شیءگرایی در C++

فرادرس

مؤلف : فرشید شیرافکن

دانشجوی دکترای بیوانفورماتیک دانشگاه تهران

ناشر: سازمان علمی آموزش فرادرس

بزرگترین پلتفرم آموزش آنلاین ایران

وب: www.faradars.org

تقدیم به:

روح پاک پدرم

- فرشید شیرافکن

فرادرس
فرادرس
فرادرس

سخن ناشر

در عین تمام نقدهای وارد شده به کنکور، هنوز راه‌حلی عملی که در جمیع جوانب، بهتر از سبک کوتاه و چند گزینه‌ای سؤالات باشد؛ ارائه نشده است. همین موضوع، کنکور را به ویژه کنکور کارشناسی ارشد به عنوان یک آزمون متمرکز و سراسری، از اهمیت دوچندانی برخوردار می‌کند.

یکی از آسیب‌های همراه با این آزمون سراسری این است که فضای رقابتی آن با ایجاد مؤسسات گوناگون، به سرعت از فضای یک رقابت علمی تبدیل به فضای رقابت اقتصادی می‌شود؛ به گونه‌ای که هزینه سرسام آور کلاس‌ها، دوره‌ها و منابع مرتبط با آزمون، از عهده بسیاری از دانشجویان خارج می‌شود. دانشجویانی که در عین استعداد تحصیلی بالا، در میدان رقابت مالی تحمیلی، در عین تمام شایستگی‌های خود، قدرت ادامه مسیر را از دست می‌دهند یا به نتیجه‌ای که در فضای مساوی مالی برای همه باید به آن می‌رسیدند، دست نمی‌یابند.

یکی از اهداف و آرمان‌های فرادرس به عنوان بزرگ‌ترین پروژه آموزش دانشگاهی اجرا شده بر بستر وب کشور، ایجاد دسترسی همگانی و یکسان به آموزش و دانش؛ مستقل از جغرافیا، زمان و سطح مالی دانشجویان بوده است. سیاست کاری فرادرس در راستای این آرمان، انتشار آموزش‌های ویدئویی تخصصی و دانشگاهی رایگان و یا بسیار کم هزینه، با تدریس مجرب‌ترین اساتید داخل و خارج کشور بوده است.

ما با انتشار رایگان این کتاب (به همراه نزدیک به ده کتاب رایگان دیگر) یکی از گام‌های دیگر خود را در راستای آرمان فرادرس برداشتیم. کتاب حاضر که حاصل نزدیک به دو دهه تدریس و پژوهش و تألیف مؤلف و مدرس فرادرس می‌باشد؛ در عین هزینه‌های بالای تألیف و آماده‌سازی، به جای انتشار و فروش؛ با تأمین مالی و سرمایه‌گذاری فرادرس به عنوان ناشر، به صورت کاملاً رایگان منتشر می‌شود. ما در گام‌های بعدی نیز تلاش خواهیم کرد که تا هر جا بتوانیم، حتی شده یک کتاب مرجع دیگر و بیشتر را با پرداخت هزینه، آزادسازی کرده و به صورت رایگان منتشر کنیم.

مؤلفین و ناشرینی که تمایل به واگذاری حق انتشار کتاب خود به فرادرس را دارند، می‌توانند با ایمیل ebooks@faradars.org مکاتبه نمایند. ما این کتاب‌ها را با پرداخت هزینه تألیف به مدرس و ناشر، به صورت رایگان منتشر خواهیم کرد تا همه دانشجویان مستقل از سطح مالی، به منابع مفید آزمون دسترسی داشته باشند. همچنین اگر ایده و نظری در خصوص کتاب‌های رایگان فرادرس داشته باشید، خوشحال می‌شویم که آن را با ایمیل ebooks@faradars.org مطرح نمایید.

سازمان علمی آموزش فرادرس

بزرگترین پلتفرم آموزش آنلاین ایران

وب: www.faradars.org

منبع مطالعاتی تکمیلی مرتبط با این کتاب

آموزش ویدئویی برنامه نویسی C++



عمومیت زبان C++ در میان زبان‌های برنامه‌نویسی بسیار بالا است و می تواند به عنوان اولین زبان نیز یاد گرفته شود و به پیش نیاز دیگر احتیاج نباشد.

مجموعه فیلم‌های آموزشی برنامه‌نویسی C++، با این فرض تهیه شده است که مخاطب هیچ دانش و تجربه قبلی در زمینه برنامه‌نویسی ندارد و در این مجموعه آموزشی، همه مباحث با بیان و تشریح مبانی نظری و سپس با پیاده‌سازی گام به گام مثال‌های عملی آموزش داده می‌شوند و از این نظر، در ایجاد یک دانش عمیق در زمینه برنامه‌نویسی، بسیار کارآمد است.

مدرس: مهندس فرشید شیر افکن

مدت زمان: ۲۰ ساعت

جهت مشاهده آموزش ویدئویی این آموزش – کلیک کنید faradars.org /fvcp9504

درباره مدرس



مهندس فرشید شیرافکن کارشناس ارشد مهندسی کامپیوتر گرایش نرم‌افزار است و در حال حاضر دانشجوی دکترای بیوانفورماتیک دانشگاه تهران هستند. ایشان از مدرسین نمونه در زمینه ارائه و آموزش دروس دانشگاهی انتخاب شده‌اند.

ایشان مشاور کنکور هستند و بیش از ۳۰ کتاب در زمینه کنکور رشته کامپیوتر تألیف نموده‌اند. ایشان در حال حاضر به عنوان یکی از برترین مدرسین فرادرس از جهت کمیت و کیفیت دروس ارائه شده، نزدیک به ۲۰ عنوان درسی را در قالب آموزش ویدئویی از طریق فرادرس منتشر کرده‌اند. این مجموعه دروس تا کنون مورد استفاده ده‌ها هزار دانشجوی سراسر کشور قرار گرفته اند.

مشاهده همه آموزش های تدریسی و تالیفی توسط مؤلف کتاب - کلیک کنید.

شماره همراه مدرس: ۰۹۱۲۱۹۷۲۰۲۸ Email: fshirafkan51@gmail.com

کتب رایگان دیگر از این مجموعه آموزشی

۱. آموزش زبان برنامه سازی C++ – کلیک کنید (+)
۲. آموزش نظریه زبان‌ها و ماشین – کلیک کنید (+)
۳. آموزش پایگاه داده‌ها – کلیک کنید (+)
۴. آموزش ساختمان داده‌ها – کلیک کنید (+)
۵. آموزش سیستم عامل – کلیک کنید (+)
۶. آموزش ذخیره و بازیابی اطلاعات – کلیک کنید (+)

برای دانلود رایگان این مجموعه کتب، به لینک زیر مراجعه کنید:

http://faradars.org/computer-engineering-exam

فرادرس

دسته‌بندی موضوعی آموزش‌های فرادرس، در ادامه آمده است:

			
برنامه نویسی – کلیک (+)	آموزش های دانشگاهی و تخصصی – کلیک (+)	هوش مصنوعی و یادگیری ماشین – کلیک (+)	مهندسی برق الکترونیک و روباتیک – کلیک (+)
			
دروس دانشگاهی – کلیک (+)	مباحث مشترک – کلیک (+)	مهارت های دانشگاهی – کلیک (+)	نرم افزارهای تخصصی – کلیک (+)
			
مهندسی نرم افزار – کلیک (+)	نرم افزارهای عمومی – کلیک (+)	طراحی و توسعه وب – کلیک (+)	آموزش های عمومی – کلیک (+)

فهرست مطالب

فصل ۱: کلاس – شی – سازنده و مخرب - تابع و کلاس دوست و ...

فصل ۲: اعضای کلاس با ویژگی استاتیک

فصل ۳ : وراثت

فصل ۴ : پلی مورفیسم

فصل ۵ : سربارگذاری عملگرها

فصل ۶ : قالب

فصل ۷ : فایل

فصل ۸ : آزمون با حل

فرادرس

فصل ۱ :

کلاس ها و اشیاء

کلاس مجموعه ای از متغیرها و توابع کار با آنها می باشد. اعضاء یک کلاس می توانند در یکی از طبقه بندی های زیر باشند:

Private	قابل دسترس از داخل کلاس
public	قابل دسترس از داخل و خارج کلاس
protected	قابل دسترس از داخل کلاس و یا کلاسهای ارث برده شده از این کلاس و توابع دوست آنها

تذکر: تا حد امکان به جای protected از private استفاده کنید. چون protected مفهوم بسته بندی را از بین می برد.

✱ در این مثال، یک کلاس به نام c با یک عضو خصوصی به نام a تعریف شده است:

```
class  c{
   private:
            int  a;
 };
void  main( ){
 c  x;
 cout<<x.a;
}
```

در ابتدای تابع main، یک متغیر به نام x از کلاس c تعریف شده است. به این متغیر ، شیء می گویند. باید توجه داشت که توسط این متغیر نمی توان از خارج کلاس به متغیر خصوصی a دسترسی داشت. بنابراین دستور x.a ، تولید خطا می کند.■

تذکر: به طور پیش فرض همه عناصر کلاس، خصوصی هستند. بنابراین کلمه private در مثال بال را می توان قرار نداد.

فرادرس

✱خروجی برنامه زیر کدام است؟

```
class a{
 private:
     int x;
 public:
    int f( ) {
        x=1;
       cout<<x;
     }
 };
void main( )
{
 a ob;
 ob.f( );
}
```

حل: خروجی ۱ می باشد. توجه کنید که به متغیر خصوصی کلاس یعنی x می توان از طریق تابع همان کلاس یعنی f دسترسی داشت.

تذکر: تابع f را می توان خارج از کلاس نیز به صورت زیر تعریف کرد:

```
class a{
 private:
     int x;
 public:
    int f( );
 };
int  a:: f( ) {
    x=1;
    cout<<x;
}
void main( ){
 a ob;
 ob.f( );
}
```

■

در هنگام تعریف کلاس نباید به متغیرها مقدار دهی اولیه کرد. بنابراین عبارت زیر نادرست است:

```
class a{
 private:
     int x=1;
 public:
    int f( );
 };
```

✱ تعریف یک کلاس برای صف حاوی کاراکترها:

صف ساختمان داده ای است که عمل اضافه کردن به انتها و عمل حذف از ابتدای آن انجام می شود. صف به کمک یک آرایه و دو متغیر front وrear پیاده سازی می شود. front به عنصر ما قبل اول و rear به عنصر آخر صف اشاره دارد. در ابتدا که صف خالی است، هر دو متغیر front وrear صفر می باشند. با هر بار اضافه کردن، یک واحد به rear اضافه شده و با هر بار حذف، یک واحد به front اضافه می شود.

```
#include  <iostream.h>
class  q{
  private:
     char    a[10];
     int      front , rear;
  public:
    void    set(void);
    void    insert(char ch);
    char    remove(void);
};
void   q ::set (void)  {
  front=0;
  rear=0;
}
void   q::insert(char  ch)
{
  if  (rear==9) { cout<<"full"; return ; }
  rear++;
  a[rear]=ch;
}
char   q::remove(void)
{
  if(front==rear){ cout<<"empty"; return  0;}
  front++;
  return   a[front];
}
main( ){
 q    x ;
  x.set ( );
  x.insert('A');
  x.insert('B');
  cout<<x.remove( );
}
```

در برنامه بالا ابتدا دو کاراکتر AوB به صف اضافه شده و سپس یک کاراکتر حذف شده و در خروجی نمایش داده می شود. بنابراین خروجی A است، چون حذف از اول صف انجام می گیرد.

فرادرس
فرادرس
فرادرس

سازنده و مخرب

در بیشتر برنامه ها، یکسری عملیات باید در هنگام ساخته شدن یک شیء و یا هنگام از بین رفتن شیء انجام شود. این عملیات توسط توابعی به نام های سازنده (constructor) و مخرب(destructor) انجام می گیرد. سازنده، تابعی هم نام با کلاس است که در هنگام ساخته شدن یک شیء از کلاس به طور خودکار صدا زده می شود و مخرب تابعی هم نام با کلاس (البته با یک علامت ~ قبل از اسم آن) می باشد که هنگام از بین رفتن شیء به طور خودکار فراخوانی می شود.

توابع سازنده و مخرب در قسمت public تعریف می شوند.

✱خروجی برنامه زیر چیست؟

```
class  c{
        private:
                int  x;
        public:
              c( ) {cout<<'a';   }
};
void main( ){
  c  t;
}
```

حل: خروجی برنامه a است. به محض ایجاد شیء t در داخل تابع main ، تابع سازنده (تابع هم نام با اسم کلاس)، به طور خودکار صدا زده می شود.

✱خروجی برنامه زیر چیست؟

```
class  c{
    private:
       int  x;
    public :
        c(int=3);
        int  fun( );
};
c::c(int n)   {  x=n; }
int  c::fun( ) { x++;   return  x; }
void main( ){
  c  ob;
  cout<<ob.fun( );
}
```

حل: خروجی برنامه 4 است. ابتدا شیء ای به نام Ob از کلاس c ساخته شده که باعث صدا زدن اتوماتیک تابع سازنده می شود. که این تابع مقدار 4 را در متغیر خصوصی x ذخیره می کند.

✱ خروجی کدام است؟

```
class  c{
       private:  int   x;
        public:
                c(int=0);
                void   print( );
};
c :: c(int a)          { x=a;}
void   c::print( )   {  cout<<x; }
void  main( ){
   c  ob1(1);
   c   ob2(2);
   ob1.print( );
   ob2.print( );
}
```

حل: خروجی 12 می باشد. ابتدا شیء ای به نام ob1 از کلاس c ایجاد شده و سازنده صدا زده مـی شـود و مقـدار x برابـر 1 خواهد شد. سپس شیء دیگری به نام ob2 از کلاس c ساخته شده و سازنده صدا زده می شود و مقدار x برای این شیء برابر 2 خواهد شد. در خط آخر برنامه، تابع print توسط هر دو شیء صدا زده شده و ابتدا مقدار 1 و سپس مقدار 2 چاپ خواهـد شد. توجه کنید که هر شیء دارای اعضای مخصوص به خود می باشد.

✱خروجی برنامه زیر چه می باشد؟

```
class  c{
   public:
           c( )    { cout<<'a'; }
           ~c( )  {cout<<'b'; }
};
void main( ){
  c  t;
 cout<<'c';
}
```

حل: خروجی برنامه acb می باشد. هنگامی که شیء t در تابع main ساخته می شود، تابع سازنده به طور خودکـار صـدا زده شده و کاراکتر a را نمایش می دهد. سپس کاراکتر c نمایش داده شده و در نهایت با اتمام برنامه، تابع مخرب به طور خودکار صدا زده شده و کاراکتر b را چاپ می کند.

یک کلاس می تواند بیش از یک تابع سازنده داشته باشد.

✱خروجی برنامه زیر چه می باشد؟

```
class  test{
 public:
   test( )          {cout<<'a';}
   test(int  m)    {cout<<m;}
```

```
};
main( ){
  test  x(1);
  test  y;
}
```

حل: خروجی برنامه 1a می باشد. زمانی که شیء x ساخته می شود، تابع سازنده که دارای پارامتر ورودی است صدا زده می شود و زمانی که شیء y ساخته می شود، تابع سازنده بدون پارامتر ورودی صدا زده می شود.

■

- تابع سازنده یک شیء سراسری، فقط یکبار آنهم در هنگام شروع اجرای برنامه، فراخوانی می شود.
- تابع سازنده یک شیء محلی، با هر اجرای دستور معرفی آنها، فراخوانی می شود.
- یک شیء سراسری زمانی که برنامه تمام شود از بین می رود.
- یک شیء محلی زمانی از بین می رود که اجرای برنامه از میدان دید آنها خارج شود.

✱خروجی برنامه زیر چه می باشد؟

```
class   c{
   public:
       c( )     { cout<<'a'; }
      ~c( )   {cout<<'b'; }
};
void  main( ){
   c   a,b;
  { c   d;  }
   cout<<'c';

}
```

حل: خروجی برنامه aaabcbb می باشد. ابتدا شی های a,b ساخته شده و دوبار سازنده صدا زده شد و aa چاپ می شود. سپس شیء دیگری به نام d ساخته شده و سازنده صدا زده شده و a چاپ می شود. بعد از پایان حوزه دید d ، مخرب صدا زده شده و یک b نمایش داده شده و بعد از آن یک c و در نهایت بعد از پایان برنامه به علت اینکه در ابتدای برنامه دو شیء تعریف شده بود، دو بار مخرب صدا زده شده و دو b نمایش داده می شود.

