

Division 1 - Problem #1 – Cell Phone Billing

Playing games on cell phones is fun, but it can be expensive. The Equinox Cell Game plan is a cell phone gaming plan that charges players based on the number of minutes the player plays each month. Players get a certain number of minutes for a fixed fee, but are charged an increasing amount for every minute over the limit. The billing structure is:

The first 600 minutes – \$10.00 (1000 cents),
every minute over 600 – add another \$0.01, and
every minute over 1000 – add another \$0.02.

Players that plays 1200 minutes would pay \$20.00, or 2000 cents. They pay \$10 for the first 600 minutes, another \$4 (at 1 cent a minute) for minutes 601-1000, and another \$6 for minutes 1001-1200 (at 3 cents a minute).

A player that plays 1 minute would pay the minimum \$10 charge.

Your task is to write a program that reads a single integer input from the keyboard (representing a number of minutes that a player played), and outputs a single integer representing how much this player should be billed (in cents).

- Use the console for all input / output (no GUIs or file I/O).
- Do not prompt the user, and do not print out extra information.
- The input value will be within the range 1...100000.
- The output value must be an integer.

Example 1:

Input:

47

Output:

1000

Example 2:

Input:

1829

Output:

3887

Example 3:

Input:

701

Output:

1101

Division 1 - Problem #2 – Cookies

Writing computer games at the Equinox Cell Game company can be a lot of fun, but it is easy for programmers to get the munchies while they program and play games. One afternoon the programmers decide to bake a batch of cookies, and they ask you to do it. After baking the cookies, a coworker finds that two of your handmade cookies have mashed together in the cooking stage. There is nothing that game developers hate more than poorly calculated cookie collision detection, and they ask you to write a program to avoid this calamity in the future.

You are to come up with an algorithm to determine whether cookies will mash together when they are cooked. You will be given the center position and final radius of each cookie on a cookie sheet (assume they are circular). It is up to you to determine whether any of the cookies collide. (There must be a small amount of space between every cookie; they may not touch at all.)

Write a program that reads cookie information from the keyboard and outputs a single word indicating if any two cookies collide. If any two of the cookies on the sheet touch each other, output “collision” to the console. If none of the cookies touch each other then output “no collision” to the console. (Quotation marks are for clarity, do not output them.)

The input will be specified in two parts. First, the user will type a number of cookies. Second, for every cookie, the user will specify the center and size of a cookie using an X position, a Y position, and a radius.

- Use the console for all input / output (no GUIs or file I/O).
- Do not prompt the user, and do not print out extra information.
- All inputs will be integers within the range 1...300.
- The output value must be either “collision” or “no collision”.

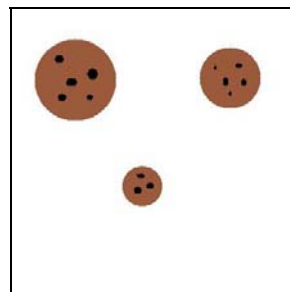
Example 1:

Input:

```
3
76 86 50
256 82 35
157 214 25
```

This input represents the following situation:

Position	Radius
(76, 86)	50
(256, 82)	35
(157, 214)	25



Output:

no collision

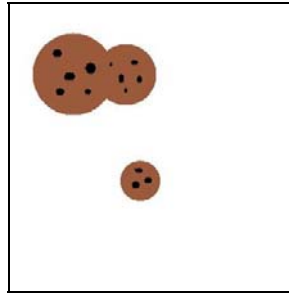
Example 2:

Input:

```
3
76 86 50
144 82 35
157 214 25
```

This input represents the following situation:

Position	Radius
(76, 86)	50
(144, 82)	35
(157, 214)	25



Output:

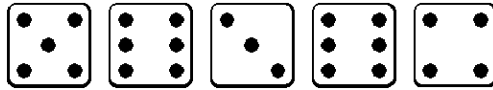
collision

These sample inputs show three cookies. Please remember that there may be as few as one cookie or as many as 300 cookies, as specified by the first integer of the input.

Division 1 - Problem #3 – Dice game

Every year when spring arrives a group of programmers likes to play a dice game. As they play, they are not sure if they are always choosing the best score for their dice rolls. They have asked you to help out by writing a program that will help them decide how to score their rolls.

The game is played by repeatedly rolling five dice. A typical roll might look like this:



For any roll of five dice you have the option of scoring your roll using one of twelve categories:

<u>Category</u>	<u>How to score it</u>
"ones"	score equal to the sum of only the 1's in the roll
"twos"	score equal to the sum of only the 2's in the roll
"threes"	score equal to the sum of only the 3's in the roll
"fours"	score equal to the sum of only the 4's in the roll
"fives"	score equal to the sum of only the 5's in the roll
"sixes"	score equal to the sum of only the 6's in the roll
"three of a kind"	score equal to the sum of all 5 dice if at least three dice are the same
"four of a kind"	score equal to the sum of all 5 dice if at least four dice are the same
"small straight"	score 30 points, but only if when sorted, at least four of the dice are in consecutive order
"large straight"	score 40 points, but only if when sorted, all five dice are in consecutive order
"full house"	score 25 points if two dice show the same number and the remaining three dice show another number
"flush"	score 50 points if all five dice show the same number

One possibility for scoring the dice roll above would be to use the “sixes” category. This would earn twelve points (there are two sixes). Because the roll has a three, a four, a five, and a six, a better category to use would be “small straight” – it would earn 30 points.

