The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

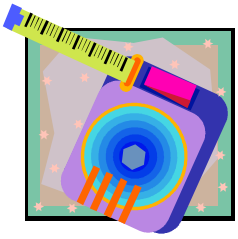
Team Packet

Java Problems

The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

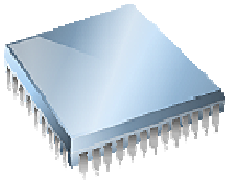
Java Problems



Distance 3D ~ 10 points

The Chinese Animal Zodiac
Year Problem ~ 10 points

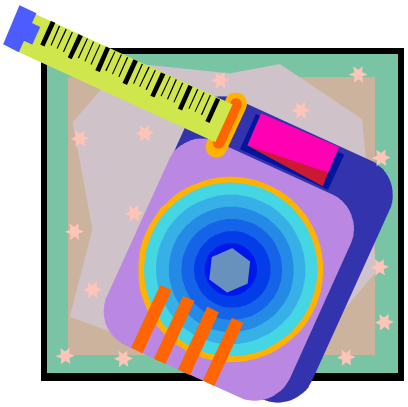
Maya Calendar ~ 30 points



The Base 64 Encoder ~ 30 points

The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009



Distance 3D (10 points)

Write a Java console program that calculates the distance between a pair of points on a three dimensional Cartesian coordinate system.

The distance *d* between two points, (*x*1,*y*1,*z*1) and (*x*2,*y*2,*z*2), on a three dimensional Cartesian coordinate system is

$$d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2}$$

Input Specification

Enter each of the six integer coordinates three per line. The first three integers represent the coordinates of one point on the coordinate system. The second three integers represent another point on the same system.

Output Specification

After entering the coordinates, output a line that says "The distance from (x,y,z) to (a,b,c) is d.ddd" where x,y,z,a,b, and c stand for the actual integers input by the user and d.ddd stands for the distance between those two points with three decimal places displayed (0's if needed) and rounded to the nearest thousandth.

Sample runs

```
Enter the three coordinates of the first  point: 0 0 0
Enter the three coordinates of the second point: 0 0 0
The distance from (0,0,0) to (0,0,0) is 0.000

Enter the three coordinates of the first  point: 1 -2 3
Enter the three coordinates of the second point: 1 -2 4
The distance from (1,-2,3) to (1,-2,4) is 1.000
```

(more sample runs on next page...)