■

✱خروجی برنامه زیر چه می باشد؟

```
class   c{
```

```
   public:
       c( )   {cout<<'a';}
      ~c( )  {cout<<'b';}
};
void  main( ){
    c   ob1,ob2;
  {c   ob3;  }
    c   ob4;
}
```

حل: خروجی aaababbb می باشد.

نمی توان آدرس توابع سازنده و مخرب را به دست آورد.

تابع مخرب دارای پارامتر ورودی و یا خروجی نمی باشد.

فرادرس

انتساب اشیاء به یکدیگر

می توان اشیاء با نوع های یکسان را به یکدیگر نسبت داد. با این عمل، داده های یک شیء در شیء دیگر کپی می شود. در مثال زیر ob1وob2 هر دو از نوع کلاس c می باشند که توسط دستور ob2=ob1; ، اعضای داده ای ob1 در ob2 کپی می شوند. خروجی برنامه 2 می باشد:

```
class  c{
 public :   int  x;
            void  f( ) { cout<<++x;}
};
void main( ){
 c   ob1,ob2;
 ob1.x=1;
 ob2=ob1;
 ob2.f( );
}
```

آرایه ای از کلاس

می توان آرایه ای تعریف کرد که اعضای آن کلاس باشند. در مثال زیر آرایه x با ۳ خانه، آرایه ای از کلاس c می باشد:

```
class  c{
              int  a;
      public: c(int i)  {a=i;}
              int  f( )  {return  a;}
};
int  main( ){
 int   i;
 c   x[3]={c(5),c(4),c(3) };
 cout<<x[0].f( ) + x[2].f( );
}
```

خروجی این برنامه 8 می باشد. در تابع main ، ابتدا آرایه ای سه خانه ای به نام x از نوع کلاس c ساخته شده که محتویات متغیر a برای این خانه ها برابر ۵و۴و۳ می باشد. این برنامه محتویات خانه اول و آخر آرایه را با هم جمع می کند.

ارسال شیء به تابع

اشیاء را می توان به عنوان آرگومان به توابع ارسال کرد.

✱خروجی برنامه زیر ۲ می باشد. در تابع main ابتدا شیء ای به نام x ساخته شده و در این زمان سازنده اجرا شده و عدد ۲ را در i ذخیره می کند. سپس در خط بعدی ، شیء x به تابع k ارسال شده که منجر به صدا زدن تابع f می شود.

```
class  c{
      int   i;  int  j;
```

```
   public:  c(int  n)    { i=n;}
              int   f( )    { j=i; return  j; }
};
void   k(c   ob) { cout<<ob.f( ); }
void main( ){
 c   x(2);
 k(x);
}
```

تابع سازنده هنگام ارسال شیء به تابع اجرا نمی شود.

تابع مخرب هنگام از بین رفتن شیء در زمان پایان اجرای تابع، اجرا می شود.

*خروجی چیست؟

```
class  c{
      int   i;  int  j;
   public:   c(int  n)    { i=n;}
              ~c( )         {cout<<'b';}
              int   f( )    { j=i; return  j; }
};
void   k(c   ob)  { cout<<ob.f( ); }
void main( ){
 c   x(2);
 k(x);
cout<<'a';
}
```

حل: خروجی 2bab می باشد. توجه کنید که تابع مخرب دو بار صدا زده شده است. (یک بار در پایان تابع k به علت از بین رفتن شی ء ob و یکبار در پایان اجرای برنامه اصلی به علت از بین رفتن شیء x)

فرادرس

تابع با خروجی از نوع شیء

خروجی یک تابع می تواند از نوع شیء باشد. در مثال زیر خروجی تابع k از نوع کلاس c می باشد:

```
class  c{ public:  int  i;};
c  k( ) {
     c  ob;
     ob.i=2;
     return ob;
 }
void main( ){
  c  x;
  x=k( );
  cout<<x.i;}
```

خروجی برنامه بالا، ۲ می باشد.

■

✱خروجی برنامه زیر چه می باشد؟

```
class  c{
   public:  int  i ;
            ~c( )  {cout<<'a';}
};
c  k( ) {
     c  ob;
     ob.i=2;
     return ob;
 }
void main( ){
  c  x;
  x=k( );
  cout<<x.i;
}
```

حل: خروجی aa2a می باشد. توجه کنید که تابع مخرب ۳ مرتبه صدا زده شده است. ■

توابع دوست (friend function)

تابعی که عضو کلاس نباشد، اما بتواند به اعضای خصوصی کلاس دسترسی داشته باشد را تابع دوست می نامیم. در مثال زیر تابع f که عضو کلاس c نمی باشد، به عضو خصوصی کلاس یعنی a می تواند دسترسی داشته باشد:

```
class  c{
    int  a;
  public:
       c(int  x)  { a=x;}
       friend  int   f(c) ;
```

```
};
int  f(c  z)
 { cout<<z.a;  }
main( ){
  c  ob(3);
  f(ob);
}
```

تذکر: توجه کنید که باید پیش الگوی تابع f را در کلاس c قرار دهیم.

تذکر: دستوری مانند ob.f( ) غلط است، چون تابع f عضوی از کلاس c نمی باشد.

تذکر: دستور cout<<a; در تابع f نادرست است، چون تابع f فقط از طریق یک شیء می تواند به عضو خصوصی کلاس دسترسی داشته باشد.

✱خروجی برنامه زیر چیست؟

```
class  c{
          int  x;
    public:
          c(int  i ) {x=i;}
        friend  f(c  z )  {  cout<<z.x; }
        k( ) { cout<<x;}
};
void  main( ){
  c  ob(2);
  f(ob);
   ob.k( );
}
```

حل: خروجی برنامه 22 است. توجه کنید که برای صدا زدن تابع دوست f در تابع main، نباید از دستور ob.f( ) استفاده شود، چون تابع f عضوی از کلاس نمی باشد ولی تابع k را به صورت ob.k( ) صدا می زنیم چون تابع k عضو کلاس است.

■

تابع دوست می تواند دوست دو کلاس باشد.

✱خروجی چه می باشد؟

```
class  d;
class  c{
      int  i;
 public:
      c(int x){i=x;}
      friend  int  f(c  ob1,d  ob2);};
class  d{
          int  i;
```

```
 public:
        d(int  y)  {i=y;}
        friend int f(c  ob1,d  ob2);
};
int f(c   ob1,d   ob2)
{
 return  ob1.i+ob2.i;
}
main( )
{
  c   m(3);
  d   n(4);
 cout<<f(m,n);
}
```

حل: خروجی برنامه ۷ می باشد. توسط تابع دوست می توان به عضو مشترک بین دو کلاس دسترسی داشت. در مثال بالا متغیر i در هر دو کلاس وجود دارد و توسط تابع دوست f به این متغیرهای مشترک دسترسی شده است.

تذکر: در تابع f موجود در کلاس c، نامی از d برده شده در حالی که کلاس d بعدا معرفی شده است. بنابراین قبل از تعریف کلاس c باید خط زیر را نوشت. به این عمل تعریف پیشاپیش (forward declaration) می گویند:

```
class  d;
```

تابع دوست یک کلاس می تواند عضو کلاس دیگری باشد.

کلاس دوست

کلاسی که دوست کلاس دیگری می باشد، می تواند به اعضای خصوصی آن دسترسی داشته باشد. در مثال زیر کلاس d دوست کلاس c است، بنابراین می تواند به متغیر i دسترسی داشته باشد:

```
class   c{
        int  i;
   public:
        c(int  x)  { i=x;}
        friend   class   d;
};
class   d{
   public:
        int  f(c  z) {cout<<z.i;}
};
main( )
{
```

```
  c   ob1(3);
  d   ob2;
  ob2.f(ob1);
}
```

خروجی برنامه بالا، ۳ می باشد.

■

فرادرس
فرادرس
فرادرس

توابع inline

توابعی که به صورت inline تعریف می شوند، فراخوانی نمی شوند و کد انها در مکانی که صدا زده شده، کپی می شود. این عمل باعث افزایش سرعت اجرای برنامه می شود (به علت عدم نیاز به انتقال اطلاعات توسط پشته)، ولی حجم برنامه زیاد می شود.

*خروجی برنامه زیر ۲ می باشد و گذاشتن کلمه inline تاثیری در خروجی ندارد:

```
inline int  f(int x)  {x=x*2; return  x; }
void  main( ){
  cout<<f(1);
}
```

وقتی تابعی در داخل اعلان کلاس ظاهر شود، به طور خودکار به یک تابع inline تبدیل می شود. در این صورت نیازی به کلمه کلیدی inline نیست.

فرادرس

فصل دوم

اعضای کلاس با ویژگی استاتیک

هر یک از اعضای کلاس(اعضای داده ای و یا توابع عضو)، را می توان به صورت static تعریف کرد.

اعضای داده ای استاتیک

وقتی یک متغیر کلاس را static معرفی می کنید، به کامپایلر می گویید که فقط یک کپی از آن وجود دارد و تمام اشیای آن کلاس، آن متغیر را به اشتراک می گذارند.

- مقدار اولیه متغیر عضو استاتیک برابر صفر می باشد.
- مقدار دهی اولیه متغیر عضو استاتیک نمی تواند در داخل کلاس انجام شود.
- وقتی یک عضو داده ای static را در داخل کلاسی اعلان کنید، حافظه به آن اختصاص نمی یابد و باید در خارج از کلاس به صورت عمومی تعریف شود. این کار به کمک عملگر تعیین حوزه (::) ، برای شناسایی کلاسی که متغیر به آن تعلق دارد، انجام می شود.

✱خروجی چیست؟

```
class  c{
        int  a;  static  int  b;
  public:
       f (int  x, int  y)  { a=x;b=y;}
       void   g( )        { cout<<a<<b;}
};
int  c::b;
main( ){
  c  ob1,ob2;
  ob1.f(1,2);
  ob2.f(3,4);
  ob1.g( );
  ob2.g( );
}
```

حل: خروجی 1434 می باشد. ابتدا دو شیء از کلاس c تعریف شده و تابع f توسط این شیء صدا زده شده و مقدار a برابر 1 و مقدار b برابر 2 می شود. سپس تابع f توسط شیء ob2 صدا زده شده و مقدار a برابر 3 و مقدار b برابر 4 خواهد شد.

سپس با فراخوانی تابع g ، مقدار a,b مربوط به شیء ob1 و سپس ob2 چاپ می شود. دقت کنید که چون متغیر b از نوع استاتیک است، مقدارش برابر آخرین مقدار ذخیره شده در آن یعنی 4 می باشد ولی مقدار a برای هر شیء متفاوت است. به عبارتی برای ob1وob2 که دو شی از کلاس c می باشند، یک b مشترک وجود دارد. (تذکر: اگر static قبل از b را برداریم، خروجی 1234 خواهد بود.)

✱خروجی چیست؟

```
class  c{
  public: static int  x;
          c( )  {x++;}
};
int  c::x;
main( ) {
   c  m,n,p; cout<<c::x;
   c  q;  cout<<c::x;
}
```

حل: خروجی۳۴ می باشد. ابتدا ۳ شی m,n,p ساخته شده و سه مرتبه سازنده صدا زده می شود و مقدار x برابر ۳ شده و در خط بعد چاپ می شود. سپس شیء دیگری به نام q ساخته شده و سازنده صدا زده شده و به مقدار x قبلی یک واحد اضافه شده و برابر ۴ می شود.

تذکر: از آنجا که قبل از ایجاد شیءای از کلاس، متغیر عضو static وجود دارد، در داخل برنامه می توان مستقل از هر نوع شیئی به آن دسترسی داشت. دستور cout<<c::x; ، در مثال قبل معرف همین موضوع است.

تذکر: از آنجا که متغیرهای سراسری، با اصل کپسوله سازی در تضاد می باشند، متغیرهای عضو استاتیک معرفی شدند.

توابع عضو استاتیک

توابع عضو را نیز می توان به صورت استاتیک معرفی کرد. این توابع فقط مستقیما می توانند به اعضای استاتیک کلاس مراجعه کنند. از این توابع، می توان برای مقداردهی اولیه داده های استاتیک خصوصی، قبل از ایجاد شیء استفاده کرد. (تذکر: تابع عضو استاتیک می تواند به اعضای عمومی کلاس مراجعه کند.)

✱خروجی چیست؟

```
class  c{
          static  int  s;
  public:  static  int  f( );
};
int  c:: s=3;
int  c::f( ){  cout<<s; }
int  main( ){
  c  x;  x.f( );
  c::f( );
}
```

حل: خروجی 33 می باشد. این برنامه، دو نحوه فراخوانی تابع static را نشان می دهد:

۱- به کمک یک شیء (دستور;( ) x.f)

۲- با استفاده از نام کلاس و عملگر تعیین حوزه (دستور ;( )c::f)

تابع عضو استاتیک می تواند به داده استاتیک خصوصی، قبل از این که شیء ایجاد شود، مقدار اولیه دهد.

✱ خروجی برنامه، 3 می باشد:

```
class c{
            static int s;
  public:  static int f(int x ) {s = x; }
            k( ) {cout<<s;}
};
int   c:: s;
int main( ){
  c::f(3);
  c ob;
  ob.k( );
}
```

تذکر: شاگر هر یک از عبارات static در برنامه بالا را حذف کنیم، کامپایلر خطا می گیرد.

✱ خروجی چه می باشد؟

```
class c{
           static  int  a;
           int  b;
 public: static  f( ){ a=1; b=2; }
};
int c::a=3;
main( ){
 c ob;
 ob.f( );
}
```

حل: برنامه خطا دارد. نمی توان به عضو b در داخل تابع f بدون استفاده از شیء دسترسی داشت. (تابع عضو استاتیک فقط می تواند مستقیما به متغیر استاتیک دسترسی داشته باشد.)

تذکر: اگر b از اعضای عمومی بود، خطا گرفته می شد.

اعضای کلاس با ویژگی ثابت

هر یک از اعضای داده ای و یا توابع عضو یک کلاس را می توان به صورت const تعریف کرد.

✱خروجی چیست؟

```
class  c{
    int   a;
 public:   c(int  b=1): a(b)  {   }
           int    f( )  const   {return  ++a;}
};
main( ){
 c  ob;  cout<<ob.f( );
}
```

حل: خطا دارد. تابع f نمی تواند مقدار a را تغییر دهد. اگر const بعد از تابع f را برداریم، خروجی 2 خواهد بود.

تابع عضو کلاس را می توان const تعریف کرد. در این صورت، تابع نمی تواند شیءای را که موجب فراخوانی آن شده است را تغییر دهد.

شیء ثابت نمی تواند تابع عضو غیر ثابت را فراخوانی کند.

✱خروجی چیست؟

```
class   c{
              int    x;
                const  int  y;
     public: c( int  a, int  b) : x(a),y(b) { };
               f1( )  { x =y+1;}
              f2( )  const {cout << x<<y;}
};
int main( ){
  c   ob(1,2);
  ob.f1( );
 ob.f2( );}
```

حل: خروجی 32 است.

تذکر: استفاده از دستوری مانند y=x+1; در تابع f1 و یا f2 نادرست است، چون مقدار ثابت y را نمی توان تغییر داد.

✱خروجی چیست؟

```
class  c{
 public:  int  x;
          c( )  { cout<<'a'; }
};
main( ){
```

```
 const  c  ob;
 ob.x=1;
}
```

حل: برنامه خطا دارد. اگر const نباشد، خروجی a می باشد.

- تابع عضو استاتیک نمی تواند حاوی اشاره گر this باشد.
- تابع عضو استاتیک نمی تواند مجازی باشد.
- تابع عضو استاتیک نمی تواند به صورت const و volatile تعریف شود.

