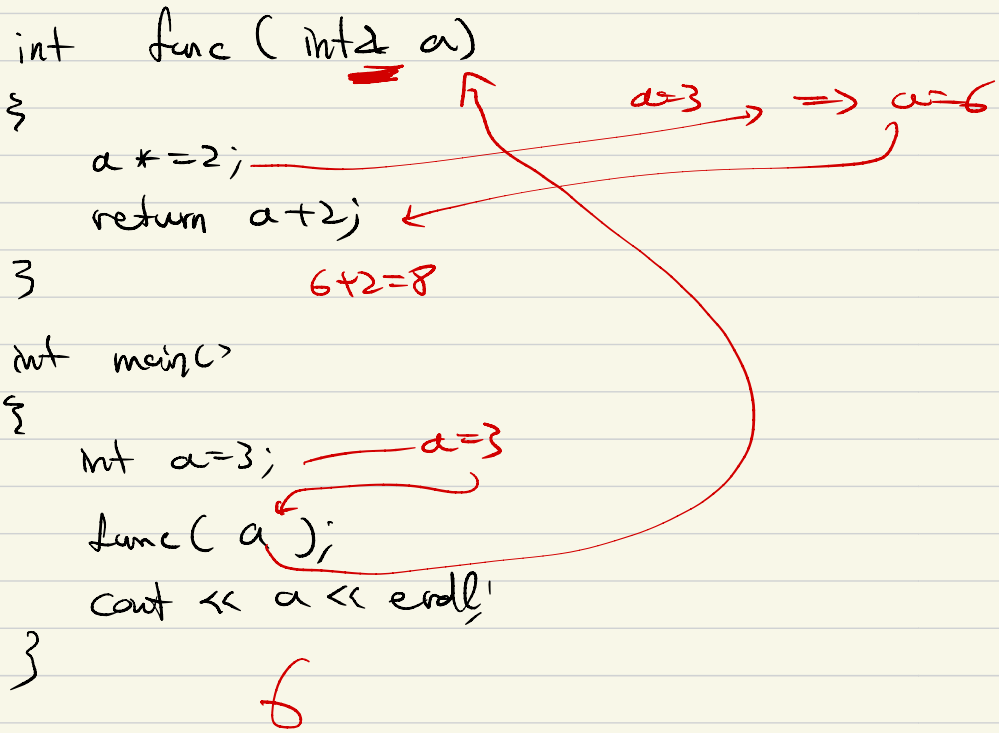
```
bool leap_year(int year)
{
    return (year - 2000) % 4 == 0
}
```

year=2000 → 0 % 4 = 0 ⇒ true

year=901 → (901-2000)%4 = -(1099)%4 =/= 0 ⇒ false

```
int main()
{
    cout << leap_year(2000) << endl;   // 1
    cout << leap_year(2009) << endl;   // 0
```


int func(int& a, int b)
{ b*=2;
a=b;
return a+2;
}
int main()
{
int a = 3;
int b = 30;
a = func(b, a);
cout<< a<< b<<endl;
}
30
3
b=3 → b=6
a=30 → a=6
a=3
b=30
8
8 6


int func (int& a)
{
a*=2;
return a+2;
}
int main()
{
int a=3;
func(a);
cout << a << endl;
}
a=3 => a=6
6+2=8
a=3
6

Exercise P5.1. Implement all member functions of the following class:

```
class Person
{
public:
   Person();
   Person(string pname, int page);
   void get_name() const;
   void get_age() const;
private:
   string name;
   int age; // 0 if unknown
};
```

constructors

member functions

Accessor

```
Person::Person()
{
   name = "John";
   age = 0;
}
Person::Person(String pname, int page)
{
   name = pname;
   age = page;
}
void Person::get_name() const
{
   cout << "name : " << name << endl;
}
void Person::get_age() const
{
   cout << "age : " << age << endl;
}
```