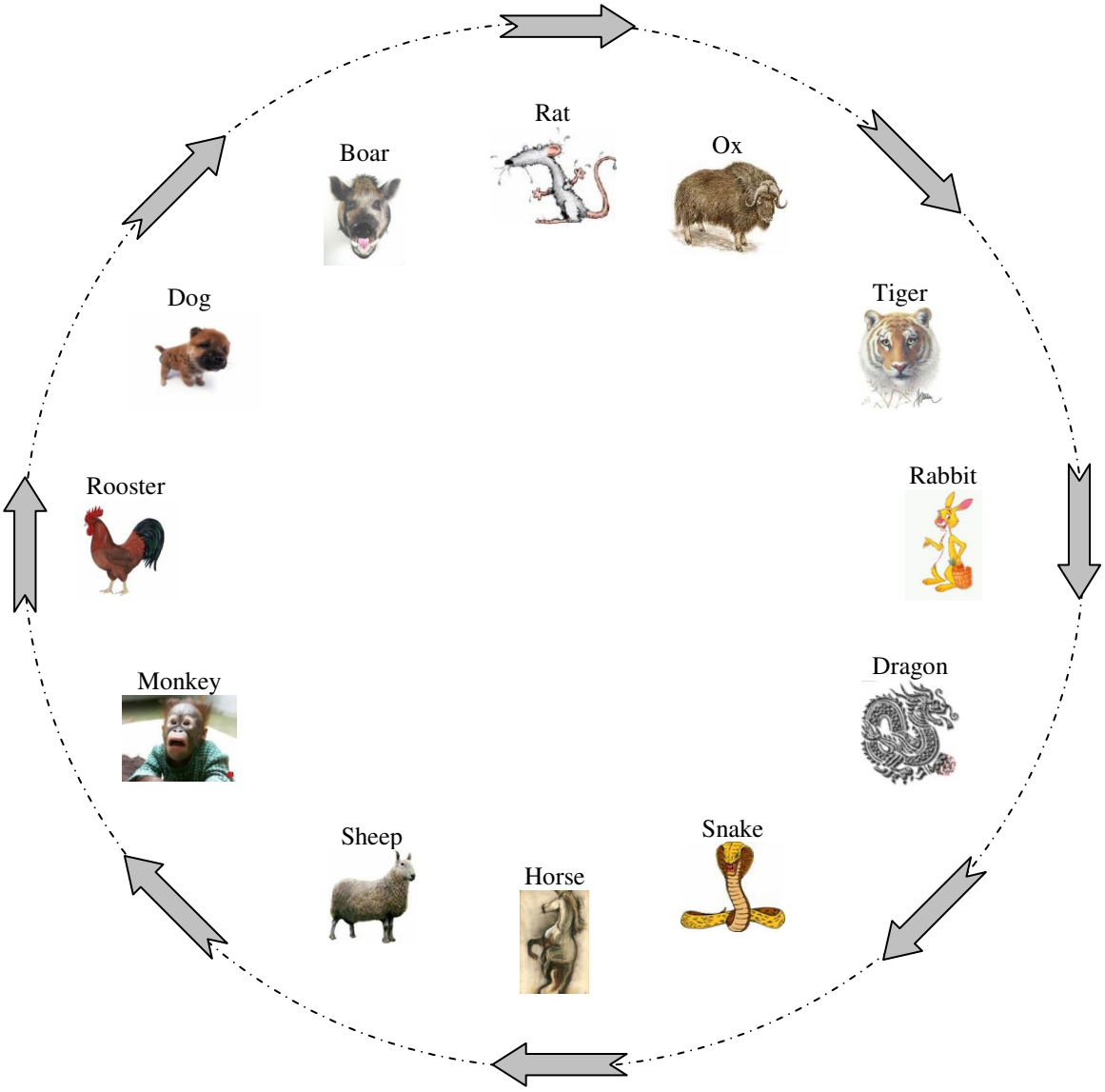
```
Enter the three coordinates of the first  point: 1 2 3
Enter the three coordinates of the second point: 4 5 6
The distance from (1,2,3) to (4,5,6) is 5.196


Enter the three coordinates of the first  point: 18 22 72
Enter the three coordinates of the second point: -122 -144 99
The distance from (18,22,72) to (-122,-144,99) is 218.826
```

The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

The Chinese Animal Zodiac Year Problem (10 points)

In the Chinese Animal Zodiac calendar, the years, for which we use numbers, are designated by twelve animals, beginning with the Rat:



Years are called "Year of the Rat", "Year of the Ox", etc.

When the "Year of the Boar" is reached, the next year is "Year of the Rat" again and the cycle repeats.

Although the Chinese New Year falls on different days yearly, somewhere between late January and early February based on the cycles of the moon, for the purposes of this problem, we will assume that Chinese Animal Zodiac years correspond exactly to years on our Western calendar (so years begin on January 1).

1996 was “The Year of the Rat”.

Write a Java program that allows the user to input a Western numerical year from 1500 to 2999 inclusive and then outputs the Chinese Animal Zodiac year in the format used in the sample run below. Use "was", "is", or "will be" properly. Replace "Sample" with your school's name.

Input repeats until a year outside the given range is entered.

Sample run:

```
Program to convert a Western Year
to a Chinese Animal Zodiac Year.
By the Java team from Sample High School.

Enter a year (1500-2999, any other year to quit): 1500
1500 was the Year of the Monkey
Enter a year (1500-2999, any other year to quit): 2007
2007 was the Year of the Boar
Enter a year (1500-2999, any other year to quit): 2008
2008 was the Year of the Rat
Enter a year (1500-2999, any other year to quit): 2008
2009 is the Year of the Ox
Enter a year (1500-2999, any other year to quit): 2999
2999 will be the Year of the Sheep
Enter a year (1500-2999, any other year to quit): 3000
Press any key to continue . . .
```

P.S.—This problem calls for a console application. You get no extra points for using a GUI.

