اشاره گر
Pointer

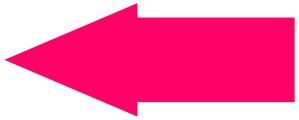
بهزاد منتظری
دانشکده فنی ومهندسی دانشگاه رازی
نیمسال دوم ۸۳-۸۲

اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

```
if (a > b) swap(a, b);
```

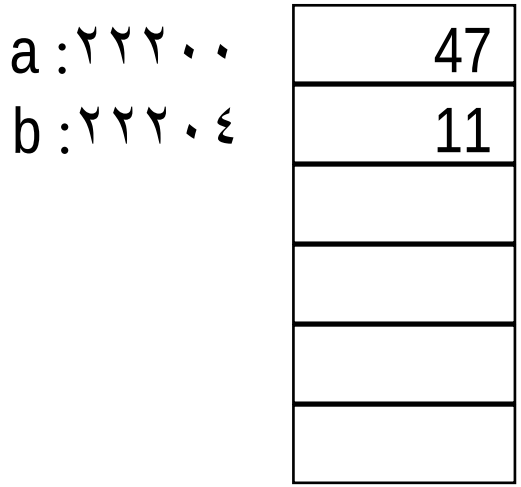


// a <= b بااجرای دستو فوق باید داشته باشیم

<u>پیاده سازی به فرم زیر</u>

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود



اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

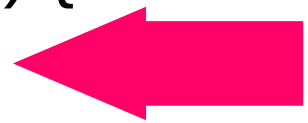
```
if (a > b) swap(a, b);
// a <= b  بااجرای دستو فوق باید داشته باشیم
```

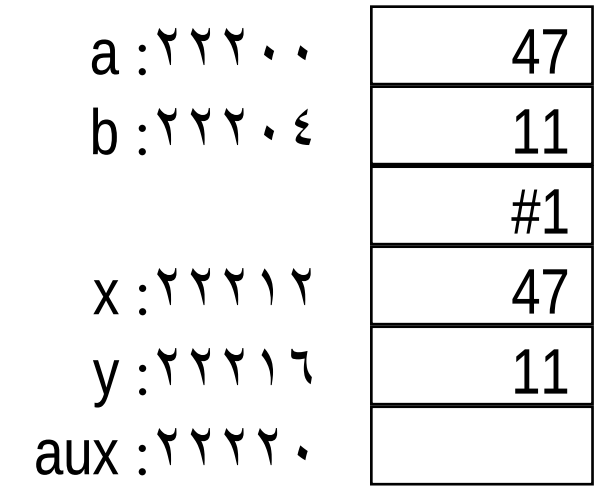
#1

<u>پیاده سازی به فرم زیر</u>

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود





اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

```
if (a > b) swap(a, b);
```

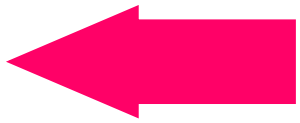
#1

// a <= b بااجرای دستو فوق باید داشته باشیم

<u>پیاده سازی به فرم زیر</u>

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود



a :۲۲۲۰۰	47
b :۲۲۲۰٤	11
	#1
x :۲۲۲۱۲	47
y :۲۲۲۱٦	11
aux :۲۲۲۲۰	47

اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

```
if (a > b) swap(a, b);
```

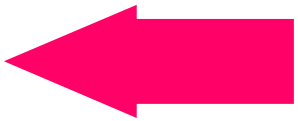
#1

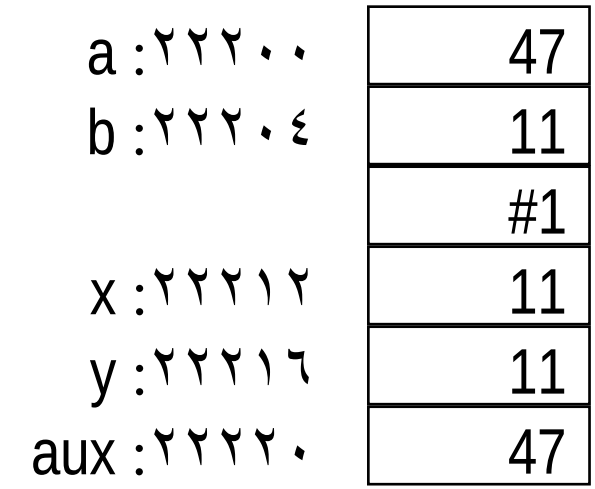
// a <= b بااجرای دستو فوق باید داشته باشیم