You are to write a program that inputs five integers from the keyboard. These represent the values of five dice. Print out to the console the name of the category that will earn the most possible points. In the event of a tie, choose the category that occurs first in the list above.

- Use the console for all input / output (no GUIs or file I/O).
- Do not prompt the user, and do not print out extra information.
- The input will be exactly five integers in the range 1...6.
- The output should be a correctly-spelled category name from the list above.

(Examples are on the back of this page.)

Example 1:

Input:

1 6 4 6 5

Output:

sixes

Example 2:

Input:

4 5 5 4 5

Output:

full house

Example 3:

Input:

3 1 3 2 1

Output:

threes

Example 4:

Input:

2 6 5 3 4

Output:

large straight

Division 1 – Web design problem – Utah peaks

On the back of this page is a text description of mountain peaks and trails in Utah. You are to create a webpage that more clearly conveys this information. You should try to organize the information in an easy-to-use manner, and you should download or draw artwork that improves the look of your webpage.

Use webpage development application(s), a text editor, or raw HTML, CSS, and/or JavaScript (if you wish) to create a webpage. Place your webpage in the file specified on the front instruction page. (To keep other teams from copying your information, do not tell this filename to other teams.)

Important: All of your web page files *must* be within your “public_html” directory, and your webpage must have the correct name. See the front instruction page for more details.

This problem is intended to be open-ended. You are allowed to be creative in how you express this information. You are allowed to get supplementary information from the Internet. You are allowed to use and modify images that you find on the Internet. You may create several linked-together pages, or you may put all the information on a single page.

There are a few restrictions:

- Your text (and tables) must be in your own words. Do not cut-and-paste text or tables from web pages.
- Your web pages must be your own design. Do not copy and/or modify existing web pages. (You may use templates built-in to an application.)
- You may not link outbound to other web pages. Any links in your webpage must be internal, relative links. Absolute links are not allowed. (No “http://www.eng.utah.edu/~myschool/image.jpg”, use “image.jpg” instead.)
- You may not copy scripts of any kind from the Internet. You may not do server-side processing, send email, write data to files, or use Java Applets.

At the end of the contest, your webpage directory will be copied into the judge’s directory and judged from there. Please, no absolute links or your webpage will not display.

Your webpage will be judged and scored based on accuracy of the information, artistic design, use of language, completeness, and overall usefulness.

Creatively present the following information on a webpage:

The Wasatch Mountains are a narrow mountain range running north to south for about 200 miles from the Idaho border to the town of Nephi in central Utah. The Wasatch Mountains are located on the western edge of the Rocky Mountains.

The highest peak in the Wasatch is Mt. Nebo, located at the southern end of the range. Mt. Nebo rises more than 1670 meters from the land surrounding it. The elevation of its highest peak is 3636 meters. A strenuous hike with 1158 meters' elevation gain will bring hikers to the summit. Parts of the mountain are covered in snow 10 months of the year.

The second highest mountain along the Wasatch is Mt. Timpanogos, at 3582 meters, which is found east of Orem and Provo, Utah. A hike to the peak from the major trailheads has 1450-1500 meters' elevation gain and may take an experienced hiker 10-11 hours round trip. Mt. Timpanogos is said to have the shape of a sleeping Indian maiden. A series of beautiful underground caves are found on the northwest slope of the mountain.

Lone Peak is not as tall, at 3430 meters, but it is known for being quite rugged. A hike to the summit requires an elevation gain of 1845 meters. Lone Peak is situated between Little Cottonwood and American Fork Canyons. It is the centerpiece of Utah's first congressionally designated Wilderness Area. Lone Peak Wilderness Area was established in 1977.

Mt. Olympus and Grandeur Peak are found just east of Salt Lake City. Both mountains are very popular for hiking and are accessible to beginning and intermediate hikers. The summit of Mt. Olympus is 2751 meters in elevation; the easier trail has an elevation gain of 1234 meters. Grandeur Peak is 2529 meters tall. A favorite route to the top is trail all the way and rises just under 800 meters in elevation. The summit can be reached in less than three hours.

Mt. Ben Lomond, located east of Ogden, Utah, is one of the better-known mountains in the northern Wasatch. The elevation of the summit is 2960 meters. Hiking trails are long but not overly steep. A round trip to the top will take 6-7 hours. Spectacular views of the Great Salt Lake and Willard Bay reward those reaching the summit. Parts of the trail are suitable for mountain biking in summer and cross-country skiing in the winter.

Many of the mountain peaks in the Wasatch have interesting histories to their names. Some are named for more famous mountains located elsewhere in the world. Mt. Olympus, for example, is named for Mount Olympus in Greece, the home of the principal gods in Greek mythology. Mt. Nebo is named after Mount Nebo of Jordan, said to be the place where Moses died. Mount Ben Lomond was named for a mountain in the highlands of Scotland. The origin of the name Timpanogos is not entirely clear; some say it is a Paiute word for "river of rock", others say it comes from the name of a legendary Indian maiden. Other mountains are named for their appearance, such as Lone Peak and Grandeur Peak.