The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

Maya Calendar (30 points)

25 points for solving the problem. 5 points for using a GUI. No GUI points if input fails.

The Maya civilization used a 365 day long year, called *Haab*, which had 19 months. Each of the first 18 months was 20 days long, and the names of the months were *pop, uo, zip, zotz, tzec, xul, yaxkin, mol, chen, yax, zac, ceh, mac, kankin, muan, pax, kayab, cumku*. Instead of having names, the days of the months were denoted by numbers starting from 0 to 19 (using their vigesimal, or base 20, numbering system). The last month of *Haab* was called *uayet* and had 5 days denoted by numbers 0, 1, 2, 3, 4. The Maya believed that this month was unlucky, the court of justice was not in session, the trade stopped, people did not even sweep the floor.

For religious purposes, the Maya used another calendar in which the year was called *Tzolkin* (holly year). The year was divided into thirteen periods, each 20 days long. Each day was denoted by a pair consisting of a number and the name of the day. They used 20 names: *imix, ik, akbal, kan, chicchan, kimi, manik, lamat, muluk, ok, chuwen, eb, ben, ix, mem, kib, kaban, etznab, kawac, ajaw* and 13 numbers; both in cycles.

Notice that each day has an unambiguous description. For example, at the beginning of the year the days were described as follows:

1 imix, 2 ik, 3 akbal, 4 kan, 5 chicchan, 6 kimi, 7 manik, 8 lamat, 9 muluk, 10 ok, 11 chuwen, 12 eb, 13 ben, 1 ix, 2 mem, 3 kib, 4 kaban, 5 etznab, 6 kawac, 7 ajaw, and again in the next period *8 imix, 9 ik, 10 akbal...*

Years (both *Haab* and *Tzolkin*) were denoted by numbers 0, 1, ..., where the number 0 was the beginning of the world. Thus, the first day was:

Haab: `0 pop 0` (Format: *NumericalDayOfTheMonth Month Year*)
Tzolkin: `1 imix 0` (Format: *Number NameOfTheDay Year*)

Write a program to convert any date in the *Haab* calendar up to and including the year 9999 to its corresponding date in the *Tzolkin* calendar. Your program should reject invalid input.

Sample conversions:

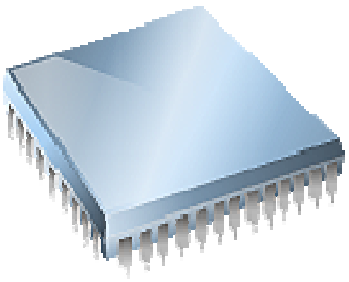
Haab	*Tzolkin*
0 pop 0	1 imix 0
5 kayab 500	7 kimi 703
8 zip 27	11 kan 38
0 pop 2006	5 chuwen 2816

Haab	*Tzolkin*
0 pop 1	2 kimi 1
3 uayet 1001	13 muluk 1406
14 cumku 9999	6 ok 14038
5 uayet 20	*invalid input*

The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

The Base 64 Encoder (30 points)



E-mail protocol (SMTP, or Simple Mail Transfer Protocol) only allows 7 bits of information to be sent at a time. Of course, information must be transfered across the Internet in bytes. And bytes contain 8 bits. So in e-mail, the 8th bit is always 0. With 7 bits, we can only represent the numbers 0 (0000 0000 in binary) to 127 (0111 1111 in binary).

This is not a problem as long we only send text in our e-mail message, which was the original purpose. The 7-bit limit was also intended to provide security by preventing unintended executable files from being sent to an unsuspecting e-mail recipient.

Then along came the World Wide Web and desktop computers and more and more people began to use the computer for business and recreational purposes, and they wanted to attach 8-bit binary files (such as images, word processor documents, and even executable programs) to their 7-bit SMTP messages. But binary files contain bytes that use all 8 bits.

Enter, the problem.

