



## AP<sup>®</sup> Computer Science A 2002 Sample Student Responses

**The materials included in these files are intended for use by AP teachers for course and exam preparation in the classroom; permission for any other use must be sought from the Advanced Placement Program<sup>®</sup>. 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. These materials and any copies made of them may not be resold, and the copyright notices must be retained as they appear here. This permission does not apply to any third-party copyrights contained herein.**

These materials were produced by Educational Testing Service<sup>®</sup> (ETS<sup>®</sup>), which develops and administers the examinations of the Advanced Placement Program for the College Board. The College Board and Educational Testing Service (ETS) are dedicated to the principle of equal opportunity, and their programs, services, and employment policies are guided by that principle.

The College Board is a national nonprofit membership association dedicated to preparing, inspiring, and connecting students to college and opportunity. Founded in 1900, the association is composed of more than 4,200 schools, colleges, universities, and other educational organizations. Each year, the College Board serves over three million students and their parents, 22,000 high schools, and 3,500 colleges, through major programs and services in college admission, guidance, assessment, financial aid, enrollment, and teaching and learning. Among its best-known programs are the SAT<sup>®</sup>, the PSAT/NMSQT<sup>®</sup>, and the Advanced Placement Program<sup>®</sup> (AP<sup>®</sup>). The College Board is committed to the principles of equity and excellence, and that commitment is embodied in all of its programs, services, activities, and concerns.

Copyright © 2002 by College Entrance Examination Board. All rights reserved. College Board, Advanced Placement Program, AP, SAT, and the acorn logo are registered trademarks of the College Entrance Examination Board. APIEL is a trademark owned by the College Entrance Examination Board. PSAT/NMSQT is a registered trademark jointly owned by the College Entrance Examination Board and the National Merit Scholarship Corporation. Educational Testing Service and ETS are registered trademarks of Educational Testing Service.

Complete function CalculateModes below.

```
apvector<int> CalculateModes(const apvector<int> & tally)
// precondition: tally.length() > 0
// postcondition: returns an apvector that contains the mode(s);
//               the apvector's length equals the number of modes
```

```
{ int mode = FindMax(tally);
  apvector<int> mode(tally.length());
  int nummodes = 0;
  int k;
  for (k = 0; k < tally.length(); k++)
  { if (mode == tally[k])
    { mode[nummodes] = k;
      nummodes++;
    }
  }
  mode.resize(nummodes);
  return mode;
}
```

**GO ON TO THE NEXT PAGE.**

- (b) You will write the function `KthDataValue`, which is described as follows. `KthDataValue` returns the  $k$ th data value when the data values are considered in sorted order. Recall that the indexes of the array represent possible data values and that each array location contains the frequency of the value corresponding to its index.

In the example reprinted below, the first ten data values are 2, the next five data values are 3, and the next ten data values are 4. `KthDataValue(tally, 1)` returns 2, `KthDataValue(tally, 14)` returns 3, `KthDataValue(tally, 15)` returns 3, and `KthDataValue(tally, 16)` returns 4.

`tally`

Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Frequency	0	0	10	5	10	0	7	1	0	6	0	10	3	0	0	1

Complete function `KthDataValue` below.

```
int KthDataValue(const apvector<int> & tally, int k)
// precondition:  tally.length() > 0;
//                0 < k ≤ total number of values in the data collection
// postcondition: returns the kth value in the data collection
//                represented by tally
```

```
{ int x;
  for (x=0; x < tally.length(); x++)
  { k = k - tally[x];
    if (k <= 0)
      return x;
  }
}
```

**GO ON TO THE NEXT PAGE.**

A1 B

Complete function CalculateModes below.

```
apvector<int> CalculateModes(const apvector<int> & tally)
// precondition:  tally.length() > 0
// postcondition: returns an apvector that contains the mode(s);
//               the apvector's length equals the number of modes
{
    apvector<int> calc(0);
    for (int i=0; i<tally.length(); i++)
    {
        if (tally[i] == FindMax(tally))
        {
            calc.resize(calc.length()+1);
            calc[calc.length()-1] = i;
        }
    }
    return calc;
}
```

**GO ON TO THE NEXT PAGE.**

- (b) You will write the function `KthDataValue`, which is described as follows. `KthDataValue` returns the  $k$ th data value when the data values are considered in sorted order. Recall that the indexes of the array represent possible data values and that each array location contains the frequency of the value corresponding to its index.

In the example reprinted below, the first ten data values are 2, the next five data values are 3, and the next ten data values are 4. `KthDataValue(tally, 1)` returns 2, `KthDataValue(tally, 14)` returns 3, `KthDataValue(tally, 15)` returns 3, and `KthDataValue(tally, 16)` returns 4.

tally

Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Frequency	0	0	10	5	10	0	7	1	0	6	0	10	3	0	0	1

Complete function `KthDataValue` below.

```
int KthDataValue(const apvector<int> & tally, int k)
// precondition: tally.length() > 0;
//                0 < k ≤ total number of values in the data collection
// postcondition: returns the kth value in the data collection
//                represented by tally
{
    int j;
    int counter=0;
    for (int i=0; i<tally.length(); i++)
    {
        for (j=0; j<tally[i]; j++)
        {
            counter++;
            if (counter == k)
                return i;
        }
    }
}
```

**GO ON TO THE NEXT PAGE.**

Complete function CalculateModes below.

A1 C

```
apvector<int> CalculateModes(const apvector<int> & tally)
// precondition: tally.length() > 0
// postcondition: returns an apvector that contains the mode(s);
// the apvector's length equals the number of modes
{
    apvector<int> modes;
    int j = 0, numModes;

    for (k = 0; k < tally.length(); k++)
    {
        if (tally[k] == FindMax(tally))
        {
            numModes++;
            modes.resize(numModes);
            modes[j] = tally[k];
            j++;
        }
    }
    return modes;
}
```

GO ON TO THE NEXT PAGE.

- (b) You will write the function `KthDataValue`, which is described as follows. `KthDataValue` returns the  $k$ th data value when the data values are considered in sorted order. Recall that the indexes of the array represent possible data values and that each array location contains the frequency of the value corresponding to its index.

In the example reprinted below, the first ten data values are **2**, the next five data values are **3**, and the next ten data values are **4**. `KthDataValue(tally, 1)` returns **2**, `KthDataValue(tally, 14)` returns **3**, `KthDataValue(tally, 15)` returns **3**, and `KthDataValue(tally, 16)` returns **4**.

`tally`

Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Frequency	0	0	10	5	10	0	7	1	0	6	0	10	3	0	0	1

Complete function `KthDataValue` below.

```
int KthDataValue(const apvector<int> & tally, int k)
// precondition:  tally.length() > 0;
//                0 < k ≤ total number of values in the data collection
// postcondition: returns the kth value in the data collection
//                represented by tally
```

2  
3

```
int j;
for (j = 0; j < tally.length(); j++)
    if (tally[j] == k)
        return j;
```

2  
3

**GO ON TO THE NEXT PAGE.**