



AP[®] Computer Science AB 2004 Sample Student Responses

The materials included in these files are intended for noncommercial use by AP teachers for course and exam preparation; permission for any other use must be sought from the Advanced Placement Program[®]. Teachers may reproduce them, in whole or in part, in limited quantities, for face-to-face teaching purposes but may not mass distribute the materials, electronically or otherwise. This permission does not apply to any third-party copyrights contained herein. These materials and any copies made of them may not be resold, and the copyright notices must be retained as they appear here.

The College Board is a not-for-profit membership association whose mission is to connect students to college success and opportunity. Founded in 1900, the association is composed of more than 4,500 schools, colleges, universities, and other educational organizations. Each year, the College Board serves over three million students and their parents, 23,000 high schools, and 3,500 colleges through major programs and services in college admissions, guidance, assessment, financial aid, enrollment, and teaching and learning. Among its best-known programs are the SAT[®], the PSAT/NMSQT[®], and the Advanced Placement Program[®] (AP[®]). The College Board is committed to the principles of excellence and equity, and that commitment is embodied in all of its programs, services, activities, and concerns.

For further information, visit www.collegeboard.com

Copyright © 2004 College Entrance Examination Board. All rights reserved. College Board, Advanced Placement Program, AP, AP Central, AP Vertical Teams, APCD, Pacesetter, Pre-AP, SAT, Student Search Service, and the acorn logo are registered trademarks of the College Entrance Examination Board. PSAT/NMSQT is a registered trademark of the College Entrance Examination Board and National Merit Scholarship Corporation. Educational Testing Service and ETS are registered trademarks of Educational Testing Service. Other products and services may be trademarks of their respective owners.

For the College Board's online home for AP professionals, visit AP Central at apcentral.collegeboard.com.

1. Consider designing and implementing a set of classes and interfaces to represent books, library items, and library books.

- (a) A library item contains two pieces of information: a unique ID (a `String`) and a holder (also a `String`). If the library item has been checked out, the holder is the name of the person who has checked out this item, otherwise the holder is `null`. The ID of a library item cannot change after it is created, but the holder can change. Write the `LibraryItem` interface that abstracts this functionality.

```
public interface LibraryItem
{
    String    get ID();
    String    get Holder();
    void      change Holder( String new Holder );
}
```

Part (b) begins on page 6.

GO ON TO THE NEXT PAGE.

A₂

- (b) A book consists of an author and a title, neither of which can change once the book is created. The Book class is represented as follows.

```
public class Book
{
    private String theTitle;
    private String theAuthor;

    public Book(String author, String title)
    { theAuthor = author; theTitle = title; }

    public String getAuthor()
    { return theAuthor; }

    public String getTitle()
    { return theTitle; }
}
```

A library book is a book that is also a library item. Write the complete class LibraryBook, implementing the required methods.

```
public class LibraryBook extends Book implements LibraryItem
{
    public LibraryBook(String author, String title, String id)
    {
        super(author, title);
        theID = id;
        theHolder = null;
    }

    public String getID()
    { return theID; }

    public String getHolder()
    { return theHolder; }

    public void changeHolder(String newHolder)
    { theHolder = newHolder; }

    private String theID;
    private String theHolder;
}
```

GO ON TO THE NEXT PAGE.

(c) A library is a collection of library items and supports the following operations.

1. Add a library item to the library.
2. Check out a library item by specifying its ID and new holder.
3. Determine the current holder of a library item, given its ID.

Consider the following incomplete class declaration.

```
public class Library
{
    private SomeClass[] items;

    public void addLibraryItem()
    { /* implementation not shown */ }

    public void checkout(String id, String holder)
    { /* implementation not shown */ }

    public String getHolder(String id)
    { /* implementation not shown */ }
}
```

Choose a data representation for **SomeClass** from the `java.util` class library that allows the operations `add`, `checkout`, and `getHolder` to be performed efficiently. In doing so, you must explain how you are using your chosen `java.util` class to organize your data and give the expected Big-Oh running time for each of these three operations in terms of n , the number of items in the library.

Your data representation (Circle one)	How data is organized
<div>ArrayList</div> <div>HashMap</div> <div>HashSet</div> <div>LinkedList</div> <div>TreeMap</div> <div>TreeSet</div>	Use IDs as keys mapping to LibraryItems.

Operations	Expected Big-Oh efficiency
add	$O(1)$
checkout	$O(1)$
getHolder	$O(1)$

GO ON TO THE NEXT PAGE.

AB1 B1

1. Consider designing and implementing a set of classes and interfaces to represent books, library items, and library books.

- (a) A library item contains two pieces of information: a unique ID (a `String`) and a holder (also a `String`). If the library item has been checked out, the holder is the name of the person who has checked out this item, otherwise the holder is `null`. The ID of a library item cannot change after it is created, but the holder can change. Write the `LibraryItem` interface that abstracts this functionality.

```
public interface LibraryItem
{
    String getID();
    String getHolder();
    void setHolder(String h);
}
```

Part (b) begins on page 6.