Default constructor

constructor

accessor

```
#include <iostream>
#include <string>

using namespace std;

class PEmployee

PEmployee :: PEmployee()
{
    name = "John";
    salary = 0;
}
PEmployee :: PEmployee(string employee_name,
                       double initial_salary)
{
    name = employee_name;
    salary = initial_salary;
}
double PEmployee :: get_salary() const
{
    return salary;
}
string PEmployee :: get_name() const
{
    return name;
}
```

```
void PEmployee::set_salary(double new_salary)
{
    salary = new_salary;
}
```

```
class Account
{
public:
    Account();
    Account(int pbalance);

    void add(double money);
    void withdraw(double money);

    double get_money() const;
private:
    double balance;
};
```

constructors

mutators

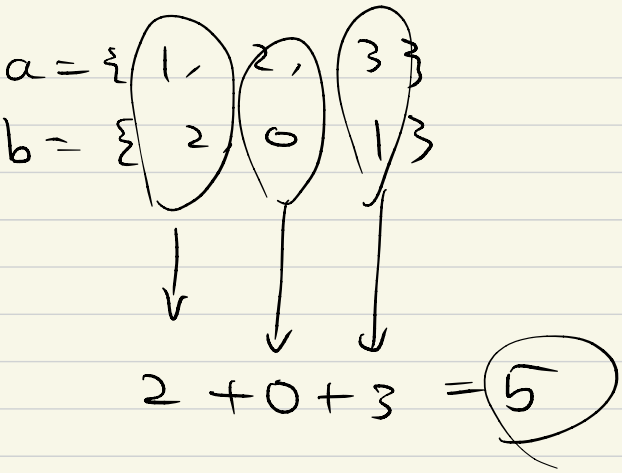
accessor, return (double)

```
#include <iostream>
#include <vector>

using namespace std;

int main()
{
    vector<int> v1;  {}        Goal: v1 = {1, 2, ..., 100}

    for (int i = 1; i <= 100; ++i)
    {
        v1.push_back(i);
    }

    cout << v1.size() << endl; //prints out 100

    for (int i=0; i < v1.size(); ++i)
    {
        cout << v1[i] << endl;
    }
```

v1 = { 1, 2, 3, 4, 5, ... , 100}

v1[0] v1[1]

1
2
3
4
:
99
100




```
#include <iostream>
#include<vector>
using namespace std;
double fun(vector<double> a, vector<double> b)
{   double s=0;
    for(int i=0; i<a.size(); ++i)
    {
        s += a[i]*b[i];
    }
    return s;
}
int main()
{
    vector<double> v1;
    vector<double> v2;
    v1.push_back(1);
    v1.push_back(2);
    v2.push_back(0);
    v2.push_back(3);
    cout << fun(v1, v2) <<endl;
}
```

a[i] b[i]

i=0 ✓ 1 * 0 = 0

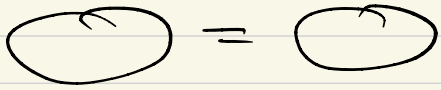
i=1 ✓ 2 * 3 = 6

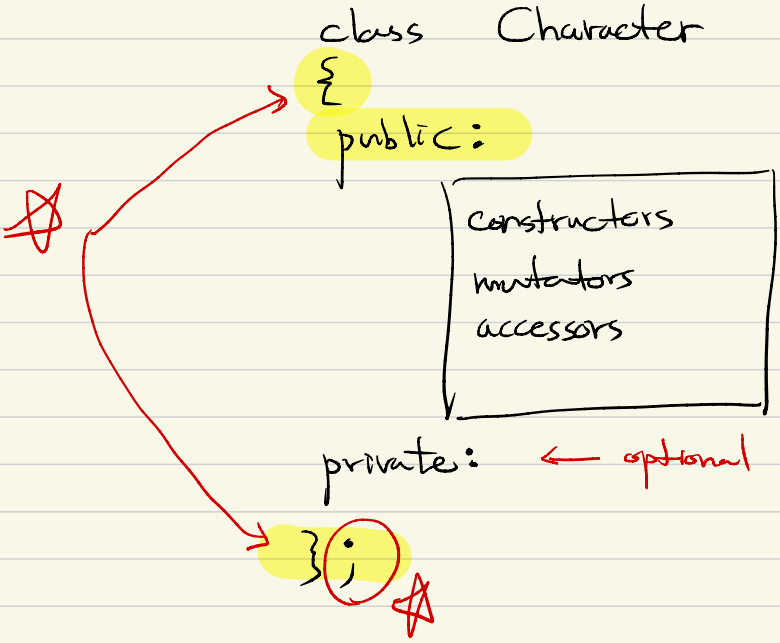
(1, 2)

(0, 3)

get 6.

```
class Character
{
public:
    Character();

void Character::set_health(int newh)
{
    health = newh;
};

private:
    string name;
    int health;
    int damage;
    int arrows;

string Character::get_name() const
{
    return name;
}

int Ch : get_health() const
{
    return health;
}
```




class Character
{
public:
constructors
mutators
accessors
private: ← optional
};