اشاره گر به اشیاء

می توان یک اشاره گر به یک شیء تعریف کرد. برای دسترسی به اعضای شیء توسط اشاره گر، از علامت <- (منها و علامت بزرگتر) ، باید استفاده کرد. در مثال زیر متغیر p اشاره گری از این نوع است:

```
class  c{
   int  x;
public:  c(int  y)  { x=y;}
         int  f( )  { return  x;}
};
int  main( ){
  c  a(2);
  c  *p;
  p=&a;
  cout << p->f( );
}
```

در ابتدا شیء a از کلاس c تعریف شده و مقدار اولیه x توسط تابع سازنده برابر ۲ می شود. سپس اشاره گر p از نوع کلاس c تعریف شده و آدرس شیء a در آن قرار می گیرد. در نهایت تابع f که عضوی از p است، اجرا شده و مقدار 2 در خروجی چاپ می گردد.

* خروجی برنامه زیر چیست؟

```
class  c{
   int  x;
public: c(int  y)  { x=y;}
         int  f( )  { return  x;}
};
int  main( ){
  c  a[2]={5,3};
```

```
  c  *p;
  p=a;
  p++;
  cout << p->f( );
}
```

حل: خروجی این برنامه 3 است. ابتدا یک آرایه از کلاس c ساخته شده و سپس یک اشاره گر به کلاس c به نام p تعریف می شود که با دستور p=a;، اشاره گر به خانه اول آرایه اشاره می کند و بعد از اجرای دستور p++;، به خانه دوم آرایه اشاره می کند و در نهایت مقدار 3 چاپ خواهد شد. ■

✱خروجی برنامه زیر چیست؟

```
class  c{
  public:
     int  x;
     void  f( ) { cout<<'a';}
  };
void main( ){
  c   ob,*p;
  p=&ob;
  ob.f( );
  p->f( );
  (*p).f( );
}
```

حل: خروجی aaa می باشد. هر سه دستور آخر معادل هم می باشند. (برای دسترسی به تابع f، از سه روش استفاده شده است.)

■

✱خروجی برنامه زیر 1 می باشد. متغیر p در این مثال، اشاره گر به x است:

```
class  c{
    public: int  x;
            c(int  a)  {x=a;}
};
int  main( ){
  c   ob(1);
  int c::*p;
  p=&c::x;
  cout<<ob.*p;
}
```

■

✱خروجی برنامه زیر ۲ می باشد. متغیر p در این مثال، اشاره گر به تابع f می باشد:

```
class  c{
  public:  int  x;
           c(int  a)  {x=a;}
```

```
        int  f( )  { return  ++x;}
};
int  main( ){
  c   ob(1);  int  (c::*p) ( ) ;
  p = &c::f;
 cout<<(ob.*p)( );
}
```

■

دستیابی به عضو عمومی شیء از طریق اشاره گر

می توان آدرس عضو عمومی شیء را به اشاره گری نسبت داد و از طریق آن اشاره گر به آن دست یافت. به مثال زیر که خروجی آن 3 می باشد، توجه کنید:

```
class  c{ public: int  a; };
int  main( ){
 c   ob;  int  *p;
 p=&ob.a;
 *p=3;
 cout<<*p;
}
```

فرادرس

ارسال مرجع به اشیاء

وقتی شیئی به عنوان آرگومان به تابعی ارسال می شود، یک کپی از آن ایجاد می شود و در زمان پایان اجـرای تـابع، مخـرب شیء کپی فراخوانی می شود. اگر نخواهید مخرب اجرا شود، می توانید شیء را از طریق مرجع ارسال کنید تـا دیگـر کپـی از آن ایجاد نشود.

✱ خروجی برنامه زیر 1a می باشد:

```
class  c{
          int   a;
   public:   int  b;
             ~c( ) {cout<<'a';}
              void  f(c   &x ) {  --x.b; }
};
int main( ){
  c   ob;  ob.b=2;   ob.f(ob);  cout<<ob.b;
}
```

تذکر: اگر & قبل از x در تابع f را برداریم، خروجی a2a می باشد. ■

✱ خروجی برنامه زیر چیست؟

```
class   c{
          private:  int  x;
         public:  c( ) : x(1) {  }
                  f( )  {cout<<x;}
                  friend  k(c &,int);
};
k(c  &a,int val) { a.x = val;}
int  main( ){
  c  ob;
  k(ob,2);
  ob.f( );
}
```

حل: خروجی ۲ است.■

فرادرس

اشاره گر this

وقتی تابع عضوی فراخوانی می شود، یک آرگومان ضمنی به طور خودکار به آن ارسال می شود که اشاره گری به شیء فراخوانی کننده است. این اشاره گر، this نام دارد. در واقع هر شیء به کمک اشاره گر this به آدرس خود دسترسی پیدا می کند.

تذکر: اشاره گر this ، جزئی از خود شیء نمی باشد.

توابع عضو استاتیک، اشاره گر this ندارند، چون مستقل از هر شیئی از کلاس وجود دارند.

* خروجی چیست؟

```
class  c {
 private:
    int  x;
 public:
   c( int  a )  {x=a; }
   void  f( ) { cout<< x << this->x; }
};
int main( ){
 c  ob(2);
 ob.f( );
}
```

حل: خروجی 22 می باشد. وقتی تابع f در دستور ;(ob.f فراخوانی می شود، به طور خودکار اشاره گر this به تابع f ارسال می شود که به شیء ob اشاره خواهد کرد.

تذکر: دستور ;cout<<x با دستور cout<<this->x معادل است.

تذکر: می توان به جای دستور ;this->x از دستور ; (*this).x نیز استفاده کرد.

فرادرس

دستورات new, delete

توسط new و delete در c++ می توان عمل تخصیص فضا از حافظه Heap و آزاد سازی را انجام داد. (بـه جـای malloc و free در زبان c). کار با دستور new ساده تر از malloc در زبان c می باشد. در این دستور نیازی به استفاده از sizeof برای محاسبه اندازه شی نمی باشد و نیازی به قالب بندی خروجی نمی باشد و می توان مقدار دهی اولیه نیز انجام داد.

✱ ایجاد یک متغیر با مقدار اولیه ۲ در heap و در نهایت از بین بردن آن:

```
int  *a;
a= new   int(2);
delete  a;
```

■

✱ برای تعریف یک آرایه با ۳ خانه در heap، از دستور زیر استفاده می کنیم:

```
int   *x;
x=new  int[3];
```

و برای حذف آرایه، از دستور delete [ ] x; و یا delete [3] x; استفاده می کنیم.

■

✱ پیاده سازی یک پشته به کمک دستور new :

```
class   stack{
          private:
               int   size,top,*a;
         public:
               stack(int  size=10):top(-1)     {a=new int[size]; }
                ~stack( ){delete[ ]  a;}
               void   push(const int& item)  {a[++top]=item;}
               void   pop(int&  item)          {item=a[top--]; }
};
void   main( )
{
  stack  s;
  int  x=4,y,z;
  s.push(x);
  s.push(x+1);
  s.pop(y);
  s.push(x+5);
  s.push(x-2);
  s.pop(z);
  s.pop(z);
  cout<<y+z;
}
```

در ابتدا با ایجاد شیء s، سازنده فراخوانی شده و آرایه ای ۱۰ خانه ای از heap گرفته می شود و top برابر 1- می شود. سپس مقادیر ۴و۵ در پشته push شده و سپس عمل pop انجام شده و عدد ۵ در y قرار می گیرد. بعد از آن دو مقدار ۹ و۲ در پشته push شده و در نهایت دو بار عمل pop انجام شده و عدد ۹ در z قرار خواهد گرفت. بنابراین خروجی یعنی مجموع مقادیر yوz برابر ۱۴ چاپ خواهد شد. در پایان برنامه، مخرب صدا زده شده و آرایه ایجاد شده از بین می رود.

فرادرس

فصل ۳ :

وراثت

یکی از خصوصیات زبانهای شیء گرا، ارث بری می باشد. فرض کنید کلاسی به نام shape داریم که شامل ویژگی های مشترک همه اشکال می باشد. حال می توان از این کلاس، کلاسهایی مانند دایره ، مثلث و یا مستطیل را به ارث برد. یا فرض کنید که کلاسی برای اتومبیل تهیه کرده اید که شامل خصوصیات مشترک اتومبیل ها مانند تعداد چرخ و تعداد سرنشین ها باشد. حال می توان کلاسی برای کامیون تعریف کرد به طوری که از کلاس اتومبیل ارث بری کند و شامل خصوصیات دیگری مانند مقدار بار قابل حمل نیز باشد. یک کلاس می تواند به یکی از ۳ حالت زیر از یک کلاس پایه، مشتق شود:

public	تمام اعضای عمومی کلاس پایه، اعضای عمومی کلاس مشتق خواهند بود و تمام اعضای محافظت شده کلاس پایه نیز به عنوان اعضای محافظت شده کلاس مشتق منظور می شوند.
private	تمام اعضای عمومی و محافظت شده کلاس پایه، به عنوان اعضای اختصاصی کلاس مشتق منظور می شوند.
protected	تمام اعضای عمومی و محافظت شده کلاس پایه به عنوان اعضای محافظت شده کلاس مشتق منظور می شود.

✱ خروجی برنامه زیر ۱۲ است: (کلاس b از کلاس a به صورت public ارث برده است)

```
class  a{
    public:
      int  f (int  n) { return  n;}
};
class  b : public  a{
    public:
      int  g(int  n) { return  n;}
};
void  main( ) {
  b  x ;
 cout<< x.f(1);
 cout<< x.g(2);
}
```

در این مثال چون ارث بری public می باشد، عضو عمومی کلاس a یعنی تابع f ، عضو عمومی کلاس b خواهد بود. بنابراین شیء x که از نوع کلاس b می باشد، می تواند به تابع f نیز دسترسی داشته باشد.

تذکر: اگر ارث بری از نوع protected و یا private بود، در خط x.f(1) خطا ایجاد می شد.

✱ خروجی برنامه زیر چیست؟

```
class   a{
    public:
            f( )  {cout<<'a' ;}
};
class  b: public a{
     public:
           g( ) {cout<<'b'; }
};
class  c : public b{
  public:
          h( ) {cout<<'c'; }
};
main( ) {
   c  x;   x.f( );  x.g( ); x.h( );
   b  y;  y.f( ); y.g( );
}
```

حل: خروجیabcab می باشد. کلاس b از کلاس a ارث برده و کلاس c از کلاس b. بنابراین شیء x که از نوع کلاس c تعریف شده به تمامی توابع f,g,h دسترسی دارد و شیء y که از کلاس b به ارث برده به توابع f,g دسترسی دارد.

تذکر: دستوری مانند y.h( ) نادرست است. چون از طریق شیء y فقط می توان به اعضای عمومی کلاسهای a وb دسترسی داشت و نمی توان به اعضای عمومی کلاس c دسترسی داشت.

■

✱خروجی چیست؟

```
class   a{
   public:
      a( )   { cout<<'1'; }
     ~a( )   { cout<<'2'; }
};
class  b : public  a{
  public:
      b( )   { cout<<'3'; }
     ~b( )   { cout<<'4'; }
};
main( ){
   b   x;
 }
```

حل: خروجی 1342 می باشد. وقتی شیء x از کلاس b ساخته می شود، ابتدا سازنده کلاس a و سپس سازنده کلاس b صدا زده می شوند و ۱۳ به نمایش در می آید. هنگامی که برنامه تمام می شود، ابتدا مخرب کلاس b و سپس مخرب کلاس a صدا زده می شوند.

■

✱خروجی چیست؟

```
class  X{
  public:
        X( )   { cout<<"1";}
       ~X( )  { cout<<"6"; }
};
class  Y : public  X{
  public :
         Y( )   { cout<<"2";}
         ~Y( ) { cout<<"5";}
};
class  Z : public  Y{
     public:
        Z( )    { cout<<"3";}
       ~Z( )    { cout<<"4";}
};
void  main( ){
 Z    ob;
}
```

حل: خروجی 123456 می باشد.

تذکر: اگر شیء ob از کلاس Y باشد، خروجی 1256 می باشد.

■

فرادرس

✱خروجی چیست؟

```
class  c1{
        protected:  int  i;
        public:  c1(int x)  {i=x; cout<<i+1;}
};
class  c2: public  c1{
         public:  c2(int x) : c1(x)  {cout<<i -1; }
};
main( ){
   c2  a(3);
}
```

حل: خروجی ۴۲ است. وقتی شیء a ایجاد می شود، ابتدا سازنده کلاس c1 صدا زده شده و ۴ چاپ می شود. سپس سازنده کلاس c2 صدا زده شده و ۲ چاپ می شود.

تذکر: به تعریف تابع سازنده c2 توجه کنید.

تذکر: اگر شیء a از کلاس c1 تعریف شود، خروجی ۴ می باشد.

■

✱خروجی چیست؟

```
class  c1{
        public:  c1(int x)  {cout<<x;}
};
class  c2: public  c1{
         public:  c2(int  x) : c1(++x)  {cout<<x; }
};
main( ){
   c1  ob1(3);
   c2  ob2(3);
}
```

حل: خروجی 344 می باشد. وقتی شیء ob1 ساخته می شود، سازنده کلاس c1 فراخوانی شده و عدد ۳ چاپ می شود. سپس با ایجاد شیء ob2 ، سازنده کلاس c1 و سپس سازنده کلاس c2 فراخوانی شده و 44 چاپ می شود.

■

✱ خروجی چیست؟

```
class  X{
       public :   X( ){  cout<<'a';}
                  ~X( )  {cout<<'b';}
};
class  Y: public  X{
         public:    Y ( )  {cout<<'c';}
                    ~Y( )  {cout<<'d';}
};
void  main( )  {
```

```
   X  x;
   Y  y;
}
```

حل: خروجی aacdbb می باشد. ابتدا شیء ای از کلاس X ساخته شده و سازنده آن صدا زده شده و a چاپ می شود. سپس شیء ای از کلاس Y ساخته شده و سازنده کلاس X و سپس سازنده کلاس Y صدا زده شده و ac چاپ می شود. بعد از بین رفتن شیء y ، تابع مخرب کلاس Y و سپس مخرب کلاس X صدا زده شده و db چاپ می شود. در نهایت هم با اتمام برنامه، مخرب کلاس X صدا زده شده و b چاپ می شود.

■

فرادرس

✱ خروجی چیست؟

```
class   X{ public :   int  i; };
class   Y : public X{ public:  int  i; };
void  main( ){
  X   x1,x2;Y   y1;
  x1.i=1;
  y1.X::i=2;
  y1.i=3;
  x2=y1;
  cout<<x2.i;
}
```

حل: خروجی ۲ می باشد.

خط ;x1.i=1 ، باعث ذخیره شدن مقدار 1 در عضو i مربوط به شیء x1 از کلاس X می شود.

خط ;y1.X::i=2 ، باعث ذخیره شدن مقدار ۲ در عضو i ارث برده از کلاس X مربوط به شیء y1 می شود.

خط ;y1.i=3 ، باعث ذخیره شدن مقدار ۳ در عضو i مربوط به شیء y1 از کلاس Y می شود.

خط ;x2=y1 ، باعث ذخیره شدن i مربوط به شیء y1 با مقدار 2 در i مربوط به شیء x2 می شود.

تذکر: در این مثال چند متغیر i داشتیم:

۱- یکی مربوط به شیء x1 با مقدار 1 ۲- دو تا مربوط به شیء y1 با مقادیر ۲ و ۳ ■

فرادرس

کلاسهای دارای چند مبنا (وراثت چندگانه)

یک کلاس می تواند مستقیما چند کلاس مبنا را به ارث ببرد. در مثال زیر، کـلاس c3 از دو کـلاس c2,c1 ارث بـرده اسـت.

خروجی این برنامه abc12 است:

```
class   c1{
     int  a;
  public :
     c1(int  x)  { a=x;cout<<'a';}
     int  f( )     {cout<<a;}
};
class  c2{
      int  b;
   public :
     c2(int  x)  { b=x;cout<<'b'; }
     int  g( )     { cout<<b; }
};
class  c3 : public  c1, public  c2{
    public :
       c3 ( int  m , int  n) : c1(m) ,c2(n) { cout<<'c';}
       void  fun ( ) { f ( ) ; g( ) ; }
};
main( ) {
  c3   t(1,2);
  t.fun( );}
```

تذکر: اگر در کلاس c3 به جای public از private و یا protected استفاده می شد، ، خروجی فرقی نمی کرد.

کلاسهای iostream , istream با وراثت یگانه از کلاس پایه ios مشتق شده اند.