GO ON TO THE NEXT PAGE.

- (b) A book consists of an author and a title, neither of which can change once the book is created. The Book class is represented as follows.

```
public class Book
{
    private String author;
    private String title;

    public Book(String author, String title)
    { theAuthor = author; theTitle = title; }

    public String getAuthor()
    { return theAuthor; }

    public String getTitle()
    { return theTitle; }
}
```

A library book is a book that is also a library item. Write the complete class LibraryBook, implementing the required methods.

```
public class LibraryBook extends LibraryItem
{
    public String myID;
    public String myHolder;
    public Book bk;

    public LibraryBook(Book novel, String id)
    {
        myID = id;
        myHolder = null;
        bk = novel;
    }

    public String getID()
    {
        return (myID);
    }

    public String getHolder()
    {
        return (myHolder);
    }

    public void setHolder(String h)
    {
        myHolder = h;
    }
}
```

GO ON TO THE NEXT PAGE.

(c) A library is a collection of library items and supports the following operations.

1. Add a library item to the library.
2. Check out a library item by specifying its ID and new holder.
3. Determine the current holder of a library item, given its ID.

Consider the following incomplete class declaration.

```
public class Library
{
    private SomeClass items;

    public Library()
    { /* implementation not shown */ }

    public void addLibraryItem(SomeClass item)
    { /* implementation not shown */ }

    public void checkoutLibraryItem(SomeClass item, String holder)
    { /* implementation not shown */ }

    public String getHolder(String id)
    { /* implementation not shown */ }
}
```

Choose a data representation for **SomeClass** from the `java.util` class library that allows the operations `add`, `checkout`, and `getHolder` to be performed efficiently. In doing so, you must explain how you are using your chosen `java.util` class to organize your data and give the expected Big-Oh running time for each of these three operations in terms of n , the number of items in the library.

Your data representation (Circle one)	How data is organized
<div>ArrayList</div> <div>LinkedList</div> <div>HashMap</div> <div>TreeMap</div> <div><u>HashSet</u></div> <div>TreeSet</div>	LibraryItems are stored in a HashSet using id to create a Hash key value.

Operations	Expected Big-Oh efficiency
add	$O(1)$
checkout	$O(1)$
getHolder	$O(1)$

GO ON TO THE NEXT PAGE

AB/C.

1. Consider designing and implementing a set of classes and interfaces to represent books, library items, and library books.
 - (a) A library item contains two pieces of information: a unique ID (a `String`) and a holder (also a `String`). If the library item has been checked out, the holder is the name of the person who has checked out this item, otherwise the holder is `null`. The ID of a library item cannot change after it is created, but the holder can change. Write the `LibraryItem` interface that abstracts this functionality.

LibraryItem Interface

```
{  
    public LibraryItem(String id, String holder)  
    public String ID, holder;  
    public void changeholder(String newholder)  
    public String getHolder()  
    public String getID()  
}
```

}

Part (b) begins on page 6.

GO ON TO THE NEXT PAGE.

C₂

- (b) A book consists of an author and a title, neither of which can change once the book is created. The `Book` class is represented as follows.

```
public class Book
{
    private String theAuthor;
    private String theTitle;

    public Book(String author, String title)
    { theAuthor = author; theTitle = title; }

    public String getAuthor()
    { return theAuthor; }

    public String getTitle()
    { return theTitle; }
}
```

A library book is a book that is also a library item. Write the complete class `LibraryBook`, implementing the required methods.

```
class LibraryBook extends Book, implements LibraryItem
{
    public LibraryBook(String author, String title)
    {
        super(author, title);
    }
}
```

GO ON TO THE NEXT PAGE.

C₃

(c) A library is a collection of library items and supports the following operations.

1. Add a library item to the library.
2. Check out a library item by specifying its ID and new holder.
3. Determine the current holder of a library item, given its ID.

Consider the following incomplete class declaration.

```
public class Library
{
    ArrayList<SomeClass> items;

    SomeClass item;
    { /* implementation not shown */ }

    void addLibraryItem(SomeClass item)
    { /* implementation not shown */ }

    SomeClass checkoutLibraryItem(int ID, String holder)
    { /* implementation not shown */ }

    String getHolder(int ID)
    { /* implementation not shown */ }
}
```

Choose a data representation for **SomeClass** from the `java.util` class library that allows the operations `add`, `checkout`, and `getHolder` to be performed efficiently. In doing so, you must explain how you are using your chosen `java.util` class to organize your data and give the expected Big-Oh running time for each of these three operations in terms of n , the number of items in the library.

Your data representation (Circle one)	How data is organized
<div>ArrayList</div> <div>HashMap</div> <div><u>HashSet</u></div> <div>LinkedList</div> <div>TreeMap</div> <div>TreeSet</div>	data organized by last 3 #'s of ID#

Operations	Expected Big-Oh efficiency
add	$O(1)$
checkout	$O(1)$
getHolder	$O(1)$

GO ON TO THE NEXT PAGE.