INSERT

```
#include <iostream>
#include <vector>

void insert (vector<int>& v1, vector<int>& v2, int start)
{
    TODO
}

int main()
{
    vector<int> v1 = {0, 1, 2, 3, 4, 5}
    vector<int> v2 = {-1, -2, -3}

    insert(v1, v2, 2);
```

v1 = { 0 1 2 3 4 5}

v1 = { 0 1 -1 -2 -3 2 3 4 5 }

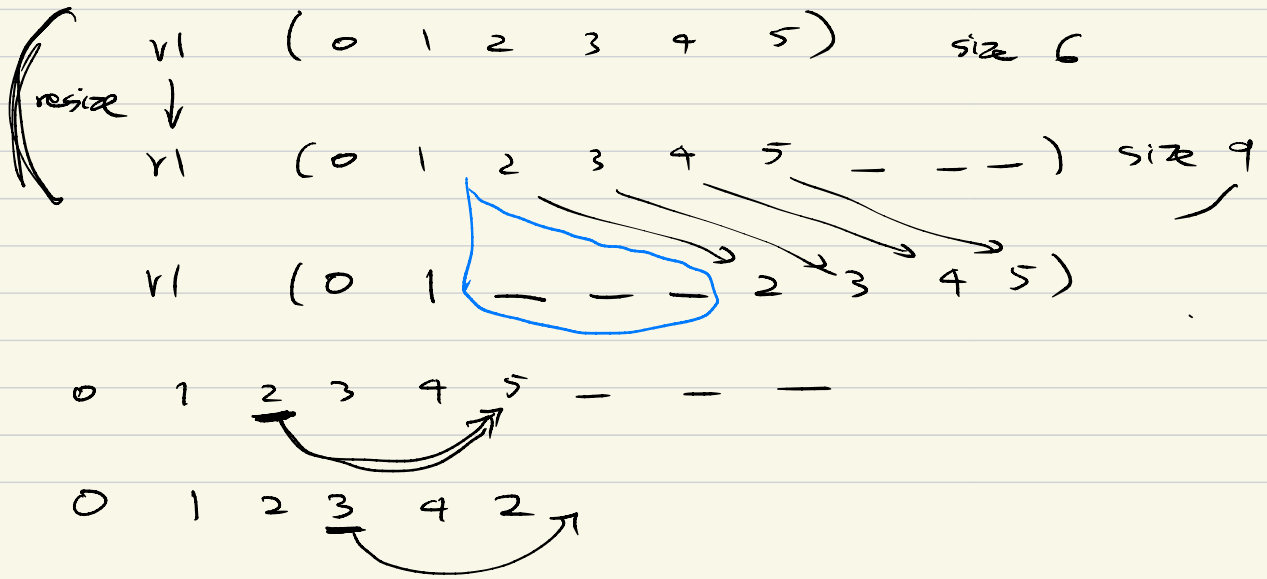
insert(v1, v2, 0) v1 = { -1 -2 -3 0 1 2 3 4 5}