کلاس iostream با وراثت چندگانه از هر دو کلاس istream وostream مشتق شده اند.

فرادرس

منتخبی از عناوین آموزشی منتشر شده بر روی فرادرس

برنامه‌نویسی	
عنوان آموزش	مدت زمان تقریبی
مبانی برنامه نویسی - کلیک کنید (+)	۳ ساعت
برنامه نویسی - C کلیک کنید (+)	۱۳ ساعت
آموزش برنامه نویسیC++	۲۰ ساعت
برنامه نویسی کاربردی سی شارپ - کلیک کنید (+)	۱۴ ساعت
آموزش جامع شی گرایی در سی شارپ - کلیک کنید (+)	۱۴ ساعت
برنامه نویسی جاوا - کلیک کنید (+)	۲۳ ساعت
آموزش برنامه نویسی - PHP کلیک کنید (+)	۲۸ ساعت
آموزش فریمورک PHP کدایگنایتر - (CodeIgniter) کلیک کنید (+)	۷ ساعت
آموزش اسکریپت برنامه نویسی - jQuery کلیک کنید (+)	۷ ساعت
آموزش ویژوال بیسیک دات نت - (Visual Basic.NET) کلیک کنید (+)	۱۳ ساعت
آموزش تکمیلی ویژوال بیسیک دات نت - (Visual Basic.NET) کلیک کنید (+)	۱۶ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان - VB.Net کلیک کنید (+)	۴ ساعت
برنامه نویسی اسمال بیسیک یا - Small Basic کلیک کنید (+)	۱۶ ساعت
آموزش ساخت بازی ساده در ویژوال بیسیک - کلیک کنید (+)	۲ ساعت
آموزش کاربردی - SQL Server کلیک کنید (+)	۱۱ ساعت
آموزش آشنایی با LINQ to SQL در - C# کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان سی شارپ - کلیک کنید (+)	۴ ساعت
آموزش برنامه نویسی تحت شبکه با سی شارپ در قالب پروژه - کلیک کنید (+)	۱ ساعت
آموزش Cryptography در دات نت - کلیک کنید (+)	۳ ساعت

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

برنامه‌نویسی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش قفل نرم افزاری در سی شارپ از طریق رجیستری	۴ ساعت
آموزش ساخت اپلیکیشن کتاب و کار با داده‌ها در اندروید - کلیک کنید (+)	۱۳ ساعت
آموزش ارتباط با دیتابیس سمت سرور در اندروید - کلیک کنید (+)	۱۴ ساعت
آموزش ساخت روبات و کنترل آن با اندروید - کلیک کنید (+)	۱۶ ساعت
آموزش ساخت اپلیکیشن دیکشنری صوتی دو زبانه با قابلیت تشخیص صوت کاربر - کلیک کنید (+)	۷ ساعت
آموزش مدیریت بانک اطلاعاتی اوراکل - کلیک کنید (+)	۹ ساعت
آموزش مدیریت بانک اطلاعاتی اوراکل پیشرفته - کلیک کنید (+)	۷ ساعت
آموزش راه اندازی اوراکل ۱۲ cدر لینوکس	۱ ساعت
آموزش دیتاگارد در اوراکل - کلیک کنید (+)	۳ ساعت
برنامه نویسی متلب - کلیک کنید (+)	۹ ساعت
متلب برای علوم و مهندسی - کلیک کنید (+)	۱۴ ساعت
برنامه نویسی متلب پیشرفته - کلیک کنید (+)	۷ ساعت
طراحی رابط های گرافیکی (GUI) در متلب - کلیک کنید (+)	۸ ساعت
آموزش برنامه نویسی R و نرم‌افزار - RStudioکلیک کنید (+)	۷ ساعت
آموزش تکمیلی برنامه نویسی R و نرم‌افزار - RStudio کلیک کنید(+)	۵ ساعت
آموزش برنامه نویسی پایتون ۱ - کلیک کنید (+)	۲۰ ساعت
آموزش برنامه نویسی پایتون ۲ - کلیک کنید (+)	۵ ساعت
آموزش گرافیک کامپیوتری با - OpenGL کلیک کنید(+)	۱۶ ساعت

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

راه اندازی و مدیریت وبسایت‌ها و سرورها	
عنوان فرادرس	**مدت زمان تقریبی**
آموزش برنامه نویسی - PHP کلیک کنید (+)	۲۸ ساعت
آموزش فریمورک PHP کدایگنایتر - (CodeIgniter) کلیک کنید (+)	۷ ساعت
آموزش طراحی وب با - HTML کلیک کنید (+)	۳ ساعت
آموزش طراحی وب با - CSS کلیک کنید (+)	۵ ساعت
آموزش پروژه محور HTML و - CSS کلیک کنید - کلیک کنید (+)	۴ ساعت
آموزش جاوا اسکریپت - (JavaScript) کلیک کنید (+)	۹ ساعت
آموزش کار با - cPanel کلیک کنید (+)	۱ ساعت
آموزش مدیریت هاست با - DirectAdmin کلیک کنید - کلیک کنید (+)	۱ ساعت
راه اندازی سایت و کار با وردپرس - کلیک کنید (+)	۷ ساعت
راه اندازی فروشگاه دیجیتال با وردپرس و - Easy Digital Downloads کلیک کنید (+)	۱ ساعت
آموزش راه اندازی سایت شخصی با وردپرس - کلیک کنید (+)	۱ ساعت
آموزش ترجمه قالب وردپرس - کلیک کنید (+)	۲ ساعت
آموزش راه اندازی سایت خبری با وردپرس - کلیک کنید (+)	۲ ساعت

علوم کامپیوتر	
عنوان آموزش	مدت زمان تقریبی
ساختمان داده‌ها -کلیک کنید (+)	۱۰ ساعت
آموزش ساختمان داده‌ها (مرور – تست کنکور ارشد) - کلیک کنید (+)	۲۰ ساعت
آموزش نظریه زبان‌ها و ماشین‌ها -کلیک کنید (+)	۹ ساعت
آموزش نظریه زبان‌ها و ماشین (مرور – تست کنکور ارشد) - کلیک کنید (+)	۸ ساعت
آموزش سیستم های عامل - کلیک کنید (+)	۱۱ ساعت
آموزش سیستم عامل (مرور اجمالی و تست کنکور) - کلیک کنید (+)	۱۲ ساعت
آموزش پایگاه داده‌ها -کلیک کنید (+)	۸ ساعت
آموزش پایگاه داده‌ها (مرور – تست کنکور ارشد) - کلیک کنید (+)	۵ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی - کلیک کنید (+)	۱۰ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی (مرور – تست کنکور ارشد) - کلیک کنید (+)	۱۲ ساعت
آموزش روش های حل روابط بازگشتی - کلیک کنید (+)	۴ ساعت
آموزش روش تقسیم و حل در طراحی الگوریتم - کلیک کنید (+)	۲ ساعت
آموزش ذخیره و بازیابی اطلاعات - کلیک کنید (+)	۸ ساعت
آموزش ساختمان گسسته با رویکرد حل مساله - کلیک کنید (+)	۱۶ ساعت
آموزش جامع مدارهای منطقی - کلیک کنید (+)	۱۰ ساعت
آموزش معماری کامپیوتر با رویکرد حل مسأله - کلیک کنید (+)	۲۰ ساعت
آموزش ساختمان گسسته (مرور و حل تست های کنکور کارشناسی ارشد) - کلیک کنید (+)	۱۲ ساعت
آموزش طراحی الگوریتم - کلیک کنید (+)	۸ ساعت
آموزش شبکه های کامپیوتری ۱ - کلیک کنید (+)	۱۹ ساعت

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

علوم کامپیوتر (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش نظریه گراف و کاربردها - کلیک کنید (+)	۱۴ ساعت
آموزش نتورک پلاس - (Network+) کلیک کنید (+)	۱۰ ساعت
آموزش مدل سازی UML با نرم‌افزار - Rational Roseکلیک کنید (+)	۳ ساعت
آموزش پردازش ویدئو - کلیک کنید (+)	۳ ساعت
پردازش تصویر در متلب - کلیک کنید (+)	۱۶ ساعت
آموزش پردازش تصویر با - OpenCV کلیک کنید (+)	۱۰ ساعت

هوش مصنوعی	
عنوان آموزش	مدت زمان تقریبی
الگوریتم ژنتیک در متلب - کلیک کنید (+)	۱۴ ساعت
الگوریتم PSO در متلب - کلیک کنید (+)	۱۰ ساعت
الگوریتم ازدحام ذرات (PSO) گسسته باینری - کلیک کنید (+)	۲ ساعت
ترکیب الگوریتم ژنتیک و PSO در متلب - کلیک کنید (+)	۱ ساعت
حل مسأله فروشنده دوره گرد با استفاده از الگوریتم ژنتیک - کلیک کنید (+)	۲ ساعت
الگوریتم مورچگان در متلب - کلیک کنید (+)	۶ ساعت
الگوریتم رقابت استعماری در متلب - کلیک کنید (+)	۱۳ ساعت
طراحی سیستم های فازی عصبی یا ANFIS با استفاده از الگوریتم های فرا ابتکاری و تکاملی - کلیک کنید (+)	۲ ساعت
الگوریتم فرهنگی یا Cultural Algorithm در متلب - کلیک کنید (+)	۲ ساعت

هوش مصنوعی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
شبیه سازی تبرید یا Simulated Annealing در متلب - کلیک کنید (+)	۴ ساعت
جستجوی ممنوع یا Tabu Search در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم کرم شب تاب یا Firefly Algorithm در متلب - کلیک کنید (+)	۱ ساعت
بهینه سازی مبتنی بر جغرافیای زیستی یا BBO در متلب - کلیک کنید (+)	۲ ساعت
جستجوی هارمونی یا Harmony Search در متلب - کلیک کنید (+)	۲ ساعت
کلونی زنبور مصنوعی یا Artificial Bee Colony در متلب - کلیک کنید (+)	۳ ساعت
الگوریتم زنبورها یا Bees Algorithm در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم تکامل تفاضلی - کلیک کنید (+)	۱ ساعت
الگوریتم بهینه سازی علف هرز مهاجم یا IWO در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم بهینه سازی مبتنی بر و یادگیری یا - TLBO کلیک کنید (+)	۱ ساعت
الگوریتم بهینه سازی جهش قورباغه یا SFLA در متلب - کلیک کنید (+)	۴ ساعت
بهینه سازی چند هدفه در متلب - کلیک کنید (+)	۱۹ ساعت
بهینه سازی مقید در متلب - کلیک کنید (+)	۹ ساعت
شبکه های عصبی مصنوعی در متلب - کلیک کنید (+)	۲۸ ساعت
آموزش کاربردی شبکه های عصبی مصنوعی - کلیک کنید (+)	۹ ساعت
آموزش استفاده از شبکه عصبی مصنوعی با نروسولوشن - کلیک کنید (+)	۳ ساعت
شبکه عصبی GMDH در متلب - کلیک کنید (+)	۴ ساعت
شبکه های عصبی گازی به همراه پیاده سازی عملی در متلب - کلیک کنید (+)	۳ ساعت
طبقه بندی و بازشناسی الگو با شبکه های عصبی LVQ در متلب - کلیک کنید (+)	۳ ساعت
آموزش پیاده سازی الگوریتم های تکاملی و فراابتکاری در سی شارپ - کلیک کنید (+)	۸ ساعت

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

آمار و داده کاوی	
عنوان آموزش	مدت زمان تقریبی
گنجینه فرادرس های یادگیری ماشین و داده کاوی - کلیک کنید (+)	۸۸ ساعت
گنجینه فرادرس های محاسبات هوشمند - کلیک کنید (+)	۷۱ ساعت
آموزش یادگیری ماشین - کلیک کنید (+)	۲۴ ساعت
داده کاوی یا Data Mining در متلب - کلیک کنید (+)	۲۴ ساعت
آموزش داده کاوی در - RapidMiner کلیک کنید (+)	۲ ساعت
آموزش وب کاوی - کلیک کنید (+)	۱۷ ساعت
شبکه های عصبی مصنوعی در متلب - کلیک کنید (+)	۲۸ ساعت
آموزش کاربردی شبکه های عصبی مصنوعی - کلیک کنید (+)	۹ ساعت
شبکه عصبی GMDH در متلب - کلیک کنید (+)	۴ ساعت
شبکه های عصبی گازی به همراه پیاده سازی عملی در متلب - کلیک کنید (+)	۳ ساعت
طبقه بندی و بازشناسی الگو با شبکه های عصبی LVQ در متلب - کلیک کنید (+)	۳ ساعت
خوشه بندی با استفاده از الگوریتم های تکاملی و فراابتکاری - کلیک کنید (+)	۳ ساعت
تخمین خطای کلاسیفایر یا - Classifier کلیک کنید (+)	۲ ساعت
انتخاب ویژگی یا - Feature Selection کلیک کنید (+)	۲ ساعت
انتخاب ویژگی با استفاده از الگوریتم های فرا ابتکاری و تکاملی - کلیک کنید (+)	۴ ساعت
کاهش تعداد رنگ تصاویر با استفاده از روش های خوشه بندی هوشمند - کلیک کنید (+)	۱ ساعت
آموزش پردازش سیگنال های واقعی در متلب - کلیک کنید (+)	۴ ساعت
مبانی و کاربردهای راهبرد تلفیق داده یا - Data Fusion کلیک کنید (+)	۹ ساعت
آمار و احتمال مهندسی - کلیک کنید (+)	۱۳ ساعت
آزمون‌های فرض مربوط به میانگین جامعه نرمال در - SPSS کلیک کنید (+)	۳ ساعت

آمار و داده کاوی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش محاسبات آماری در اکسل - کلیک کنید (+)	۲ ساعت
آموزش کنترل کیفیت آماری - کلیک کنید (+)	۵ ساعت
آموزش کنترل کیفیت آماری با - SPSS کلیک کنید (+)	۲ ساعت
آموزش مدل سازی معادلات ساختاری با - Amos کلیک کنید (+)	۷ ساعت
تجزیه و تحلیل اطلاعات با نرم‌افزار - SASکلیک کنید (+)	۴ ساعت

مهندسی برق	
عنوان آموزش	مدت زمان تقریبی
طراحی دیجیتال با استفاده از وریلوگ یا - Verilog کلیک کنید (+)	۷ ساعت
آموزش جامع مدارهای منطقی - کلیک کنید (+)	۱۰ ساعت
آموزش مروری طراحی و پیاده سازی مدارات منطقی - کلیک کنید (+)	۴ ساعت
آموزش میکروکنترلر AVR و نرم‌افزار - CodevisionAVRکلیک کنید (+)	۴ ساعت
آموزش تکمیلی میکروکنترلر AVR و نرم‌افزار - CodevisionAVRکلیک کنید (+)	۴ ساعت
آشنایی باPLC های ساخت شرکت های Omron و - Keyence کلیک کنید (+)	۶ ساعت
میکروکنترلر PIC با کامپایلر - CCS کلیک کنید (+)	۹ ساعت
آموزش تحلیل و طراحی مدارات الکترونیکی با - Proteus کلیک کنید (+)	۳ ساعت
آموزش شبیه سازی و تحلیل مدارهای الکتریکی و الکترونیکی با پی اسپایس (PSpice) -کلیک کنید (+)	۳ ساعت
آموزش مقدماتی - ADS کلیک کنید (+)	۳ ساعت
آموزش تکمیلی آنالیز مدار با نرم‌افزار - ADSکلیک کنید (+)	۲ ساعت