How do you transfer up to 256 different bytes (0-255) with only 128 different bytes (0-127)? Well the easiest way would have been to simply split each byte in half and make two bytes. That way any byte, such as 1101 1001 (in binary), would end up being two bytes, 0000 1101 and 0000 1001, each of which would never be larger than 15 (0000 1111 in binary). To convert back, all we would need to do is multiply the first byte by 16 and add . But you know computer programmers. They just shuttered at the idea of doubling the size of every binary file attached to an SMTP message. They just hated to waste band-width. And besides, until recently transfer speeds using older modems was very slow.

Enter, Base 64.

Several different schemes for encoding binary into SMTP-friendly text were devised, but the one that seems to be most popular lately is called "Base 64". In this scheme, three bytes of binary information are converted into four bytes of text. Here's how:

First, there are 64 "base" characters (hence, the name "Base 64"). They are:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

The base characters are indexed beginning with 0 and ending with 63. So base character 0 is "A", base character 1 is "B", and base character 63 is "/", to name a few.

Suppose we have three bytes of information that look like this:
aaaa aaaa bbbb bbbb cccc cccc
The bits of the first byte are represented by 8 letter a's, the bits of the second byte are represented by 8 letter b's, and the bits of the third byte are represented by 8 letter c's.

To convert three bytes into four, we group the bits into groups of 6, like this:
aaaa aa|aa bbbb| bbbb cc|cc cccc
And then we place each group into a new byte whose first two bits are zero (0), like this:
00aa aaaa 00aa bbbb 00bb bbcc 00cc cccc
The result is four new bytes, each of whose smallest possible value is 0 (0000 0000 in binary) and whose largest possible value is 63 (0011 1111 in binary).

The last thing we do is look up the base character whose index is the value of each new byte. This results in four characters, each chosen from the list of base characters.

For example, suppose our three bytes are 83, 17, and 204. In binary, they would look like this:
0101 0011 0001 0001 1100 1100
If we group their bits into groups of six and put each group into a new byte whose first two bytes are 0, we get:
0001 0100 0011 0001 0000 0111 0000 1100
These respective values are 20, 49, 7, and 12.

The four base characters located at these four indexes are "U", "x", "H", and "M".

Therefore, the Base 64 code for 83, 17, and 204 is "UxHM".

YOUR JOB

Your job is to write a Java program that accepts three integers as input and outputs the resulting four characters of Base 64 code. The three integers must be within the range of 0 to 255 inclusive to be valid. Input halts when one or more of the integers is equal to -1. If none of the integers is -1, but one or more is invalid, "INVALID" is displayed.

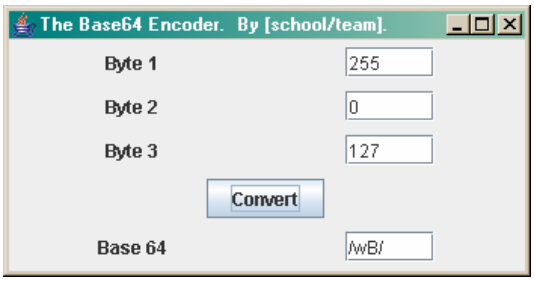
If you use a Console Application, you can get a maximum of 25 points if everything works correctly.

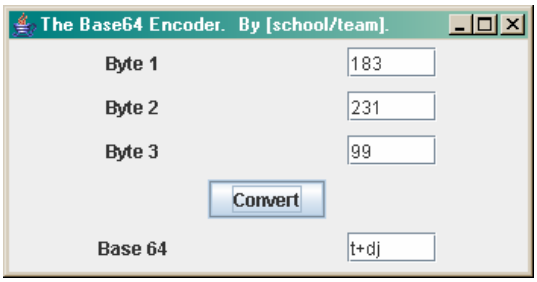
Sample Console run:

