Exercise: For Loops

1.Back to the Past

Josh is 18 years old and receives a legacy that consists of **X amount of money and time machine**.

He decided to return **until 1800**, but he did not know if the money would suffice to live without working.

- Write a program that calculates whether Josh will have enough money to not have to work until a certain year.
- Assuming that for every even year **(1800, 1802, etc.)** will spend **12 000 dollars**.
- For each odd **(1801, 1803, etc.)** will spend **12 000 + 50 * \\**[the years age in a given year\\].

Input

The input is read from the console and contains exactly 2 lines:

- Legacy money – real number in range **\\[1.00... 1,000,000.00\\]**
- Year to be lived (inclusive) – integer in range **\\[1801... 1900\\]**

Output

Print on the console a **single** line. The amount must be formatted to **two decimal digits**:

- If the money are enough:
    - "Yes! He will live a carefree life and will have \\{N\\} dollars left." – where N are the money that will remain
- If the money are **NOT** enough:
    - "He will need \\{M\\} dollars to survive." - where M are money he needs

Example

Input	Output	Comments
50000	Yes! He will live a carefree life and will have 13050.00 dollars left.	- 1800 -> even year; Josh spends 12000 dollars - 50000 – 12000 = 38000 dollars remain - 1801 -> odd year - he spends 12000 + 19 * 50 = 12000 + 950 = 12950 dollars - 38000 – 12950 = 25050 dollars remain
1802		- 1802 -> even year - he spends 12000 dollars - 25050 – 12000 = 13050 dollars remain

2. Bills

Description

Write a program to calculate the average cost per month for a family over a period of time. For each month the costs are as follows:

- For **electricity** - every month the bill is different and will be read from the console
- For **water** - 20$
- For **Internet** - 15$
- For **others** - sum the bills for electricity, water and Internet and add 20%

For each bill, you need to calculate how much total is paid for **all months**.

Input

The input is read from the console:

- The **months** for which the average cost is searched – integer in range **\[1... 100\]**
- For each month – the bill for electricity – a real number in range **\[1.00... 1000.00\]**

Output

Print to the console **5** rows:

- "Electricity: \{electricity for all months\} $"
- "Water: \{water for all months\} $"
- "Internet: \{Internet for all months\} $"
- "Other: \{Other for all months\} $"
- "Average: \{Average all costs per month\} $ "

All bills should be **formatted to the 2nd digit after the decimal point**.

Example

Input	Output
`5`	`Electricity: 447.16 $`
`68.63`	`Water: 100.00 $`
`89.25`	`Internet: 75.00 $`
`132.53`	`Other: 746.59 $`
`93.53`	`Average: 273.75 $`
`63.22`	

Comments

For **5** months:

- **Electricity** -> 68.63 \+ 89.25 \+ 132.53 \+ 93.53 \+ 63.22 = **447.16**$
- **Water** -> 5 months * 20$ = 100$
- **Internet** -> 5 months * 15$ = 75$
- **Others**:
  - 68.63+20+15) + 20% = 124.356
  - (89.25+20+15) + 20% = 149.1
  - (132.53+20+15) + 20% = 201.036
  - (93.53+20+15) + 20% = 154.236
  - (63.22+20+15) + 20% = 117.864
- ***Total = 746.592$**
- **Average** bills per month = (447.16+100+75+746.592)/5 = **273.7504**$

3. Hospital

Description

- **For a period of time, patients are arriving every day** in the hospital for examination.
- It has initially **7** doctors.
- Each doctor can only review **one patient per day**, but sometimes there is a shortage of doctors, so other patients are sent to other hospitals.
- **Every third day** the hospital makes calculations and **if the number of unreviewed patients is greater than the number of reviewed, one more doctor is appointed**.
- As the **appointment of the doctor occurs before the intake of patients for the day**.
- Write a program that calculates the number of reviewed and unreviewed patients for the given period.

Input

The input is read from the console and contains:

- **The period** for which you need to perform calculations - integer in range \[1... 1000\]
- On the **following lines (equal to the number of days)** – the number of patients arriving for review for the current day - integer in range \[0... 10 000\]

Output

Print on the console 2 lines:

- First line: "Treated patients: \{Number of patients reviewed\}."
- Second line: "Untreated patients: \{Number of unreviewed patients\}."