مهندسی برق (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش تحلیل ریاضی مدارات الکتریکی با - OrCAD کلیک کنید (+)	۲ ساعت
آموزش شبیه سازی مدارات الکترونیکی با - Orcad Capture کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی آردوینو (- (Arduinoکلیک کنید (+)	۸ ساعت
آموزش تکمیلی برنامه نویسی آردوینو - (Arduino) کلیک کنید (+)	۷ ساعت
آموزش طراحی برد مدار چاپی به کمک نرم‌افزار - Altium Designer کلیک کنید (+)	۷ ساعت
آموزش مبانی ربات های برنامه پذیر - کلیک کنید (+)	۵ ساعت
آموزش ساخت روبات و کنترل آن با اندروید - کلیک کنید (+)	۱۶ ساعت
آموزش مدارهای الکتریکی ۱ - کلیک کنید (+)	۹ ساعت
آموزش مدارهای الکتریکی ۲ - کلیک کنید (+)	۱۱ ساعت
آموزش سیستم های کنترل خطی - کلیک کنید (+)	۱۰ ساعت
آموزش مکاترونیک کاربردی ۱ - کلیک کنید (+)	۱۳ ساعت
آموزش کامسول (مباحث منتخب) - کلیک کنید (+)	۳ ساعت
آموزش سینماتیک مستقیم و معکوس ربات‌ها -کلیک کنید (+)	۳ ساعت
آموزش تجزیه و تحلیل سیگنال‌ها و سیستم ها - کلیک کنید (+)	۲۷ ساعت
آموزش متلب با نگرش تحلیل آماری، تحلیل سری های زمانی و داده های مکانی - کلیک کنید (+)	۸ ساعت
پردازش سیگنال های دیجیتال با استفاده از نرم‌افزار متلب - کلیک کنید (+)	۴ ساعت
شبیه سازی سیستم با سیمیولینک - کلیک کنید (+)	۴ ساعت
آموزش سیستم های قدرت در سیمولینک و متلب - کلیک کنید (+)	۱۱ ساعت
آنالیز پایداری و کنترل سیستم های قدرت با استفاده از جعبه ابزارهای نرم‌افزار متلب - کلیک کنید (+)	۲ ساعت
آشنایی با SimPowerSystems در شبیه سازی سیستم های قدرت - کلیک کنید (+)	۳ ساعت

مهندسی برق (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
شبیه سازی ماشین های الکتریکی در تولباکس های Simulink و SimPowerSystem در نرم‌افزار متلب - کلیک کنید (+)	۴ ساعت
آموزش الکترونیک قدرت – شبیه سازی در متلب و سیمولینک - کلیک کنید (+)	۸ ساعت
آموزش شبیه سازی عملکرد انواع ماشین های الکتریکی در سیمولینک متلب - کلیک کنید (+)	۱۰ ساعت
برنامه های پاسخگویی بار - کلیک کنید (+)	۴ ساعت
آموزش نرم‌افزار ETAP برای تحلیل سیستم های قدرت - کلیک کنید (+)	۲۱ ساعت
آموزش مقدماتی نرم‌افزار GAMSبرای حل مسائل بازار برق - کلیک کنید (+)	۵ ساعت
آموزش پخش بار اقتصادی (دیسپاچینگ اقتصادی) در GAMS - کلیک کنید (+)	۲ ساعت
کاربرد فازی در سیستم های قدرت - کلیک کنید (+)	۲ ساعت
آموزش نرم‌افزار HFSS - کلیک کنید (+)	۵ ساعت
طراحی آنتن مایکرواستریپ به کمک نرم‌افزار HFSS - کلیک کنید (+)	۱ ساعت
آموزش طراحی و شبیه سازی آنتن های SIW با HFSS - کلیک کنید (+)	۱ ساعت
آموزش بررسی کامل آنتن های مایکرواستریپ و طراحی آن توسط CST - کلیک کنید (+)	۳ ساعت
آموزش تجزیه سیگنال به مولفه های مود ذاتی یا Empirical Mode Decomposition -کلیک کنید (+)	۴۰ دقیقه
نمونه برداری و بازسازی اطلاعات در سیستم های کنترل دیجیتال - کلیک کنید (+)	۳ ساعت
بررسی پاسخ ورودی پله در شناسایی فرآیندهای صنعتی - کلیک کنید (+)	۱ ساعت
مدل سازی و شناسایی سیستم های دینامیکی با استفاده از مدل ARX و شبکه فازی عصبی - ANFIS کلیک کنید (+)	۱ ساعت
طراحی و تنظیم ضرایب کنترل کننده PID با منطق فازی - کلیک کنید (+)	۲ ساعت
آموزش کنترل سیستم چهار تانک - کلیک کنید (+)	۲ ساعت

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

پلی مورفیسم (چند شکلی)

استفاده از یک نام در چندین مورد مربوط به هم و یا به عبارتی استفاده از یک رابط برای یک دسته کلی از اعمال را پلی مورفیسم می نامند. در مثال زیر دو تابع به نام های یکسان fun وجود دارد که کامپایلر با توجه به ورودی آنها، می تواند تشخیص دهد که کدام تابع را باید اجرا کند. خروجی این برنامه 6B می باشد:

```
int     fun (int   a)   { return  ++a;  }
char    fun(char b)   { return  ++b;  }
void  main( ){
  cout<<fun(5);
  cout<<fun('A');
}
```

✱ خروجی چیست؟

```
int      f (int   a)    { return  ++a;  }
char     f (int   b)    { return  ++b;  }
void  main( ){
  cout<<f(1);
  cout<<f(2);
}
```

حل: برنامه خطا دارد. کامپایلر نمی تواند تابع ها را از هم تشخیص دهد. اگر نوع متغیر b برابر char یا float بود، خروجی 266 بود.

■

فرادرس

تابع مجازی

تابعی که در یک کلاس پایه اعلان شده و مجددا در کلاس مشتق تعریف می گردد را تابع مجازی می گویند. مثلا همه اشیاء دایره، مثلث و مستطیل که از کلاس شکل مشتق می شوند، دارای تابعی مثلا به نام draw می باشند که شکل را رسم می کند و برای هر یک از اشیاء متفاوت است. حال برای اینکه برنامه در زمان اجرا خودش تشخیص دهد که کدام تابع مد نظر است، تابع را در کلاس پایه به صورت **virtual** تعریف می کنیم.

تابع مجازی موجب پشتیبانی چندریختی در زمان اجرا می شود.

✱خروجی برنامه زیر چیست؟

```
class  c1{
 public:   virtual  void  f( ){ cout<<'1'; }
};
class  c2 : public  c1{
 public:  void  f( )  {cout<<'2'; }
};
class  c3 : public c1{
  public:  void  f( )  {cout<<'3'; }
};
main( ){
   c1  *p,  ob1;    c2   ob2;    c3   ob3;
   p=&ob1;
   p->f( );
   p=&ob2;
   p->f( );
   p=&ob3;
  p->f( );}
```

حل: خروجی 123 است. اگر کلمه virtual را برداریم، خروجی 111 خواهد بود.

تذکر: اگر p به کلاس c1 یا c2 اشاره کند، برنامه بالا دارای خطا خواهد بود. به عبارتی فقط با اشاره گر به کلاس پایه، می توان به کلاسهای مشتق نیز دسترسی داشت.

✱ خروجی برنامه زیر چیست؟

```
class  c1{
 public:
          virtual  void  f( ){ cout<<'1'; }
};
class  c2 : public  c1{
 public:
          void  f( )  {cout<<'2'; }
```

```
 };
main( ){
   c1   ob1,*p1;
   c2   ob2,*p2;
   p1=&ob1;
   p1->f( );
   p2=&ob2;
   p2->f( );
}
```

حل: خروجی 12 است. اگر کلمه virtual را برداریم، خروجی باز هم 12 خواهد بود. چون هر تابع ( )f را به کمک اشاره گر مربوط به آن کلاس صدا زده ایم.

■

✱خروجی برنامه زیر چیست؟

```
class  c1{ public: virtual void f( ){ cout<<'1'; } };
class  c2 : public c1{ public: void f( ) {cout<<'2'; } };
class c3 : public c1{  public: void f( ) {cout<<'3'; } };
main( ){
   c1   ob1;
   c2   ob2;
   c3   ob3;
   ob1.f( );
   ob2.f( );
   ob3.f( );}
```

حل: خروجی 123 است. اگر کلمه virtual را برداریم، باز هم خروجی 123 خواهد بود. چون به هر تابع با متغیر از نوع کلاس مربوطه دسترسی داشته ایم.

صفت مجازی موروثی است. یعنی اگر کلاس c3 از کلاس c2 در مثال بالا مشتق شود، باز هم خروجی فرقی نمی کند.

فرادرس

فراخوانی تابع مجازی از طریق مرجع کلاس پایه

می توان یک تابع مجازی را از طریق مرجع کلاس پایه صدا زد. در مثال زیر تابع مجازی f( ) ، توسط مرجع p از کلاس پایه صدا زده شده است. خروجی برنامه 12 است:

```
class c1{  public:  virtual void f( ){ cout<<'1'; } };
class  c2 : public c1{ public: void f( ) {cout<<'2'; } };
void  k(c1 &p) {
    p.f( );
 }
main( ){
   c1 ob1;
   c2  ob2;
   k(ob1);
   k(ob2);
}
```

تذکر: اگر متغیر مرجع p در تعریف تابع k از کلاس c2 باشد، از خط k(ob1) در برنامه، خطا گرفته می شود.

تابع مجازی محض

تابع مجازی محض، نوعی تابع مجازی است که در کلاس پایه تعریف نشده باشد. در این حالت هر کلاس مشتق باید تعریف خاص خودش را ارائه کند.

✱تابع f در برنامه زیر، یک تابع مجازی محض است. این تابع در کلاس پایه تعریف نشده و کلاسهای cdec و chex که از این کلاس مشتق شده اند، تعریف خاص خودشان از این تابع را دارند.

```
class  number{
   protected:  int  x;
   public:   void    set(int  i ){ x=i;}
             virtual  void  f( )=0;
};
class cdec : public number{ public:  void f( ) {cout<<x; }  };
class chex : public number{ public: void f( ) {cout<<hex<<x; } };
main( ){
   cdec  d; chex  h;
   d.set(5);
   d.f( );
   h.set(10);
   h.f( );
}
```

خروجی برنامه 5A می باشد. (متغیر x از نوع protected اعلان شده تا بتوان از آن در کلاسهای مشتق استفاده کرد.)

✱ خروجی چیست؟

```
class   X{          public : void  f( ) {cout<<"a";}};
class   Y : public X{public : void  f( ){ cout<<"b";}};
void  main( ){
  X  x;  Y  y;
  x.f( );
  y.X::f( );
  y.f( );
  y.Y::f( );}
```

حل: خروجی aabb می باشد. (تذکر: دستوری مثل x.Y::f( ); نادرست است.)

■

کلاس انتزاعی

کلاسی که حداقل دارای یک تابع مجازی محض باشد را کلاس انتزاعی می گویند. هیچ شیئی از این نوع کلاس نمی توان ایجاد کرد. البته می توان اشاره گر و مرجع به این کلاس ایجاد کرد.

✱ خروجی برنامه زیر چیست؟

```
class  c1{ public:  virtual  void  f( )=0; };
class  c2 : public  c1{    public: void  f( )  {cout<<'a'; } };
main( ){
   c1   ob1;  c2  ob2;
   ob2.f( );
}
```

حل: کلاس c1 انتزاعی می باشد، بنابراین خط c1 ob1; ، که از کلاس مجازی، شیء ای تعریف شده، خطا دارد.

■

فرادرس

فصل ۵:

سربارگذاری عملگرها (تعریف مجدد عملگرها)

سربارگذاری یک عملگر، یعنی استفاده دیگری از عملگر به جز کار اصلی اش، مثلا استفاده از عملگر == برای تعیین برابری دو نقطه در مختصات دو بعدی یا استفاده از عملگر + برای جمع دو ماتریس و ....

برای سربارگذاری از تابع operator استفاده می کنیم.

✱ سربار گذاری عملگر == برای کلاس مختصات

می خواهیم تعریف جدیدی برای عملگر == ارائه دهیم تا بتواند مثلا دو عضو (2,2) و (2,3) را با هم مقایسه کند. اگر درایه اول هر دو با هم برابر بود و همچنین درایه دوم آنها نیز با هم برابر بود، آن دو با هم برابر می باشند. به عبارتی عمل مساوی بودن در مختصات دو بعدی را بررسی می کنیم.

```
class  c{
    int  x,y;
public:
   c( int  i , int  j)   { x=i,y=j;}
   int  operator == (c  k);
};
int  c::operator == (c  k){
  if( x==k.x  &&  y==k.y )   return  1;   else  return 0;
}
main ( ){
 c    ob1(2,2);
 c    ob2(2,3);
 if (ob1==ob2)
       cout<<"equal";
 else
      cout<< " not equal";
}
```

خروجی این برنامه not equal است. در واقع عملگر == وقتی دو شیء را مساوی فرض می کند که هر دو عنصر آنها با یکدیگر برابر باشند.

✱ تعریف مجدد عملگر جمع (می خواهیم عملگر + را طوری تعریف کنیم که بتواند دو عنصر را به صورت زیر جمع کند:)

(1,2) + (3,4)=(1+3,2+4)=(4,6)

```
class   c{
     int  a; int  b;
  public:
     c( )  { }
     c( int  x , int  y)   {  a=x;  b=y; }
     void  f( )            { cout<<a<<b; }
     c   operator + (c  k);
};
c    c::operator + (c  k){
    c   t;
    t.a = k.a+a;
    t.b = k.b+b;
    return  t;
}
main ( ){
    c   ob1(1,2);
    c   ob2(3,4);
    ob1=ob1+ob2;
    ob1.f( );
}
```

خروجی این برنامه 46 است. در ابتدا دو شیء ob1,ob2 ساخته شده و توسط دستور ob1=ob1+ob2; ،عمل جمع با تعریف جدید انجام می شود. (تذکر: تابع operator+، دارای یک پارامتر است، چون پارامتر دیگر به صورت ضمنی از طریق اشاره گر this به تابع ارسال می شود.)

*** تعریف مجدد عملگر ++ پیشوندی:**

می خواهیم عملگر ++ را طوری تعریف کنیم که بتواند روی یک جفت عنصر کار کند:

++(2,5)=(3,6)

```
class   c{
     int  a;  int  b;
  public:
     c( )  { }
     c( int   x , int   y)   {a=x; b=y; }
     void  f( )  {cout<<a<<b;}
     c   operator++(  );
};
c    c::operator ++ ( ){
    a++;  b++;
   return   *this;
}
main ( ){
```

```
   c   ob1(1,2);
   ++ob1;
   ob1.f( );
}
```

خروجی ۲۳ است.(برای تعریف عملگر ++ پسوندی، تابع را به صورت زیر تعریف می کنیم:

```
c    c::operator ++ (int  k )
```

تعریف مجدد عملگرها به کمک تابع دوست

عملگرها را می توان به کمک توابعی که عضو کلاس نمی باشند، مجددا تعریف کرد. البته این توابع باید دوست این کلاس باشند. توجه کنید که چون توابع دوست فاقد اشاره گر this می باشند، عملوندها صریحا ارسال می شوند. بنابراین تابعی که عملگر با یک عملوند را تعریف می کند، یک پارامتر دارد و تابعی که عملگری با دو عملوند را تعریف می کند، دارای دو پارامتر است.

با استفاده از این توابع دوست، امکان تعریف عملگرهای <- , [ ] , ( ),= نمی باشد.

برای تعریف مجدد عملگرهای -- , ++ ، باید از پارامترهای مرجع استفاده کرد.

✱ تعریف مجدد عملگر + توسط تابع دوست :

```
class   c{
          int  a;  int  b;
 public: c( )  { }
          c( int  x , int  y)   {  a=x;  b=y; }
          void  f( )            { cout<<a<<b; }
          friend   c   operator + (c   m, int  k);
};
c    operator + (c   m, int   k){
   c   t;
   t.a =  m.a+k;
   t.b =  m.b+k;
   return  t;
}
main ( ){
  c   ob1(1,2);  c   ob2;
  ob2=ob1+2;
  ob2.f( );
}
```

توسط عملگر جمع، به هر یک از مقادیر ob1 ، دو واحد اضافه می شود :

(1+2,2+2)=(3,4)

بنابراین خروجی 34 می باشد.

تعریف مجدد عملگرهای >> و <<

در زبان C++ می توان با استفاده از عملگرهـای << و >> اعمـال ورودی و خروجـی را بـر روی انـواع اولیـه انجـام داد. ایـن عملگرها را می توان مجددا تعریف کرد تا اشیایی از کلاس ها را پردازش کنند.

✱ سربارگذاری عملگر درج کننده >> : (تذکر: درج کننده باید به صورت تابع friend تعریف شود.)

```
class  c{
      int  x;
 public:  c(int  y) { x=y;}
          friend   ostream   &operator << (ostream   &s , c  ob);
};
ostream   &operator << (ostream   &s , c   k) { s <<k.x + 4;  return   s;}
main( ){
  c   ob(2);
  cout <<ob;
}
```

خروجی برنامه بالا ۶ می باشد.