<u>پیاده سازی به فرم زیر</u>

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود





اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

```
if (a > b) swap(a, b);
// a <= b  بااجرای دستو فوق باید داشته باشیم
```

#1

پیاده سازی به فرم زیر

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود

a :۲۲۲۰۰	47
b :۲۲۲۰٤	11
	#1
x :۲۲۲۱۲	11
y :۲۲۲۱٦	47
aux :۲۲۲۲۰	47

اشاره گر

طرح مشکل: میخواهیم تابع swap(a,b) را برای جابجا کردن محتوای دو متغیرپیاده سازی کنیم.

مثال برای روش استفاده:

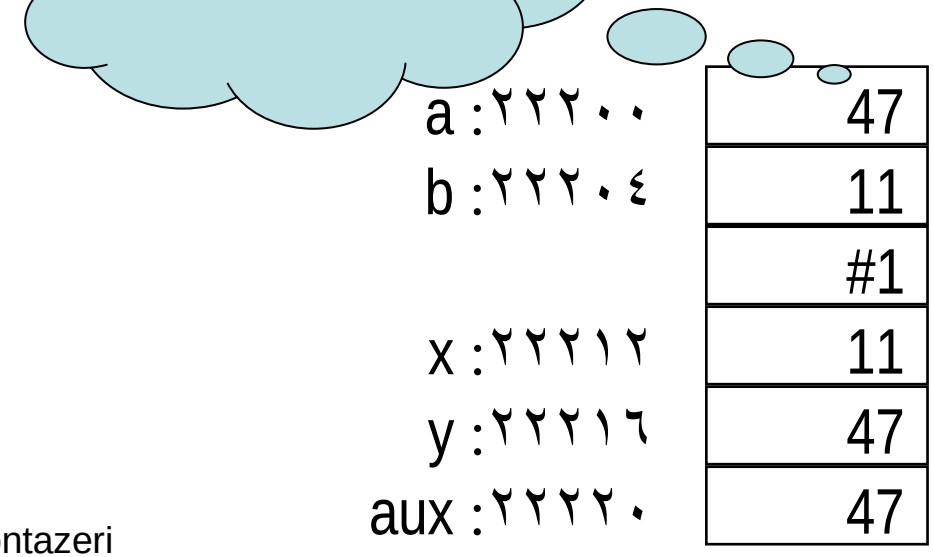
```
if (a > b) swap(a, b);
// a <= b  بااجرای دستو فوق باید داشته باشیم
```

پیاده سازی به فرم زیر

```
void swap(int x, int y) {
    int aux = x;
    x = y;
    y = aux;
}
```

حتما ناموفق خواهد بود

مقادیر a و b تغییر نکردند



اشاره گر

- مشکل:

مقادیر a و b در تابع کپی شدند. از این رو تغیر x و y اثری روی مقادیر a و b ندارند.

- راه حل:

اگر قرار است پارامترها توسط تابع تغییر کنند باید آدرس های متغیرهای مورد نظر (ونه مقادیر آنها) به تابع ارسال شوند.

متغیری که در آن آدرس یک شیئی (متغیر دیگری) قرار دارد متغیر اشاره گر Pointer variable نام دارد.

اشاره گر

- اظهار یک متغیرازنوع اشاره گر: در یک اظهار متعارف قبل از نام متغیر نماد * را قرار میدهیم.

```
int * x, * y, z;
```

x و y اشاره گر هایی به اشیاء ازگونه int هستند. متغیر میتواند یک شیئ از گونه int را قبول کند.

- آدرس یک متغیررا میتوان با عملگرتک عملوندی & بدست آورد.

```
&a → 22200, &b → 22204
```

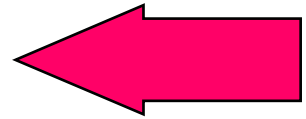
- ارزش شیئی که با یک اشاره گر به آن رجوع شده میتواند بکمک عملگر تک عملوندی * (dereference) عاید شود.