Example

Input	Output
4	`Treated patients: 23.`
7	`Untreated patients: 21.`
27	
9	
1	

Comments

- Day 1: 7 treated and 0 untreated patients for the day
- Day 2: 7 treated and 20 untreated patients for the day
- Day 3: Until now, the patients treated were 14 and untreated – 20 – > A new doctor is appointed – > 8 treated and 1 untreated patient for the day
- Day 4: 1 treated and 0 untreated patient for the day
- Total: 23 treated and 21 untreated patients

4. Numbers Ending in 7

Description

- Write a program that prints the numbers in the range **\[1...1000\], which end in 7**.
- Print the numbers on a **single** line, separated by a **single space**.

Example

Input	Output
(no input)	7 17 27 ... 997

5. Odd / Even Position

Description

Write a program that reads the **n** - **count** of **numbers** entered by the user, and **calculates** the **amount** of the **minimum** and **maximum** numbers of **odd** and **even** positions (counting from **1**).

If there **is no** minimum/maximum element print "**No**".

Input

- On the first input line read count of numbers - N - integer \[0...100\]
- On every next line read a number - floating-point number \[-100...100\]

Output

The output should be **formatted** in the following form:

"Odd Sum:" + \{**sum** of the numbers on **odd** positions\}

"Odd Min:" + \{**minimum** value of the numbers of **odd** positions\} / \{**"No"**\}

"Odd Max:" + \{**maximum** value of the numbers of **odd** positions\} / \{**"No"**\}

"Even Sum:" + \{**sum** of the numbers of **even** positions\}

"Even Min:" + \{**minimum** value of the numbers of **even** positions\} / \{"**No**"\}

"Even Max:" + \{**maximum** value of the numbers of **even** positions\} / \{**"No"**\}

Each number should be formatted to the second decimal point.

Example

Input	Output
5	Odd Sum: 8.00 Odd Min: -3.00 Odd Max: 8.00 Even Sum: 9.00 Even Min: -2.00 Even Max: 11.00
3 -2 8 11 -3	

Input	Output
0	Odd Sum: 0.00 Odd Min: No Odd Max: No Even Sum: 0.00 Even Min: No Even Max: No

6. Grades

Description

Write a program to **calculate statistics** of the exam **grades**.

In the beginning, the program receives the **number** of **students** attended the examination and **its grade for each student**.

In the end, the program should print **the percentage of students** with grades between **2.00 and 2.99**, between **3.00 and 3.99**, **4.00 and 4.99**, and between **5.00 or more**.

Also the **average grade** of the exam.

Input

Read from the console a **series of numbers**, each on a **separate** line:

- The **first** line - **number** of students **attended** the exam - an integer in the range \[1 ... 1000\]
- **For each student on a separate line** - grade of the exam - real number in the range \[2.00 ... 6.00\]

Output

Print on the console **5 rows** that contain the following information:

- "Top students: \{percent student with grade 5.00 or more\}%"
- "Between 4.00 and 4.99: \{between 4.00 and 4.99 inclusive\}%"
- "Between 3.00 and 3.99: \{between 3.00 and 3.99 inclusive\}%"
- "Fail: \{less than 3.00\}%"
- "Average: \{average grade\}"

All numbers must be **formatted** to the s**econd decimal place**.

Example

Input	Output
10	Top students: 30.00%
3.00	Between 4.00 and 4.99: 30.00%
2.99	Between 3.00 and 3.99: 20.00%
5.68	Fail: 20.00%
3.01	Average: 4.06
4	

4	
6.00	
4.50	
2.44	
5	

Comments

- Students with grade 5 and more – three = 30% of 10
- Between 4 and 4.99 – three = 30% of 10
- Between 3 and 3.99 – two = 20% of 10
- Less than 3 – two = 20% of 10

Average grade: 3 + 2.99 + 5.68 + 3.01 + 4 + 4 + 6 + 4.50 + 2.44 + 5 = 40.62 / 10 = 4.062