عملگرهای مقابل نمی توانند مجددا تعریف شوند:

.* :: ? sizeof

فرادرس

فصل ۶:

قالب ها

توسط قالب می توان مجموعه کاملی از توابع مرتبط به هم(توابع همنام) را که توابع قالب نام دارند، یا مجموعه ای از کلاسهای مرتبط به هم را که کلاسهای قالب نام دارند، مشخص کرد.

تابع قالب

تابع کلی یا تابع قالب با کلمه کلیدی template ایجاد می شود. این نوع تابع مجموعه ای از اعمال را تعریف می کند که بر روی انواع مختلفی از داده ها انجام می شوند. نوع داده ای که تابع باید بر روی آن عمل کند، به عنوان آرگومان به آن ارسال می شود. به عنوان مثال الگوریتم مرتب سازی سریع برای آرایه صحیح و یا آرایه اعشاری، با یک روش عمل می کند و با ایجاد یک تابع کلی، الگوریتم را مستقل از نوع داده تعریف می کنیم. در این حالت کامپایلر کد مناسب را در زمان اجرا تولید می کند. شکل کلی تعریف تابع قالب به صورت زیر می باشد:

template <class نوع> نوع برگشتی نام تابع (لیست پارامترها)

{ بدنه تابع }

تابع کلی f، محتویات دو متغیر را با هم تعویض می کند. (چون شیوه تعویض محتویات دو متغیر به نوع آنها بستگی ندارد، از تابع استفاده می کنیم)

```
template <class X> void f(X &a,X &b)
{ X t; temp=a; a=b; b=temp; }
```

در اینجا X یک نوع کلی است. اگر در تابع main( ) تابع f( ) یکبار با نوع صحیح و یکبار با نوع اعشاری فراخوانی شود، کامپایلر دو نسخه از این تابع را ایجاد می کند. تابع f( ) را می توان به صورت زیر نیز نوشت:

```
template <class X>
void f(X &a,X &b)
  { X t; temp=a; a=b; b=temp; }
```

تذکر: در این حالت، نباید دستوری بین دو خط اول نوشت.

■

✱ روش پیدا کردن کوچکترین عنصر بین سه عنصر، به نوع آنها بستگی دارد. بنابراین تابع **main( )** را به صورت تابع قالب می نویسیم:

```
template <class T>
T min( T x, T y, T z ){
  T m = x;
  if ( y < m ) m = y;
  if ( z < m ) m = z;
  return m;
}
```

```
int  main( ){
  int     a,b, c;
  cin >> a >> b >> c;
  cout << min(a,b,c);
  float    m,n,p;
  cin >>m >>n >>p;
  cout << min(m,n,p);
 }
```

حل: این برنامه ۳ عدد را از ورودی خوانده و کوچکترین آنها را چاپ می کند.(یکبار اعداد صحیح خوانده و یکبار اعداد اعشاری)

■

فرادرس

*عملکرد برنامه زیر چیست؟

```
template  < class   T>
void   p ( const  T   *x ,  const  int   size ){
  int  i;
  for ( i = 0; i <size ; i++ )
    cout <<x[i];
}
int   main( ){
  const  int   n=4,m=2;
  int    a[n] = {3,2,5,8};
  float  b[m] = {1.5,2.5};
  p(a,n);
  p(b,m);
}
```

حل: برنامه داده شده محتویات آرایه با نوع int و یا نوع float را چاپ می کند.

■

تابعی با دو نوع کلی

می توان در دستور template بیش از یک نوع داده را تعریف کرد. در مثال زیر،AوB دو نوع کلی می باشند که در فراخوانی های مختلف، مقادیر متفاوتی می گیرند:

```
template   <class  A , class   B>
void   f(A  x,B   y){
   cout<<x<<y;
}
main( ){
   f("ok",2.5);
   f(4L,15);
}
```

در فراخوانی اول تابع f( ) ، انواع char * و float به جای نوع های A وB قرار گرفتند و در فراخوانی دوم تابع f( ) ، انواع long و int به جای نوع های A وB قرار گرفتند.

کلاس های کلی

علاوه بر توابع کلی، می توان کلاس ها را نیز به صورت کلی تعریف کرد. در این نوع کلاس، همه الگوریتمهایی که توسط کلاس استفاده می شود، تعریف می شود اما نوع واقعی که باید مورد استفاده قرار بگیرد، هنگامی که اشیایی از آن کلاس ایجاد می شوند، به صورت پارامتر مشخص می گردد. وقتی کلاس کلی ایجاد کنید، عمل مورد نظر را می توان بر روی هر نوع داده ای انجام داد. بر اساس نوع تعیین شده در زمان ایجاد شیء، کامپایلر کد مربوطه را تولید می کند.

دانلود رایگان مجموعه کتب کارشناسی و کنکور ارشد کامپیوتر http://faradars.org/computer-engineering-exam

شیوه اعلان کلاس کلی:

```
template  <class  نوع>  class  نام کلاس {
....
}
```

که نوع در هنگام ایجاد شیء مشخص می شود.

✱ خروجی برنامه زیر A3 می باشد:

```
template <class T>
class c{
   private:
        T  x;
   public:
        c(T  n);
        void  f( );
};
template <class T>
c<T>::c(T n) {  x=n;}
template <class T>
void c<T>::f( ){ cout<<x;}
int main( ){
  c<char> x('A');
  x.f( );
  c<int> y(3);
  y.f( );
}
```

- قالب کلاس می تواند از قالب کلاس دیگر مشتق شود.
- قالب کلاس می تواند از کلاس غیر قالب مشتق شود.
- کلاس قالب می تواند از قالب کلاس مشتق شود.
- کلاس غیر قالب می تواند از قالب کلاس مشتق شود.

فرادرس

فصل ۷:

فایل

در زبان C++، فایل به صورت مجموعه ای از بایت ها در نظر گرفته شده که انتهای هر فایل با کاراکتر انتهای فایل مشخص می شود. با باز کردن یک فایل، شیئی ایجاد شده و یک جریان(stream) به آن مربوط می شود.

کلاس های فایل

برای ورودی-خروجی فایل در C++ ، باید فایل <fstream.h> را به برنامه ضمیمه کرد. این فایل کلاسهای زیر را تعریف می کند:

۱- ifstream (برای خواندن از فایل)

۲- ofstream (برای نوشتن در فایل)

۳- fstream (برای خواندن و نوشتن در فایل)

تذکر: کلاس ifstream از کلاس istream ، کلاس ofstream از کلاس ostream و کلاس fstream از کلاس iostream مشتق می شود.

تذکر: کلاسهای istream ، ostream و iostream ، از ios مشتق می شوند.

فرادرس

بازکردن فایل

برای باز کردن فایل، حالتهای زیر وجود دارد:

ios:in	باز کردن فایل متنی در حالت ورودی
ios:out	باز کردن فایل متنی در حالت خروجی
ios:app	باز کردن فایل متنی در حالت افزودن به انتها
ios:ate	باز کردن فایل در حالت خروجی(در هر جای فایل می توان نوشت.)
ios:binary	باز کردن فایل باینری
ios:trunk	اگر فایل موجود باشد، محتویات قبلی آن از بین می رود.
ios:nocreate	اگر فایل موجود نباشد، عمل باز کردن با شکست مواجه می شود.
ios:noreplace	اگر فایل موجود باشد، عمل باز کردن با شکست مواجه می شود.

*بازکردن فایل a.dat در حالت خروجی:

```
ofstream   f;   f.open("a.dat", ios::out);
```

که می توان از دستورهای زیر نیز استفاده کرد:

```
ofstream    f;  f.open("a");
```

البته کلاس ofstream دارای سازنده ای است که به طور خودکار فایل را باز می کند و نیازی به استفاده از دستور open نمی باشد، بنابراین دستور را می توان ساده تر نیز نوشت:

```
ofstream   f("a");
```

■

فرادرس

دستورات کار با فایل

get	خواندن کاراکتر
put	نوشتن کاراکتر
read	خواندن رکورد
write	نوشتن رکورد
ignore	نادیده گرفتن کاراکترهای خوانده شده از یک جریان ورودی
seekg	انتقال اشاره گر خواندن به رکورد دلخواه
seekp	انتقال اشاره گر نوشتن به رکورد دلخواه
tellg	تعیین مکان اشاره گر خواندن
tellp	تعیین مکان اشاره گر نوشتن
rdstate	بدست آوردن اطلاعات وضعیت I/O

تذکر: هر فایل دارای دو اشاره گر get و put می باشد. اشاره گر get مشخص می کند که عملیات خواندن بعدی باید از آنجا شروع شود و اشاره گر put نشان می دهد که عملیات نوشتن بعدی از آنجا باید انجام شود.

تابع seekg اشاره گر get و تابع seekp اشاره گر put را حرکت می دهد. این توابع دارای دو ورودی هستند که اولین ورودی تعداد بایتهای حرکت را نشان می دهد و دومین ورودی محل شروع را نشان می دهد که مقادیر beg وcur و end می باشند.(ابتدا، جاری، انتها)

تابع tellg محل اشاره گر get و تابع tellp محل اشاره گر put را تعیین می کند.

✱ نحوه کار با توابع seekg و seekp :

```
fstream   f("test",ios::in | ios:out);
f.seekg(0);
f.seekp(20,ios::cur);
```

دستور اول فایل test را به صورت ورودی و خروجی باز می کند و به جریان f نسبت می دهد. دستور دوم اشاره گر خواندن فایل را به ابتدای فایل انتقال می دهد و دستور سوم، اشاره گر فایل را با شروع از موقعیت فعلی، ۲۰ بایت به طرف انتهای فایل حرکت می دهد.

✱ خواندن خط به خط محتویات یک فایل متنی و نمایش آن در صفحه نمایش:

(یک فایل متنی به عنوان آرگومان برنامه به آن داده می شود)

```
main( int  argc , char  *argv[ ] ){
  char  s[255];
  ifstream   f(argv[1] );
  f.getline(s,255);
```

```
 while(!f.eof( ) ) {
      cout<<s<<endl;
      f.getline(s,255);
    }
 f.close( )
}
```

✱ توسط دستورات زیر یک فایل باینری باز شده و عدد ۳ در آن نوشته می شود:

```
int   a=3;
ofstream   f("test", ios::out |  ios::binary);
f.write((char *) &a, sizeof(int));
```

✱دستورات زیر یک فایل باینری را باز کرده و عمل خواندن را انجام می دهد:

```
int  a;
ifstream   f("test",ios :: in | ios :: binary);
f.read( (*char) &a,sizeof(int) );
```

✱گرفتن اطلاعات دانشجویان و ذخیره آن در فایل:

```
void   enter( ) {
    struct   student {char   name[11];char   family[21]; int  stno;}s;
    ofstream    fp("st.dat", ios::out  |  ios :: binary);
    while(1) {
         cin.getline(s.name,10);
         if (strln(s.name) ==0)  break;
         cin.getline(s.family,20);
         cin >> s.stno;
         cin.get( );
         fp.seekp( sizeof(struct  student) *s.stno , ios ::beg );
         fp.write( (char  *) &s , sizeof(struct  student) );
       }
   fp.close( );
}
```

فرادرس

فصل ۸ :

آزمون

۱- خروجی برنامه زیر کدام است؟

```
class  c{
 private:
      int  a;
 public:
     int  f( ) { a=5; cout<<a;}
 };
void main( ){
 c  x;
 x.f( );
}
```

پاسخ : جواب 5 است.

۲- خروجی برنامه زیر کدام است؟

```
class  a{
   private:
      int  x;
   public :
       a(int=2);
       b(int=5);
       void  p( ) const;
};
a::a(int n)   {x=n;}
a::b(int m)   {x=m;cout <<"a"<<x;  }
void   a::p( )  const{cout<<"b"<<x;  }
void   main( ){
 a  f;
 f.p( );
 cout<<"c";
 f.b( ) ;
}
```

پاسخ : جواب b2ca5 است.

۳- خروجی چیست؟

```
class  c{
```

```
    int  a;
 public:
   c( )        { a=1;}
   c(int  x)  { a=x;}
   void  f( )  { cout<<a; }
};
main( ){
  c  o1;
  o1.f( );
  c  o2(3);
  o2.f();
}
```

پاسخ : جواب ۱۳ است.

۴- خروجی چیست؟

```
class  c{
    int  a;  int  b;
 public:
   c(int  x=1,int  y=2)  { a=x;b=y;}
   void  f( )  { cout<<a<<b; }
};
main( ){
  c  o1;
  o1.f( );
  c  o2(3);
  o2.f( );
}
```

پاسخ : جواب ۱۲۳۲ است.

۵- خروجی برنامه زیر کدام است؟

```
class  a{
   public:
      static  int  c;
       a( )   { ++c; }
      ~a( )  {cout<<"a"; }
};
int  a::c=0;
void  main( ){
```

```
   a   w,x;
   cout<<w.c;
   { a   w,x,y,z;  cout<<w.c;  }
   cout<<w.c;
   a   y;
   cout<<w.c;
}
```

پاسخ : جواب 26aaaa67aaa است.

۶- خروجی برنامه زیر کدام است؟

```
class   a{
   public:
       static  int  c;
       a( )    {++c; }
      ~a( )  {--c; }
};
int  a::c=0;
void main( ){
    a   w,x;
    cout<<w.c;
   { a  w,x,y,z;    cout<<w.c;  }
   cout<<w.c;
   a  y;
   cout<<w.c;
}
```

پاسخ : جواب 2623 است.

۷- خروجی برنامه زیر کدام است؟

```
class   t{
    float  a,b;
 public :
    t(float  x=2 , float y=3) : a(x) , b(y)    { cout<<"a";  }
    void      f( )    { --a; ++b; }
    double  k( )   { return  a+b; }
    void      p( )   { cout<<a<<" "<<b; }
};
```

```
void main( ){
  t  d;   cout <<d.k( );
  d.f( );
  d.p( );
}
```

پاسخ : جواب 4 51 a است.

۸- خروجی برنامه زیر کدام است؟

```
class   a{
     int    x;
  public:
           int   f (int n) { return  --n;}
};
class  b : public  a{
     int  y;
  public:
          int   g(int  n) { return ++n;}
};
void  main( ) {
   b  t ;
  cout<< t.f(2);
  cout<< t.g(4);
}
```

پاسخ : جواب ۱۵ است.

۹- خروجی برنامه زیر کدام است؟

```
class  a{
      int   x;
  public:
          int   f (int n)  { return  --n; }
};
class  b : private  a{
    int    y;
  public:
           int  g(int  n) { return   ++n; }
};
void main( ){
```

```
  b  t;
  cout<< t.f(2);
  cout<< t.g(4);
}
```

پاسخ : خطا دارد.

۱۰- خروجی برنامه زیر کدام است؟

```
class  a{
    protected:  static  int   x;
};
int   a::x=3;
class   b : public  a{
  public :   void  f(int n)  {cout<<x+n;}
};
void   main( ){
  b   t;
  t.f(2);
}
```

پاسخ : جواب ۵ است.

۱۱- کدام خط(ها) خطا دارد؟

```
class   a{
  protected:  int  x;
  public:
          void   f(int  i) {x=i;}
};
class  b : private  a{
    int   y;
    public:
         void  g( ) { y=i+1;}     //1
};
class  c : public  b{
    int  z;
 public:
          void  h( )   {z=i+2;}      //2
};
```

```
main( ){
  b   ob1;
  c   ob2;
  ob1.f(1);          //3
  ob2.f(5);          //4
}
```

پاسخ : هر چهار خط.

۱۲- خروجی چیست؟

```
class  a{
   public:
       a( )    { cout<<"A"; }
      ~a( )   { cout<<"B"; }
};
class  b : public  a{
 public:
       b( )    { cout<<"C"; }
      ~b( )   { cout<<"D"; }
};
main( ){
   b   ob;
 }
```

پاسخ : جواب ACDB است.