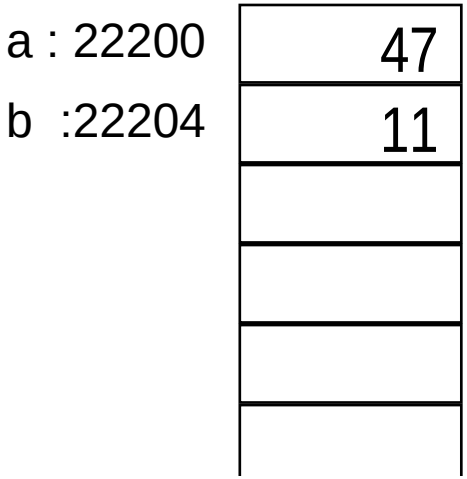
```
x = &a; cout << *x;  // خروجی ۴۷
y = &b; z = *x + *y; // z =47+11= 58
```

اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

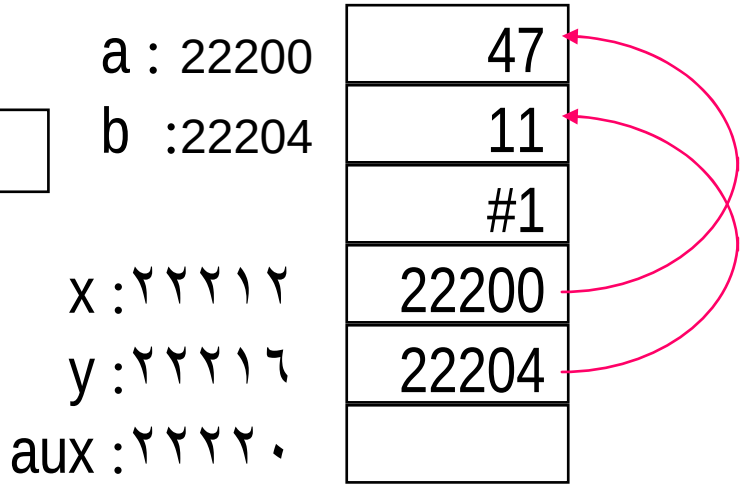




اشاره گر

- // swap اصلاح

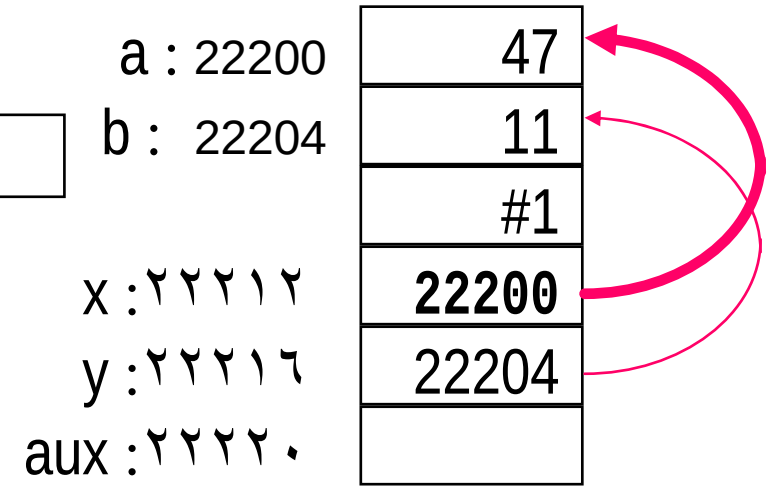
```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```



اشاره گر

- // swap اصلاح

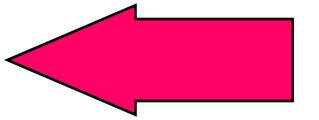
```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