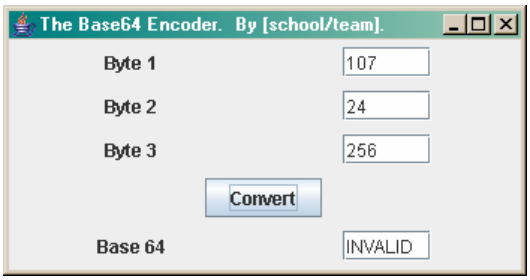
```
The Base64 Encoder. By [school/team name here].
Enter three bytes: 255 0 127
In base 64, that is /wB/
Enter three bytes: 183 231 99
In base 64, that is t+dj
Enter three bytes: 107 24 256
INVALID
Enter three bytes: -1 0 0
Press any key to continue ...
```

If you use a GUI and your program works correctly, you will receive 5 additional points. If you use a GUI and your program does not work correctly, you will not receive 5 additional points nor any portion thereof. A GUI version should not halt if the user enters -1, instead -1 should also be treated as invalid input.

Sample GUI runs:







The 8th Annual Newport News Computer Challenge

Wednesday, February 18, 2009

Java – Ruberic for Teams

Distance 3D (10 points)

	Max Points

Correct input/prompt:
Prompts and inputs 3 integers on one line and three on another.
Sample:
`Enter the three coordinates of the first  point: 1 -2 3`
`Enter the three coordinates of the second point: 1 -2 4`
	1

Output line is correctly formatted:
`"The distance from (x,y,z) to (a,b,c) is d.ddd"`
• Attempts to echo input (1)
Values for all six points are there, but not correctly formatted.
• Correctly echoes input (1)
Values for all six points are there and correctly formatted up to and including the word "is":
`"The distance from (x,y,z) to (a,b,c) is"`
• Formats distance correctly (1)
Three decimal places rounded to the nearest thousandth, padded with zeros if needed.
	3

Correct distance	6
TOTAL	10

Chinese Animal Zodiac Year Problem (10 points)

	Max Points
Heading displays program name and team name.	1
Correctly prompts user to enter a year from 1500 to 2999, any other year to quit.	1
Displays the correct answer for any valid western year. ("Rat", "Ox", etc.)	5
Displays the correct answer correctly formatted regardless of "was", "is", or "will be".	1
Displays the correct answer correctly formatted including "was", "is", or "will be".	1
If displays a correct answer for years in range, quits if any other year is entered.	1
TOTAL	10

Maya Calendar (30 points)

	Max Points
Heading displays program name.	1
Prompts user to input a Haab date. *1 point for specifying a Haab date and 1 point for indicating year range (0 to 9999).*	2
Rejects (or prevents) invalid input	2
Code indicates attempt to use all 19 Haab month names and all 20 Tzolkin day names. *(using arrays, ifs or switches, etc.)*	2
Test dates work correctly: *5 points for one date* *10 points for two dates* *15 points for three dates*	15
Tzolkin dates correctly formatted	2
Tzolkin dates correctly labeled "Tzolkin"	1
SUBTOTAL	25
Uses GUI AND input is successfully stored in variable(s) *(if Haab date is input in separate parts, each part must be successfully stored in a variable)*	5
TOTAL	30

Base 64 – Console Application (25 points)

	Max Points
Heading displays program name and team name.	1
Accepts three integers as input without crashing.	1
Input halts when one or more of the integers equals -1.	2
If none of the integers is -1, but one or more is invalid, "INVALID" is displayed.	2
Input repeats after a correct "INVALID" is displayed.	2
Input repeats after outputting base 64 characters of three valid bytes (whether or not output is correct).	2
Outputs correct base 64 values.	15
TOTAL	25

Base 64 – GUI Application (30 points)

	Max Points
Heading (title bar or label) displays program name and team name.	1
Provides 3 text fields, properly labeled, for input of 3 bytes, and a "Convert" button.	1
When "Convert" button is clicked:	
A reasonable attempt at a computed solution (even if wrong) is displayed, either in a text field, a label, or a popup.	2
Solution is properly labeled.	2
If one or more bytes is invalid, "INVALID" is displayed.	2
If all bytes are valid, "INVALID" is NOT displayed.	2
Outputs correct base 64 values.	15
Correct program uses a GUI. (add 5 only if score prior to this is 25)	5
TOTAL	30