۱۳- خروجی چیست؟

```
class  a{
  protected:   int  i;
  public:      a(int  x )    {i=x; cout<<"A"; }
 };
class  b : public  a{
      int  j;
  public:
       b(int x, int y ) : a(y)   { j=x ; cout<<"C"; }
      ~b( )          { cout<<"D"; }
       void  f( )  { cout<<i<<j;}
};
main( ){
```

```
   b  ob(1,2);
  ob.f( );
}
```

پاسخ : جواب AC21D است.

۱۴- خروجی برنامه زیر کدام است؟

```
class  a{
      int  i;
   public:  a(int  n);
      ~a( );
       void  f1(int  n)  { i=n; }
       int   f2( )        { return  i; }
};
a::a(int  n)       {  i=n;   cout<<'b';}
a::~a( )           { cout<<'a'; }
void  f3(a  ob)  { cout<<ob.f2( ); }
void main( ){
   a  x(2);
   f3(x);
   cout<<x.f2( );
 }
```

پاسخ : جواب b2a2a است.

۱۵- خروجی کدام است؟

```
class  c1{
    int  a;
 public :  c1(int  x)  { a=x;}
           int  f( )    { return  a; }
};

class  c2{
     int  b;
  public :   c2(int  x)  { b=x; }
             int  g( )    {  return  b; }
};
```

```
class  c3 : public  c1, public  c2{
      int  c;
   public :    c3 ( int  x , int y , int z) : c1(z) ,c2(y) { c=x;}
               void  p ( ) { cout <<f ( ) << g( ) << c;  }
};

main( ) {
 c3  ob(5,6,7);
 ob.p( );
}
```

پاسخ : جواب 765 است.

۱۶- خروجی کدام است؟

```
class   a{
 protected :
    int  i;
 public :
    a(int  x)    { i=x; cout<<"A";}
    ~a( )        {cout<<"B";}
    };

class  b{
  protected:   int  j;
  public :    b(int  x)   { j=x; cout<<"C";}
              ~b( )  { cout<<"D";}
};

class  c : public  a, public  b{
     int  k;
   public :
          c ( int  x , int y , int z) : a(y) , b(z) { k=x; cout<<"E";}
          ~c ( )          {  cout<<"F";}
            void  p ( )   {  cout <<i << j << k;  }
};

main( ) {
 c   ob(1,2,3);
 ob.p( );
```

```
}
```

پاسخ : جواب ACE231FDB است.

۱۷- خروجی برنامه زیر کدام است؟

```
class a{
    int x;
    int y;
  public:
    a(int i, int j) {x=i; y=j;}
    int f()  { return x;}
    int g( ) { return y;}
};

int main( ){
 int i;
 a ob[3]={a(1,2) ,a(3,4) , a(5,6) };
 int s=0;
 for(i=0 ; i<3 ;i++)
    s+=ob[i].f( ) + ob[i].g( );
 cout<<s;
}
```

پاسخ : جواب ۲۱ است.

۱۸- خروجی چیست؟

```
class c{
     int a;
   public:
     c(int x) { a=x;}
     friend int f(c) ;
 };
int f(c z)
 { cout<<z.a; }

main( ){
  c k(1);
  f(k);
```

```
}
```

پاسخ : جواب ۱ است.

۱۹- خروجی برنامه زیر کدام است؟

```
class a{
  int x;
  public:
   f( ) { cout<<"a"; }
   friend void g() { cout<<"b";}
};

void main( ){
  a t;
  t.f( );
  g( );
}
```

پاسخ : جواب ab است.

۲۰- خروجی برنامه زیر کدام است؟

```
class a {
   private: int x;
  public:
      a( ) : x(5)   {cout<<"a";}
      void      print( ) const { cout << x ; }
      friend   void   f( a &, int );
};
void f( a &c , int val )
 { c.x = val; }
int main( ){
  a ob;
  ob.print( );
  f ( ob, 6 );
  ob.print( );
}
```

پاسخ : جواب a56 است.

۲۱- خروجی چیست؟

```
class c{
    int a;
 public:
   c(int x) { a=x;}
   friend class d;
};
class d{
 public :
    int f(c z) { cout<<z.a;}
};
main( ){
  c m(1);
  d n;
  n.f(m);
}
```

پاسخ : جواب ۱ است.

۲۲- خروجی چیست؟

```
class c{
     static int a;
     int b;
  public:    f (int x, int y) { a=x;b=y;}
             void g( ) { cout<<a<<b;}
};
int c::a;
main( ){
  c o1,o2;
  o1.f(1,2);
  o1.g( );
  o2.f(3,4);
  o2.g( );
  o1.g( );
}
```

پاسخ : جواب 123432 است.

۲۳- خروجی چیست؟

```
class c{
     public:
            static int  a;
};
int c::a;
main( ){
 c ::a=5;
 cout<<c::a;
 c  x;
 cout<<x.a;
}
```

پاسخ : جواب ۵۵ است.

۲۴- خروجی چیست؟

```
class  a{
  public:   static int  c;
            a( )  {c++;}
           ~a( ) { c--;}
};
int a::c;
void f( )
{
   a b;
   cout<<a::c;
}
main( ){
   a  ob1,ob2;
   cout<<a::c;
   a  ob3;
   cout<<a::c;
   f( );
   cout<<a::c;
}
```

پاسخ : جواب 2343 است.

۲۵- کدام خط خطا دارد؟

```
class c{
     static   int   a; int  b;
 public:
     int  d;
     static  f( )     //1
      {
         a=2;        //2
         b=3;        //3
         d=4;        //4
       }
};
int  c::a=5;
main( ){
  c  o;  o.f( );          //5
}
```

پاسخ : خط ۳و۴

۲۶- خروجی چیست؟

```
class  a{
     static  int  r;
    public:
           static  int  f( );
};
int  a:: r=1;
int  a::f( )
  { cout<<r; }
int  main( ) {
   a  ob1;
   a::f( );
   ob1.f( );
  }
```

پاسخ : جواب 11 است.

۲۷- خروجی چیست؟

```
class  a{
```

```
  public:
     static  int  r;
     static  int  f(int  x )  { r = x; }
};
int    a:: r;
int  main( ){
   a::f(5);
  cout <<a::r;
  a   ob1;
  cout <<a::r;
}
```

پاسخ : جواب 55 است.

۲۸- خروجی چیست؟

```
class   c {
     int      x;
     const  int  c;
  public:
     c( int  c=0, int  i=1 );
     void   f( )  { x = c; }
     void   p( )  const;
};
c::c (int  c, int  i ) : x(c), c(i)
   {  x=c;   c=i;  }
void  c::p( ) const
  { cout << x<< c ; }
int  main( ) {
  c   a(2,5);
  for ( int  j = 0; j <2; j++ )
   {
     a.f();
     cout << j ;
     a.p( );
  }
}
```

پاسخ : جواب 055155 است.

۲۹- کدام خط(ها)، خطا دارد؟

```
class  c{
  public:
```

```
   int  a;
   c( ) {cout<<"ok";}       //1
};
main( ) {
 const  c  o ;              //2
 o.a=5;                     //3
}
```

پاسخ : جواب 3 است.

۳۰- چند خطا وجود دارد؟

```
class  a{
    int      data;
    static  int  count;
 public:
    a(int  y=10) : data(y)   {  cout<<"a"; }
    int   f( ) const    {  return  ++data;  }
    static   int   g( )   {  cout<<data;  return  count; }
};
```

پاسخ : جواب 2 است.

۳۱- کدام درست است؟

```
class  a{
  private:  int    x;
            static  int  c;
  public:
          a(int  y=5)   { x=++y;}
          int  f( )       { return ++x; }
          static  int  g( )  const   {cout<<x; }
};
main( ) {
  a   t;
  cout <<t.f( );
  t.g( );
}
```

پاسخ : اگر static قبل از تابع g را برداریم، خروجی دارد.

۳۲- خروجی چیست؟

```
class  a{
      int  x;
public:
     a( )       { x=0;}
     a(int  y)  { x=y;}
     int  f( )   { return  x;}
};
int  main( ) {
  a    z[4]={1,2,3,4};
  a   *p;
  int  i;
  p= z;
  for(i=0 ; i<4 ; i++)
     {
      cout<<p->f( );
      p++;
     }
}
```

پاسخ : جواب 1234 است.

۳۳- خروجی برنامه زیر کدام است؟

```
class  count{
  public:
         int  x;
         void  p( ) { cout<<x++;}
  };

void   main( ) {
 count   c,*k=&c;
 c.x=1;
 c.p( );
 k->x=3;
  k->p( );
 (*k).x=4;
 (*k).p( );
```

```
}
```

پاسخ : جواب 134 است.

۳۴- خروجی کدام است؟

```
class a{
 public:
        int x;
        a( int y) { x=y;}
};

int main( ){
 a   ob=5;
 int *p;
 p=&ob.x;
 cout<<*p;
}
```

پاسخ : جواب5 است.

۳۵- خروجی برنامه زیر کدام است؟

```
class a{
 public :
          int b;
          void f( ) { cout<<b++;}
};
void main( ){
 a  c; c.b=5;  c.f( ); c.f( );
 a t;  t=c;  t.f( );
 a *p;
 p=&c;
 p->b=3;
 p->f( );
 (*p).b=8;
 (*p).f( );
}
```

پاسخ : جواب 56738 است.

۳۶- خروجی برنامه زیر کدام است؟

```
class  a{
     int  x;
  public:
    a(int y) {x=y;}
    int  f( ) {return x;}
};
void main( ){
  a  t(5);
  a  *p;
  p=&t;
  cout<<t.f( );
  cout<<p->f( );
}
```

پاسخ : جواب 55 است.

۳۷- خروجی چیست؟

```
class  a{
   public:
     a(int  x)  {y=x;};
      int  f( )  { return  y+2;}
      int  y;
};
int  main( ){
   a   ob(2);
   int    a::*p1;
   p1=&a::y;
   cout<<ob.*p1;
   int    (a::*p2) ( ) ;
   p2 = & a::f;
  cout<<(ob.*p2)( );
}
```

پاسخ : جواب 24 است.

۳۸- با توجه به برنامه زیر، در محل خالی کدام باید قرار بگیرد؟

```
void  f( int  &a)
 { a++; }
main( ) {
  int x=5;
  .........
  cout<<x;
}
```

پاسخ : جواب f(x); است.

۳۹- خروجی چیست؟

```
main( ) {
  int  x;
  int &p=x;
  x=2;
  cout<<x+p;
  p=3;
 cout<<x+p;
  ++p;
  cout<<x+p;
}
```

پاسخ : جواب 468 است.

۴۰- خروجی چیست؟

```
int main( ){
  int    x=3;
  int  &y=x;
  cout <<x<<y ;
  y=7;
  cout <<x<<y;
}
```

پاسخ : جواب 3377 است.

۴۱- خروجی چیست؟

```
class  a{
  public:
    a(int  x)    { cout<<'a'; k=x;}
    ~a( )        { cout<< 'b';}
     int  f( )   { return  k+2;}
     int  k;
};
int  main( ){
  int   (a::*p) ( ) ;
  a      ob(2);
  p =&a::f;
  cout << (ob.*p)( );
}
```

پاسخ : جواب a4b است.

۴۲- خروجی برنامه زیر کدام است؟

```
class  count {
  public:
            int  x;
            void  print( ) { cout <<x; }
};
int main( ){
  count  a;
  count  *p=&a;
  count  &c=a;
 a.x=1;  a.print( );
  c.x = 2;  c.print( );
  p->x = 3;  p->print( );
 }
```

پاسخ : جواب 123 است.

۴۳- خروجی چیست؟

```
class  test {
 private:  int  x;
 public:  test( int = 0 );   void  print( )  const;
};
test::test( int  value ) : x( value )
```

```
  { cout <<x; }
void  test::print( ) const
 { cout <<x;   cout<<this->x;   cout<<(*this ).x ; }
int main( ) {
   test   a(5);
   a.print( );
}
```

پاسخ : جواب 5555 است.

۴۴- خروجی برنامه زیر کدام است؟

```
int    f (int x)         { return ++x ; }
int    f (int y,int z)  { return y+z;  }
char  f (char ch)    { return --ch;  }
void  main( ) {
 cout<<f(3);
 cout<<f(3,5);
 cout<<f('B');
}
```

پاسخ : جواب 48A است.

۴۵- خروجی برنامه زیر کدام است؟

```
void   f (int  x)           { cout<< ++x; }
void   f (int  y, int z)   { cout<<y+z;  }
void   main( ){
  void  (*p1) (int);
  void  (*p2) (int,int);
  p1=f;   p2=f;
  p1(3);   p2(6,2);
}
```

پاسخ : جواب 48 است.

۴۶- خروجی برنامه زیر کدام است؟

```
int   f(int x)  { return ++x;}
int   f(int y)  { return ++y;}
```

```
void  main( ){
 cout<<f(3);  cout<<f(5);
}
```

۱) 43 ۲) 45 ۳) 46 ۴) خطا دارد.

پاسخ : خطا دارد.

۴۷- خروجی برنامه زیر کدام است؟

```
int   f( int  a=1, int   b=1, int   c=1 );
int  main( ){
  cout << f( );
  cout << f(2);
  cout << f(3,4);
  cout << f(5,6,7);
}
int   f( int   a, int   b, int   c )
{
  return   a+b+c;
}
```

پاسخ : جواب 34818 است.

۴۸- خروجی برنامه زیر کدام است؟

```
int   f( int   a=1, int   b=2 )
{
  return   a+b;
}
int  main( ){
  cout << f( );
  cout << f(3);
  cout << f(4,5);
}
```

پاسخ : جواب 359 است.

۴۹- خروجی چیست؟

```
class  a{
   public :  virtual  void  f( ){ cout<<"A"; }
```

```
};
class  b : public a{
   public :  void  f( )  {cout<<"B"; }
};
class  c : public a{
    public:  void  f( )  {cout<<"C"; }
};
main( ){
   a  *p,x; b   ob1;  c  ob2;
   p=&x;         p->f( );
   p=&ob1;     p->f( );
   p=&ob2;     p->f( );
}
```

پاسخ : جواب ABC است.

۵۰- اگر کلمه virtual در تست قبل را برداریم، خروجی چیست؟

پاسخ : جواب AAA است.

۵۱- خروجی چیست؟

```
class  a{
   public:
          virtual  void  f( ){ cout<<"A"; }
};
class  b : public a{
 public:
          void  f( )  {cout<<"B"; }
};
class  c: public a{
  public:
          void  f( )  {cout<<"C"; }
};
void   m(a  &r) { r.f( ); }
main( ){
   a   x;
   b   ob1;
   c  ob2;
```

```
   m(x);
   m(ob1);
   m(ob2);
}
```

پاسخ : جواب ABC است.

۵۲- خروجی چیست؟

```
class  number{
   protected:
            int  xl;
   public:
            void     set(int  i ){ x=i;}
            virtual   void  f( )=0;
 };
class  cdec : public number{
   public:
           void  f( )  {cout<<x; }
 };
class  chex : public number{
   public:
           void  f( )  {cout<<hex<<x; }
 };
main( ) {
   cdec   d;
   chex   h;
   d.set(2);
   d.f();
   h.set(15);
   h.f();
}
```

پاسخ : خطا دارد.

۵۳- خروجی چیست؟

```
class  c{
    int  a;
    int  b;
```

```
 public:
          c( ) { }
          c( int  x , int  y)  {  a=x;  b=y; }
          void  f( )           { cout<<a<<b; }
          c   operator + (c  op2);
};
c   c::operator + (c  op2)
 {
    c  t;
    t.a=op2.a+a;
    t.b=op2.b+b;
    return  t;
}
main ( ) {
   c   ob1(1,2);
   c   ob2(3,4);
   ob1=ob1+ob2;
   ob1.f( );
}
```

پاسخ : جواب 46 است.

۵۴- خروجی چیست؟

```
class  c{
    int  a;
    int  b;
 public:
    c( ) { }
    c( int  x , int  y)  {  a=x;  b=y; }
    void  f( )  { cout<<a<<b; }
    friend  c   operator + (c   op1, int  op2);
    friend  c   operator + (int  op1, c   op2);
};

c   operator + (c  op1,int  op2)
  { c   t;  t.a =  op1.a+op2;  t.b =  op1.b+op2;  return  t; }

c   operator + (int  op1,c  op2)
 { c  t; t.a =  op1+op2.a; t.b =  op1+op2.b; return  t; }
```

```
main ( ) {
  c   ob1(1,2);
  c   ob2(3,4);
  ob2=ob1+5;
  ob2.f( );
  ob1=2+ob2;
  ob1.f( );
}
```

پاسخ : جواب 6789 است.

