Write a program to control the packing and shipping of pallets. You have **N** pallets available, each of which can hold up to 200 pounds. You should use an array to keep track of how much weight in currently on each of the **N** pallets. Next, you are given the function `get_package()` that returns the weight of a single package to be placed on a pallet.

Based on the above description, what must the signature for get_package() look like. Write it here: [Note you are not being asked to implement it!]

____

Write one line of code that shows how you would use get_package():

int packageWeight = get_package();

You should place the package (more specifically, the packages weight) on any pallet such that the amount of weight on the pallet does not exceed 200 pounds**. [You don't need to keep track of the actual packages, just the weights.]**

If the package weighs too much to be placed on any of the pallets, then you should determine which pallet is the heaviest (see note on `max()` below) and "ship it" (ie: display a message saying you are shipping pallet #X containing Y pounds, and then set the weight on the pallet to 0 (as the contents have now been removed)) - then add the current package (its weight) to the pallet that was just shipped.

Note, you may use the `max()` function to determine which pallet has the most weight. `max()`, when given a list of items as input, will return the **index** of the largest item in the list. Show how you would use the max function here: [Create a variable to represent the weights of the packets and then use the max() function on it. You should write two lines of code here:]

double pallets[10];

int heaviestPackage = max(pallets);

When a package is added to a pallet, if the pallet's weight exceeds 180 pounds, the pallet should be shipped (This just means: display a message and set the pallet's weight back to 0). Your code should continue until the `get_packages()` function returns a 0 weight (ie, no packages are available) at which point it ships all remaining pallets.

Example 1) You start with 3 pallets (ie. weights on pallets are: [0, 0, 0]). You then call `get_package()` which tells you that you have a new package weighing 150 pounds. Since it will fit on pallet 1, you place it there resulting in [150, 0, 0]. Since the pallet does not contain 180 or more pounds, you should continue with the next package. (Do **not** assume you have 3 pallets. You will have N pallets where N has already been set for you.)

Example 2) Your 3 pallets have [150, 120, 0] weight on them. Again you call `get_package()` which tells you that your next package weighs 170 pounds. Since it will not fit on pallets 1 or 2 (because the pallets' weight would then exceed 200 pounds), you should place it on the 3rd pallet resulting in [150, 120, 170].

Example 3) Your pallets currently have [150, 120, 170] pounds on them. The next package weighs 100 pounds. Since no pallet can hold an additional 100 pounds, you must determine which pallet is the heaviest (using `max`) and "ship" it:

In this case you would have just shipped pallet 3 (to do so: display a message saying you shipped pallet # and its weight). Now you can place the current 100 pound package on the newly empty 3rd pallet, resulting in weights of [150, 120, 100]