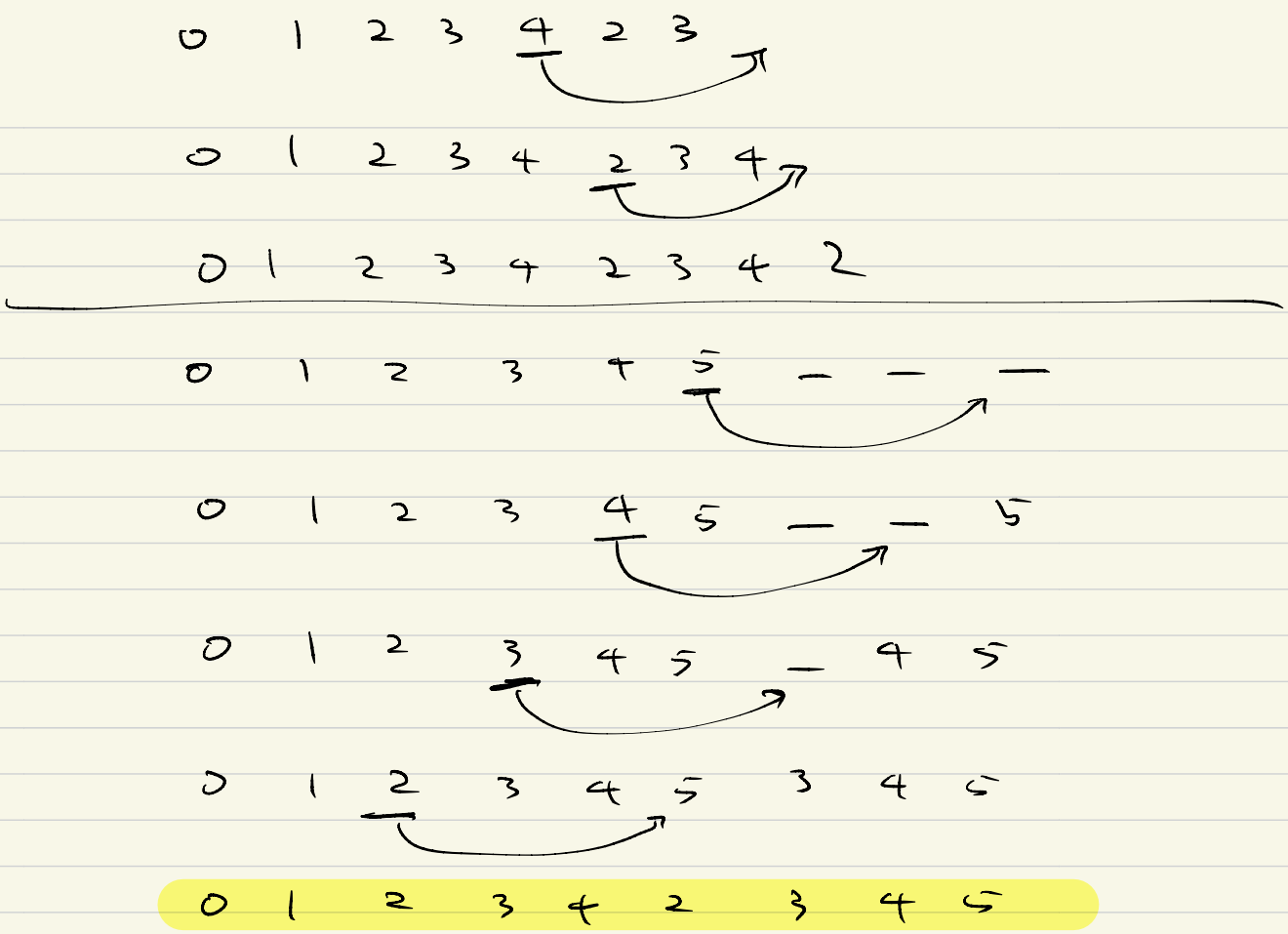
insert(v1, v2, 1) v1 = { 0 -1 -2 -3 1 2 3 4 5}

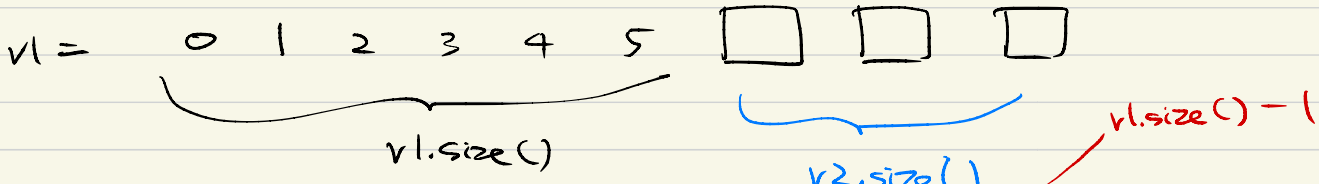
insert(v2, v2, 2) v1 = { 0 1 -1 -2 -3 2 3 4 5}

⋮

insert(v1, v2, 6) v1 = { 0 1 2 3 4 5 -1 -2 -3 }








```
v1[8] = v1[5];
v1[7] = v1[4];
v1[6] = v1[3];
v1[5] = v1[2];
```

```
for(int i=5; i>=2; i--)
{
    v1[i+3] = v1[i];
}
```

v1.size() - 1

start

v2.size()

0 1 2 3 4 2 3 4 5

↓ we want

0 1 -1 -2 -3 2 3 4 5

Recall

r2 = {-1, -2, -3}

```
v1[2] = v2[0];
v1[3] = v2[1];
v1[4] = v2[2];
```

```
for(int i=0; i<3; ++i)
{
    v1[i+2] = v2[i];
}
```