۵۵- خروجی چیست؟

```
class  c{
    int  a;  int  b;
 public:
    c( )  { }
    c( int  x  , int  y)  {  a=x; b=y;  }
    void  f( )              { cout<<a<<b; }
    c  operator ++ (  );
};
c   c::operator ++ ( )
{a++;  b++;   return  *this; }
main ( ) {
   c   ob1(1,2);
   ++ob1;
   ob1.f( );
}
```

پاسخ : جواب 23 است.

۵۶- در صورت ورود عدد ۴ ، خروجی کدام است؟

```
class  a{
    int   x;
 public:  friend   istream  &operator>>(istream &stream , a  &ob);    // extractor
          friend   ostream  &operator<<(ostream &stream , a  ob);     //insertion
};
istream  &operator >> (istream  &stream, a  &ob)
```

```
  { stream>>ob.x;  return stream; }
ostream  &operator <<(ostream  &stream , a  ob)
  { stream <<ob.x+2;  return  stream; }
main( ) {
 a   m;
 cin >>m;
 cout <<m;
}
```

پاسخ : جواب 6 است.

۵۷- خروجی کدام است؟

```
class  a{
   int x,y;
public:  a( int  i , int  j)  { x=i,y=j;}
          int  operator == ( a  ob2);
};

int  a::operator == ( a  ob2){
  if( x==ob2.x  &&  y==ob2.y )  return  1;  else  return 0;
}

main ( ){
 a  m(1,2), n(1,3);
 if (m==n)  cout<<"equal";  else  cout<<"not equal";
}
```

پاسخ : جواب not equal است.

۵۸- خروجی برنامه زیر کدام است؟

```
template  <class  A>
void  f(A  x) { cout<<x; }
void  main( ) {
   int  b=2; char  c='M';
  f(b);
  f(c);
}
```

پاسخ : جواب 2M است.

۵۹- اگر از ورودی اعداد 1 2 3 ↵ وارد شوند، خروجی کدام است؟

```
template <class T>
T f( T v1, T v2, T v3 )
{
  T m = v1;
  if ( v2 > m ) m = v2;
  if ( v3 > m ) m = v3;
  return m;
}
int main( ) {
  int a,b, c;
  cin >> a >> b >> c;
  cout << f( a,b,c);
 }
```

پاسخ : جواب 3 است.

۶۰- خروجی چیست؟

```
template < class T >
void f( const T *x , const int n ){
  for ( int i = 0; i <n; i++ )
       cout << x[i]<<" " ;
}
int main( ){
  const int x =3, y = 2, z = 5;
  int a[x] = { 1, 2, 3 };
  double b[y] = { 1.5, 2.5 };
  char c[z] = "HELLO";
  f(a,x);
  f(b,y);
  f(c, z);
}
```

پاسخ : جواب 1 2 3 1.5 2.5 H E L L O است.

۶۱- خروجی برنامه زیر کدام است؟

```
class c{ protected: int a; };
void main( ){
 c x;
 cout<<x.a;
}
```

پاسخ : خطا دارد.

۶۲- خروجی برنامه زیر کدام است؟

```
class a{
   private:
          int x;
   public:
      static int c;
      a( )          { ++c; cout<<'(';}
      a(int m )  { x=m;cout<<'*'; }
     ~a( )          {cout<<")"; }
};
int  a::c=0;

void main( ){
   a w,x,t(1);
   cout<<w.c;
   { a y(2),z; cout<<w.c; }
   cout<<w.c<<x;
}
```

پاسخ : جواب (((3)(3)*2*)) است.

۶۳- خروجی برنامه زیر کدام است؟

```
class a{
   public:
      static int c;
      a( )  {c-=2; }
     ~a( )  {--c; }
     ~a( )  {++c; }
};
int a::c=8;
```

```
void main( ){
   a   x;
   cout<<x.c;
   a  y;
   cout<<x.c;
}
```

پاسخ : خطا دارد.

۶۴- کدام خط(ها) دارای خطا است؟

```
class   t{
     int   a,b;
  public :
     t(int  x=2 , int y=3) : a(x) , b(++y) {p(); }       //1
     p( )   { cout<<a+b; }      //2
};
void main( ){
  t   d;
  d.t( );        //3
}
```

پاسخ : جواب ۳ است.

۶۵- خروجی برنامه زیر کدام است؟

```
class   a{
      int    x;
       public:
                int   f (int  n) { cout<< n+1;}
};
class  b : public  a{
     int  y;
      public:
               int   g(int  n) { f(5); cout<<n+2;}
};
void  main( ) {
   b  t ;
  t.f(1);
  t.g(2);
}
```

پاسخ : جواب 264 است.

۶۶- خروجی برنامه زیر کدام است؟

```
class  a{
     public:
               int   f ( )  { cout<<'1'; }
};
class  b : a{
      private: int   t;
};
void main( ){
  b  x;
 a   y;
  x.f( );
 y.f( );
}
```

پاسخ : خطا دارد.

۶۷- اعضای قسمت protected کلاس c2 عبارتند از :

```
class  c1{
    private      : int   a;
    protected  :  int   b;
    public       :  int    c;
 };

class   c2 : protected   c1{
    private      : int   d;
    protected  :  int   e;
    public       :  int    f;
};
```

پاسخ : جواب e,b,c است.

۶۸- اعضای قسمت public کلاس c2 عبارتند از :

```
class  c1{
    private      : int   a;
    protected  :  int   b;
```

```
    public     : int   c;
 };

class  c2 : public  c1{
    private     : int  d;
    protected  : int  e;
    public     : int   f;
};
```

پاسخ : جواب c,f است.

۶۹- کدام خط(ها) خطا دارد؟

```
class  a{
  protected:  int  x;
  public:      f(int  i) {x=i;}
};
class  b : private  a{
    public:  void  g( ) { y=x+1;}    //1
    int  y;
};
class  c : public  b{
    int  z;
 public:  void  h( )  {z=y+2;}     //2
};
 main( ){
  b   m;
  c   n;
  m.g( );          //3
  n.f(1);          //4
}
```

پاسخ : خط ۴ .

۷۰- خروجی چیست؟

```
class  a{
int  i;
public:  a(int  x ){i=x;}
        ~a( )   {cout<<i;}
```

```
};
class  b : public a{
int  j;
public:          b(int  x, int  y ) : a(x)   { j=y ;cout<<j; }
};
main( ) {
b   ob1(1,2);
b   ob2(3,4);
}
```

پاسخ : جواب ۲۴۳۱ است.

۷۱- خروجی کدام است؟

```
class   c1{
    int  a;
 public :   c1(int  x)  { a=x;}
            int  f( )     { return   a; }
};
class  c2{
     int  b;
  public :    c2(int  x)  { b=x; }
};
class  c3 : public  c1, public  c2{
      int  c;
   public :    c3 ( int  x , int y , int z) : c1(z) ,c2(y) { c=x;}
               void  p ( ) { cout <<f ( ) + c;  }
};
main( ) {
 c3  ob(1,2,3);
 ob.p( );
}
```

پاسخ : جواب 4 است.

۷۲- خروجی کدام است؟

```
class   a{
   public:  int  i;
        a(int  x)  { i=x; cout<<i;}
};
class  b: public  a{
```

```
    public :  b(int  x) : a(x++) {  }
};
class  c : public  b{
    public : c( int  x ) : b(++x) {  }
};
main( ) {
  c   ob(1);
}
```

پاسخ : جواب 2 است.

۷۳- خروجی برنامه زیر کدام است؟

```
class  a{
    int  x; int  y;
  public:
    a(int  i, int  j) {x=++i;  y=i+j;}
    int   f()   { return  x;}
};
int  main( ){
 int   i;
 a   ob[2]={a(3,1) ,a(2,4) };
 int   s=0;
 for(i=0 ; i<2 ;i++)
    s+=ob[i].f( );
 cout<<s;
}
```

پاسخ : جواب 7 است.

۷۴- خروجی چیست؟

```
class  c{
    int  a;
  public:
       c(int  x)              { a=x;}
       friend  int   h(c)   {c  z;  cout<<z.a; };
 };
main( ){
  c  k(5);
```

```
  h(k);
}
```

پاسخ : خطا دارد.

۷۵- خروجی برنامه زیر کدام است؟

```
class a{
  int  x;
  public:
        friend   void g( ) { cout<<'1';}     //1
};
void   main( ){
  a  t;
  g( );                 //2
  t.g( ) ;              //3
}
```

پاسخ : خط ۳ خطا دارد.

۷۶- خروجی برنامه زیر کدام است؟

```
class   a {
    int  x;
  public:
      a( ) : x(2)    {cout<<x<<"*";}
      friend   void    f( a  &,  int );
      void   p( )  { cout <<++x ; }
};

void   f( a  &k ,  int  n )
  { k.x =--n;  }

int  main( ){
  a    ob;
  ob.p( );
  f ( ob,7);
  ob.p( );
}
```

پاسخ : جواب 2*38 است.

۷۷- خروجی چیست؟

```
class  c{
    int  a;
 public:
     c(int  x)  { a=x;}
     friend   class   d;
};

class  d{
 public :
    int  f(c  z) { cout<<z.a+3;}
};
main( ){
  c   ob1(5);
  d   ob2;
  ob2.f(ob1);
}
```

پاسخ : جواب 8 است.

۷۸- خروجی چیست؟

```
class   c{
     static  int   a;
     int  b;
  public:    f (int  x, int  y)  { a=x;b=y;}
             void   g( ) { cout<<a+b++;}
};
int  c::a;
main( ){
  c  o1,o2;
  o1.f(5,4);
  o1.g();
  o2.f(2,6);
  o2.g( );
  o1.g( );
}
```

پاسخ : جواب 987 است.

۷۹- خروجی چیست؟

```
class c{
      public:
            static int  a;
};
int c::a;
main( ){
 c ::a=1;
 cout<<c::a++;
 c x;
 cout<<x.a;
}
```

پاسخ : جواب 12 است.

۸۰- خروجی چیست؟

```
class a{
  public: static int c;
           a( ) {c++;}
          ~a( ) { c++;}
};
int a::c;
void f( ) { a b; cout<<a::c; }
main( ){
   a ob1,ob2;
  f( );
  cout<<a::c;
   a ob3,ob4;
   cout<<a::c;
   f( );
   cout<<a::c;
}
```

پاسخ : جواب 34678 است.

۸۱- کدام خط خطا دارد؟

```
class c{
     int  a;
     static int b;
 public:
    int d;
    static g( )          //1
     {
        a=1;            //2
        b=2;            //3
        d=3;            //4
      }
};
int c::a=4;             //5
main( ){
 c ob;
 ob.g( );
}
```

پاسخ : خط های : 2,4,5

۸۲- خروجی چیست؟

```
class  c {
    int      x;
    const int c;
  public:
    c( int c=0, int i=1 );
    void   f( ) { x =++c; }
    void   p( ) const;
};
c::c (int c, int i ) : x(c), c(i) {   x = c;   c = i; }
void c::p( ) const {   cout << x<< c ; }
int main( ) {
  c   a(1,2);
 a.f( ); a.p( );   a.f( ); a.p( );
}
```

پاسخ : خطا دارد.

۸۳- کدام خط(ها) دارای خطا می باشد؟

```
class  a{
    static  int      x;
    const  int       y;
 public:
    a(int  m=1) : x(m)      { cout<<x; }        //1
    static   int   f( ) const   { return  ++x; }   //2
    g( )   { cout<<x;  return  y;   }              //3
};
```

پاسخ : خط های ۲ و۳ .

۸۴- خروجی چیست؟

```
class   c{
      int  x;
public:
    c( )          { x=1;}
    c(int  y)     { x=++y;}
    int  fun( )   { return  x--;}
};
int  main( ) {
  c  z[3]={1,2,3};
  c  *q;
  int  i;
  q= z;
  for(i=0 ; i<3 ; i++)
   {
     cout<<q->fun( );
     q++;
   }
}
```

پاسخ : جواب 234 است.

۸۵- خروجی برنامه زیر کدام است؟

```
class  count{
  public :
     int  x;
     void  p( ) { cout<<++x;}
  };
```

```
void   main( ) {
 count   c,*k=&c;
 c.x=2;
 c.p( );
 k->x=4;
  k->p( );
 (++*k).x=1;
 (*k).p( );
}
```

پاسخ : جواب 352 است.

۸۶- خروجی چیست؟

```
class   a{
  public:
     a(int  x)  {y=x+2;}
      int  f( )  { return  ++y;}
      int  y;
};
int  main( ){
  a   ob(1);
  int    a::*p1;
  p1=&a::y;
  cout<<ob.*p1;
  int    (a::*p2) ( ) ;
  p2 = & a::f;
 cout<<(ob.*p2)( );
}
```

پاسخ : جواب 34 است.

۸۷- خروجی برنامه زیر کدام است؟

```
void   h (int  x)          { cout<< x+2; }
void   h(int  y, int z)   { cout<<y*z;  }
void   main( ){
  void  (*p) (int);
  void  (*q) (int,int);
```

```
  p=h;
  q=h;
  p(1);
  q(2,3);
}
```

پاسخ : جواب36 است.

۸۸- خروجی چیست؟

```
class  a{
   public :  virtual  void  f( ){ cout<<'1'; }
 };
class  b : public a{
   public :  void  f( )  {cout<<'2'; }
 };
class  c : public a{
    public:  void  f( )  {cout<<'3'; }
 };
main( ){
   a  *p,x;
   b   ob1;
   c  ob2;
   p=&x;         p->f( );
   p=&ob1;     p->f( );
   p=&ob2;     p->f( );
}
```

پاسخ : جواب 123 است.

۸۹- خروجی چیست؟

```
class  c{
          int  a;
          int  b;
 public:
          c( )  { }
          c( int  x , int  y)  {  a=x;  b=y; }
          void  f( )              { cout<<a<<b; }
          c   operator -(c  op2);
```

```
};
c   c::operator - (c  op2)
 {
    c  t;
   t.a=op2.a-a;
   t.b=op2.b-b;
   return  t;
}
main ( ) {
   c   ob1(1,2);
   c   ob2(3,4);
   ob1=ob1-ob2;
   ob1.f( );
}
```

پاسخ : جواب -2-2 است.

فرادرس

دسته‌بندی موضوعی آموزش‌های فرادرس، در ادامه آمده است:

 برنامه نویسی – کلیک (+)	 آموزش های دانشگاهی و تخصصی – کلیک (+)	 هوش مصنوعی و یادگیری ماشین – کلیک (+)	 مهندسی برق الکترونیک و روباتیک – کلیک (+)
 دروس دانشگاهی – کلیک (+)	 مباحث مشترک – کلیک (+)	 مهارت های دانشگاهی – کلیک (+)	 نرم افزارهای تخصصی – کلیک (+)
 مهندسی نرم افزار – کلیک (+)	 نرم افزارهای عمومی – کلیک (+)	 طراحی و توسعه وب – کلیک (+)	 آموزش های عمومی – کلیک (+)

منبع مطالعاتی تکمیلی مرتبط با این کتاب

آموزش ویدئویی برنامه نویسی C++



عمومیت زبان C++ در میان زبان‌های برنامه‌نویسی بسیار بالا است و می تواند به عنوان اولین زبان نیز یاد گرفته شود و به پیش نیاز دیگر احتیاج نباشد.

مجموعه فیلم‌های آموزشی برنامه‌نویسی C++، با این فرض تهیه شده است که مخاطب هیچ دانش و تجربه قبلی در زمینه برنامه‌نویسی ندارد و در این مجموعه آموزشی، همه مباحث با بیان و تشریح مبانی نظری و سپس با پیاده‌سازی گام به گام مثال‌های عملی آموزش داده می‌شوند و از این نظر، در ایجاد یک دانش عمیق در زمینه برنامه‌نویسی، بسیار کارآمد است.

مدرس: مهندس فرشید شیر افکن

مدت زمان: ۲۰ ساعت

جهت مشاهده آموزش ویدئویی این آموزش – کلیک کنید **faradars.org /fvcp9504**

فرادرس