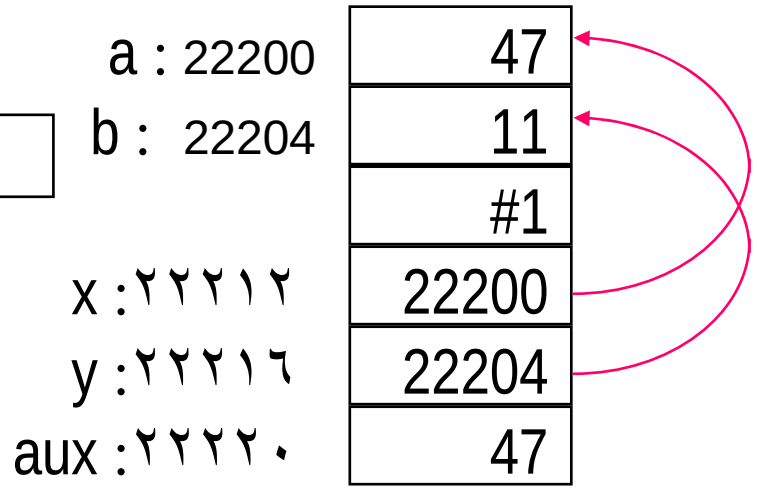


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

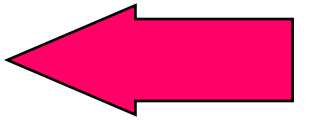


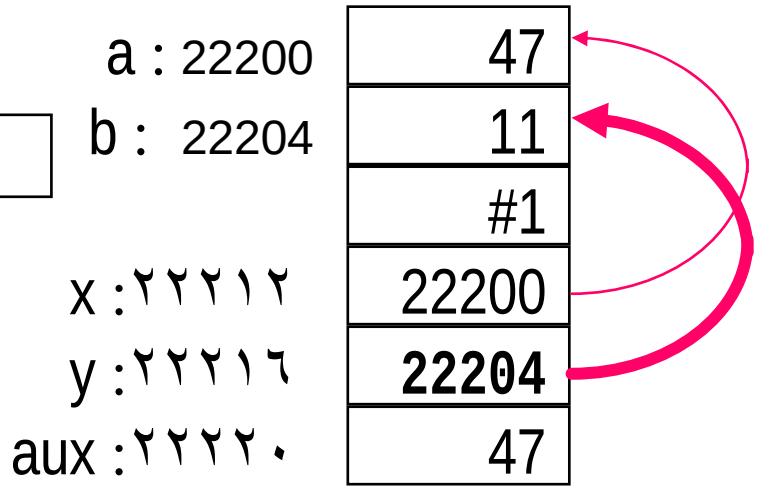


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

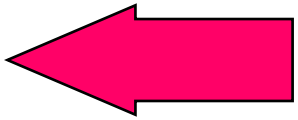


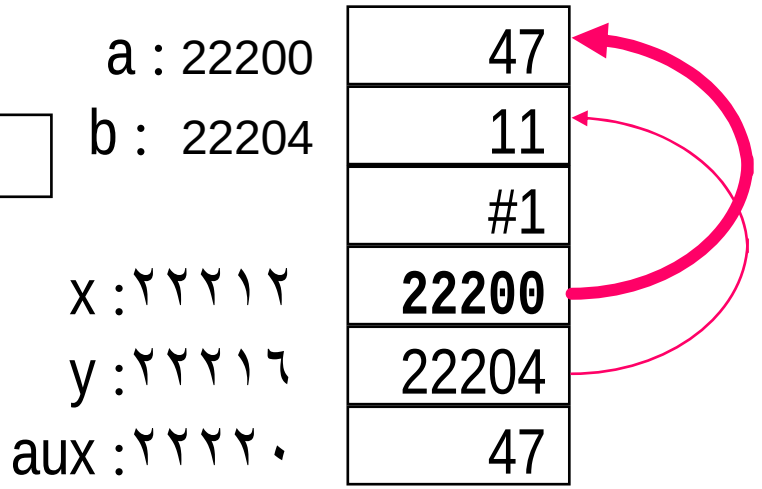


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

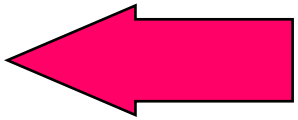


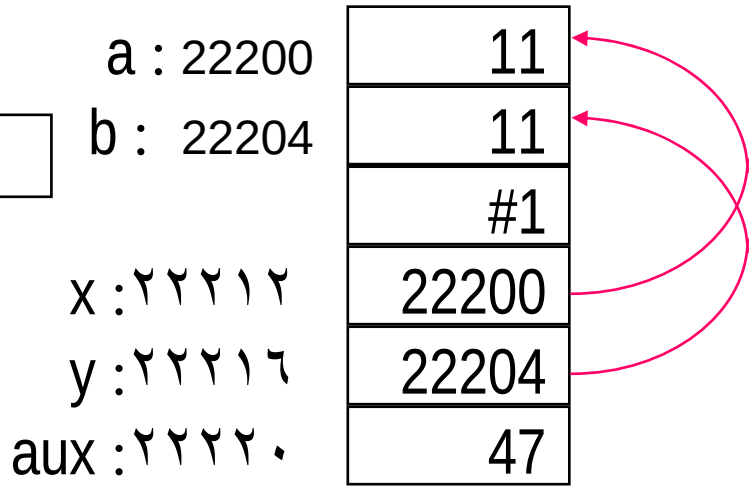


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

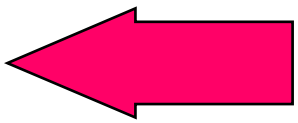


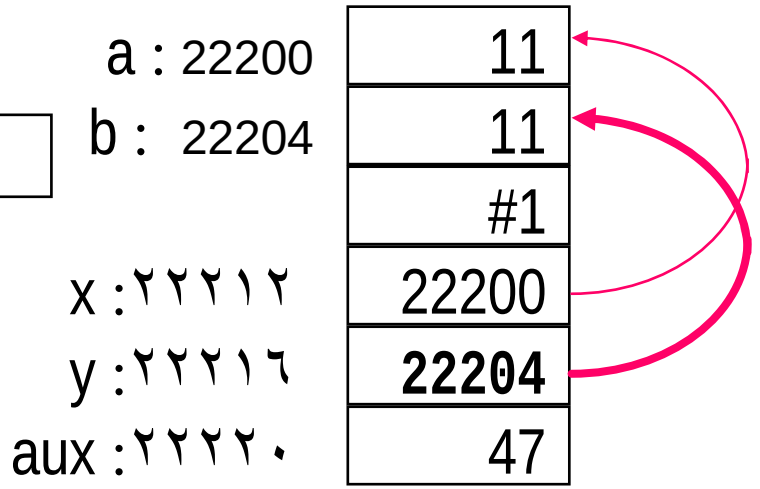


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
}