v2size

start

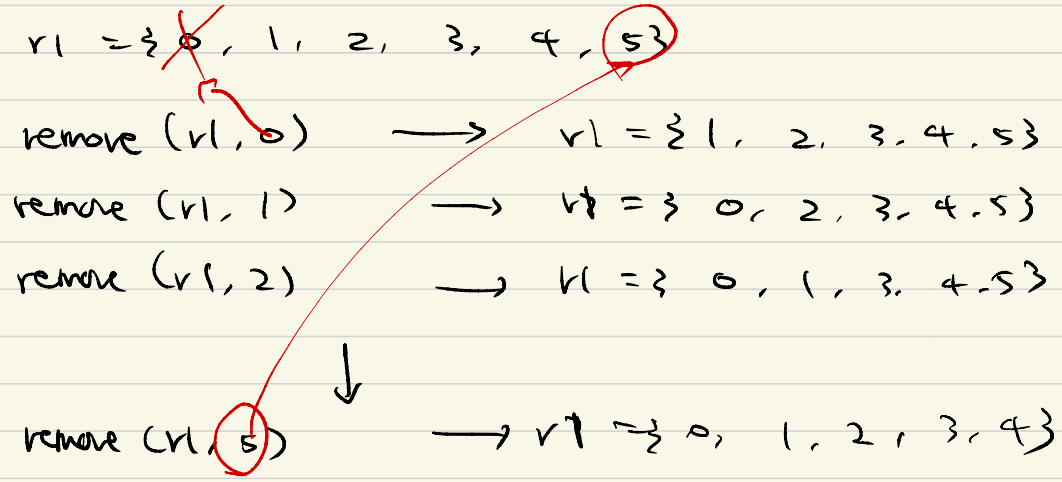
```
void insert( ... )
{
    int v1s = v1.size()   // 6
    int r2s = r2.size()   // 3
    v1.resize( v1s + r2s );
    for(int i=v1s-1; i>=start; --i)
    {
        v1[i+r2s] = v1[i];
    }

    for(int i=0; i<r2s; ++i)
    {
        v1[i+start] = r2[i];
    }
}
```

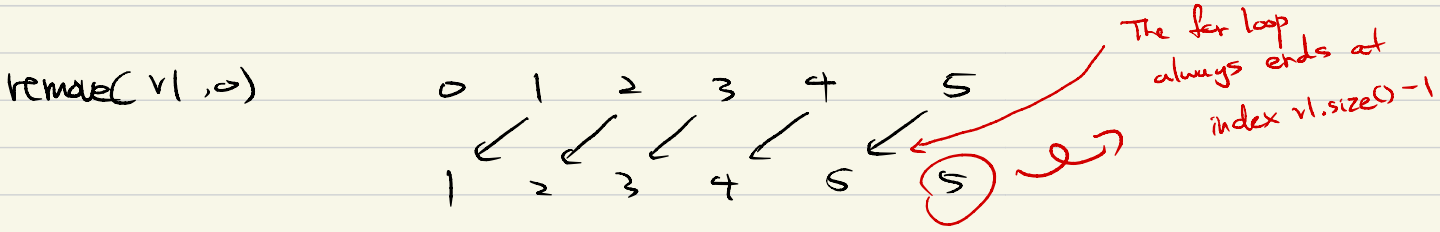
resize ✓

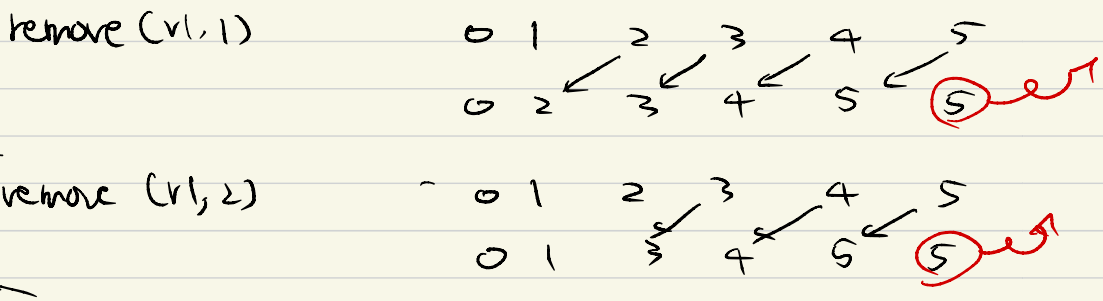
move ✓

replace ✓

Remove








```
v1[2] = v1[3];
v1[3] = v1[4];
v1[4] = v1[5];
```

```
for(int i = 2; i < 5; ++i)
{
    v1[i] = v1[i+1];
}
```

v1.size - 1

```
void remove(vector<int> v1, int start)
{
    int v1s = v1.size()
    for(int i = start; i < v1s - 1; ++i)
    {
        v1[i] = v1[i+1];
    }
    v1.pop_back();
}
```

move ✓

remove the last ✓