Exercise Youth Hostel (1)

- The hostel in which you plan to spend the night tonight offers very interesting rates, as long as you do not arrive too late. Housekeeping finishes preparing rooms by noon, and the sooner guests arrive after noon, the less they have to pay. Write a C program that calculates your price to pay based on your arrival time.
- Your program will read an integer indicating the number of hours past noon of your arrival. For example, 0 indicates a noon arrival, 1 a 1pm arrival.
- The base price is 10 yuan, and 5 yuan are added for every hour after noon. Thankfully the total is capped at 53 yuan, so you'll never have to pay more than that. Your program should print the price (an integer) you have to pay, given the input arrival time.

Exercise Youth Hostel (2)

- Test case 1 :
  Input:
  `7`
  Output:
  `45`

- Test case 2 :
  Input:
  `10`
  Output:
  `53`

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise Youth Hostel Hints

- Declare variables to store hour, price
- Use scanf, %d, &hour to read the hour
- Compute the price according to the problem description
- Use if to check whether the price is more than 53
  - if true, reset the price back to 53
- Print the price
  - don't forget to end with new line character

Exercise Tug of War (1)

- You want to bet on the final match of the Tug of War Championship.
- Prior to the match the names and weights of the players are presented, alternating by team (team 1 player 1, team 2 player 1, team 1 player 2, and so on). There is the same number of players on each side. You record the player weights and calculate a total weight to inform your bet.
- Your program should first read an integer indicating the number of members per team. Then, the program should read the player weights **alternating** by team.
- The program should then display which team has an advantage, that is, the team that has a greater total weight. Then, display the total weight for each team, as shown in Test case 1.

Exercise Tug of War (2)

- Test case 1 :

  Input:

  ```
  4
  110
  106
  113
  102
  112
  121
  117
  111
  ```

  Output:

  ```
  Team 1 has an advantage
  Total weight for team 1: 452
  Total weight for team 2: 440
  ```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise Tug of War Hints

- Declare integer counter and variables to store read values
- Declare and initialize two integer sums to 0
- Read number of players
- Loop using for number of players times
  - read first weight, add it to first sum
  - read second weight, add it to second sum
- If first sum is greater or the same than second sum
  - print team 1 advantage
- Else
  - print team 2 advantage
- Print total weight of team 1 and team 2

Exercise Fantastic Cookie Recipe

- Your grandparents gave you a fantastic cookie recipe. There are 10 ingredients in the recipe and the quantities needed for each of them are given as input (in grams).
- Your program must read 10 integers (the quantities needed for each of the ingredients, in order) and store them in an array. It should then read an integer which represents an ingredient's number (between 1 and 10), and output the corresponding quantity.
- Test case 1 :
  Input:
  ```
  200 180 650 15 600 36 420 1 370 2
  5
  ```
  Output:
  ```
  600
  ```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise Fantastic Cookie Recipe Hints

- Declare an integer array of size 10
- Declare integer variables that you will need, including an index
- Use for to read 10 times
  - Use scanf, "%d", and &array[i] to read and store the values
- Use scanf to read the ingredient number
- Use printf and array access to print it
  - Remember that the array index starts from 0
  - While the ingredient number starts from 1
  - So you need to make an adjustment to the index!

Exercise Balanced Train Wagons (1)

- You are responsible for a train of goods consisting of several boxcars. You realize that some boxcars are overloaded and weigh too heavily on the rails while others are dangerously light. So you decide to spread the weight more evenly so that all the boxcars have exactly the same weight without changing the total weight. You write a program which helps you in the distribution of the weight.
- Your program should first read the number of cars to be weighed (integer) followed by the weights of the cars (doubles). Then your program should calculate and display how much weight to **add** or **subtract** from each car such that every car has the **same** weight. The total weight of all of the cars should not change. These additions and subtractions of weights should be displayed with ***one decimal place***.
- You may assume that there are no more than 20 boxcars.

Exercise Balanced Train Wagons (2)

- Test case 1 :

  Input:

  ```
  5
  40.0
  12.0
  20.0
  5.0
  33.0
  ```

  Output:

  ```
  -18.0
  10.0
  2.0
  17.0
  -11.0
  ```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise Balanced Train Wagons Hints

- Since there are no more than 20 boxcars, create double array weights of size 20
- Declare your integer and double variables, in particular, declare and initialize a double variable sum to 0.0
- Use scanf to read the number of boxcars
- Use for to loop boxcars times
  - read the input into weight[i]
  - add weight[i] into sum
- Compute the average
- Use for once more like the above
  - print the weight correction which is the average minus weight[i] with one decimal point and a new line