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
}
```

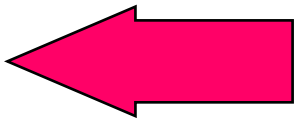


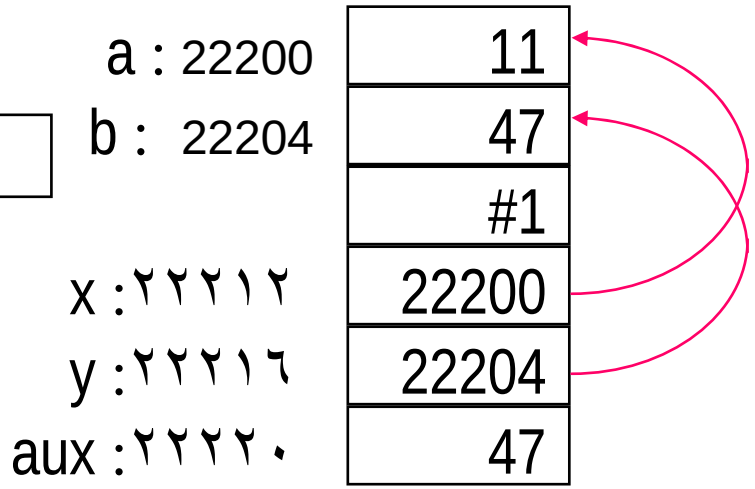


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```

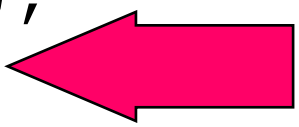


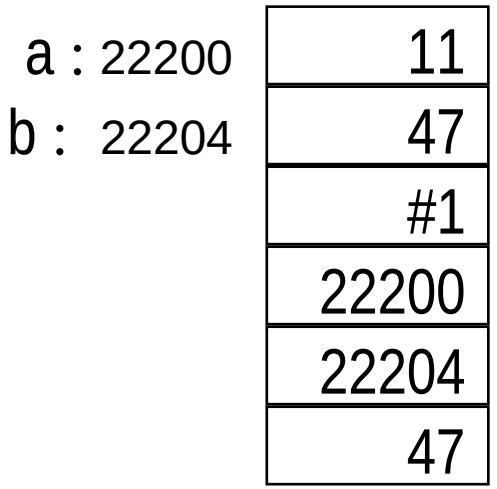


اشاره گر

- // swap اصلاح

```
void swap(int * x, int * y) {
     int aux = *x;
     *x = *y;
     *y = aux;
   }
int main () {
     int a, b;
     /* ... */
     if (a > b)
       swap(&a, &b);
   }
```



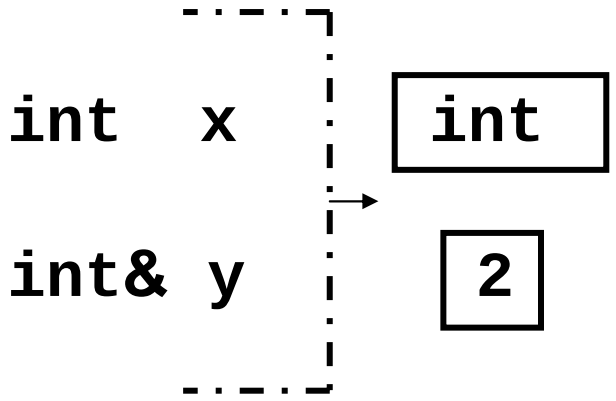


Reference مرجع

- مرجع نام مستعاری (alias) برای یک شیئ است.
- مرجع اجازه میدهد چندین متغیر به یک شیئ دستیابی داشته باشند.

