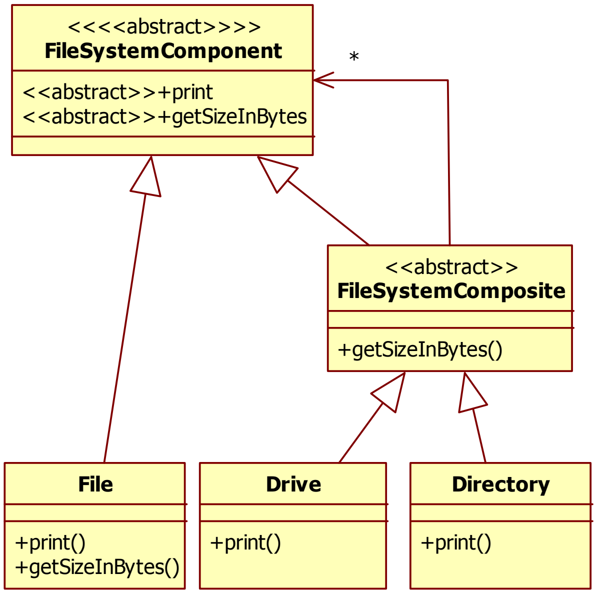

To clarify the UML diagram, here is the corresponding code:

```
public class Application {

      public static void main(String[] args) {
            Drive cdrive = new Drive("C");
            Directory appdir = new Directory("applications");
            Directory datadir = new Directory("my data");
            Directory coursedir = new Directory("cs525");
            File excelfile = new File("msexcel.exe", 2353256);
            File wordfile = new File("msword.exe", 3363858);
            File studentsfile = new File("students.doc", 34252);
            cdrive.addComponent(appdir);
            cdrive.addComponent(datadir);
            datadir.addComponent(coursedir);
            appdir.addComponent(excelfile);
            appdir.addComponent(wordfile);
            coursedir.addComponent(studentsfile);
            cdrive.print();
      }
}
public abstract class FileSystemComponent {
    protected String name;

      public FileSystemComponent(String name) {
            this.name = name;
      }

      public abstract void print();
      public abstract int getSizeInBytes();
}
```

```
public abstract class FileSystemComposite extends FileSystemComponent{
    protected Collection<FileSystemComponent> list = new
                ArrayList<FileSystemComponent>();

    public FileSystemComposite(String name) {
      super(name);
      }

    public void addComponent(FileSystemComponent component){
      list.add(component);
     }

    public int getSizeInBytes(){
      int sizeInBytes=0;
      for (FileSystemComponent component : list){
            sizeInBytes+=component.getSizeInBytes();
      }
      return sizeInBytes;
    }
}

public class Directory extends FileSystemComposite{
    public Directory(String name) {
      super(name);
      }
    public void print(){
      System.out.println("-- dir "+name+" size="+getSizeInBytes()+"
bytes");
      for (FileSystemComponent component : list){
            component.print();
      }
    }
}
public class Drive extends FileSystemComposite{
    public Drive(String name) {
            super(name);
      }
    public void print(){
      System.out.println("- drive "+name+" size="+getSizeInBytes()+"
bytes");
      for (FileSystemComponent component : list){
            component.print();
      }
   }
}

public class File extends FileSystemComponent{
    private int sizeInBytes;

    public File(String name, int sizeInBytes) {
      super(name);
            this.sizeInBytes = sizeInBytes;
      }

      public int getSizeInBytes() {
            return sizeInBytes;
      }

    public void print(){
      System.out.println("--- file "+name+" size="+getSizeInBytes()+"
bytes");
    }
}
```

Important here is that you have a (abstract)composite class that has a list of objects of type (abstract)component so it is easy to add new File system types.