```
int x = 2;
int& y = x;
```

y is a reference to an integer object



Reference مرجع

- اظهار یک مرجع همانند اظهار اشاره گر بوده فقط بجای * از & استفاده میشود.

- در C++ مرجع باید در زمان تعریف مقدارآغازی پیدا کند.

- در C++ مرجع پس از دادن مقدار آغازی نمیتواند به شیئ دیگری رجوع کند.

Reference مرجع

مقایسه مرجع و اشاره گر

```
int x = 7,y = 8;
int *p = &x;
cout << x << y << *p;
p = &y;
cout << x << y << *p;
```

خروجی

7 8 7 7 8 8

```
int x = 7, y = 8;
int &r = x;
cout << x << y << r;
r = &y; r = y;
cout << x << y << r;
```

مقدار x را تغییر میدهد

7 8 7 **8** 8 8

Passing by Reference

- از مرجع اغلب بعنوان پارمتر توابع استفاده میشود:

```
void swap(int *x,int *y){
    int aux = *x;
    *x = *y;
    *y = aux;
  }
```

تبدیل میشود به

```
void swap(int &x,int &y){
    int aux = x;
    x = y;
    y = aux;
  }
```

- روش استفاده از این توابع(ممکن است هر دو نگارش پیاده سازی شوند؛ چرا؟):

```
int a=7, b=12;
swap (&a, &b);
swap (a, b);
```

نگارش اشاره گر

نگارش مرجع

اشکال مرجع: در هنگام فراخوانی, امکان تغییر پارامتر توسط تابع قابل روئیت نیست.

Constant pointers and references

- برخلاف مرجع یک اشاره گر الزاماهمواره به یک شیئ رجوع نیکند.
- ولی میتوان با const آنها را به یک شیئ فیکس کرد.

px is a constant pointer to a double

```
double x = 32.6;
double* const px = &x;
x = 21.2; // ok
```

- توجه داشته باشید که در اظهار فوق px همواره به یک شیئ اشاره دارد، ولی مقدار خود شیئ میتواند تغییر کند.

Pointer to a constant variable

- میتوان اشاره گر را بنحوی اظهار کرد که به یک شیئ ثابت اشاره کند ولی خود اشاره گر ثابت نباشد.

```
double x = 32.6;
const double PI = 3.14;
PI = 3.2; // error
const double* pPI = &PI;
pPI = &x; // ok
```

- میتوان با دوبار استفاده از const در یک اظهار، اشاره گر ثابتی به یک شیئ ثابت اظهار نمود.

```
const double* const cpPI = &PI;
cpPI = &x // error
```

cpPI is a constant pointer to a constant double

Pointers to functions

- هرچند توابع متغیرنیستند، لیکن کد اجرایی تابع در حافظه اصلی دارای آدرس بوده و میتوان این آدرس را به اشاره گرارسال کرد.
- اشاره گر به توابع این امکان را به مامیدهد که توابع را بعنوان آرگومان به توابع دیگر بفرستیم.
- آدرس یک تابع ازطریق نام تابع بدون پرانتزبدست می آید.
- روش اظهار:

```
int (*f)(int)
```

اشاره گربه توابعی با یک پارامترازنوع int ومقداربازگشتی ازنوع int.

Pointers to functions

- پرانتز (f*) به دلیل اولویت ها لازم است. مثلا اظهاری به فرم زیر

```
int *f(int)
```

تابعی را اظهار میکند که مقدار بازگشتی آن اشاره گر به int است.

- مثال برای کاربرد اشاره گر به توابع:

```
double simpson(double(*f)(double),double a,double b,int n){…}
```

- روش فراخوانی تابع فوق به شکل زیر است:

```
cout << simpson(g,0,1,20);
cout << simpson(f,0,1,20);
```