|  |  |  |
| --- | --- | --- |
| **American University of Sharjah**  **College of Engineering**  Dept of Computer Science & Engg  P. O. Box 26666  Sharjah, UAE | A picture containing logo  Description automatically generated | **Instructors:** Dr.Aliaa Moualla  **Lab Instructor:** Sameer Alawnah  **Office:** EB1-0012C  **Phone**: 971-6-515-4940  **e-mail**: salawnah@aus.edu  **Semester**: Spring 2024 |

**CMP 220L - Programming II**

**Lab #1 –****Functions Call by value vs Call by Reference, and Dynamic Memory Allocation**

**Objectives:**

* To practice Functions, call by value vs call by reference.
* To practice dynamic memory allocation.

Using Visual Studio 2022, write the below programs, compile and provide screenshots of output.

Note: you are required to submit copy of the code + screenshots of program run for each exercise.

**Exercise #1**

A zookeeper is responsible for counting the number of herbivores and carnivores in the zoo.

Write a C++ program to help the zookeeper keep track of the lion’s health.

* Define the function **void recordAnimal(int& carnCount, int& herbCount, char food)** which will increment the value of either **carnCount** or **herbCount** based on the **food** ('H' for herbs or 'M' for meat).
* In the **main**:
  + Declare two integers **carnCount** and **herbCount** and initialize them to zero.
  + Use a **while** loop and ask the user to input the animal food. The loop should stop When the user enters ‘E’, you can make it an indefinite loop and use break to exit it.
  + If the user inputs a status other than 'H’, 'M' or 'E', print an error message and skip the iteration without calling the function.
  + After the loop, print the values of **carnCount** and **herbCount**.
* In the **recordAnimal** function, if you replace char food with const char food, how it will affect the program? Is it safe to do so (will the value of the variable reference passed to the input changed if we change input)?

**Sample output**

Animal 1: is eating Herbs (H)) or Meat (M)? (E) to exit : H

Animal 2: is eating Herbs (H)) or Meat (M)? (E) to exit : M

Animal 3: is eating Herbs (H)) or Meat (M)? (E) to exit : H

Animal 4: is eating Herbs (H)) or Meat (M)? (E) to exit : M

Animal 5: is eating Herbs (H)) or Meat (M)? (E) to exit : A

Invalid food, please enter H, M or E

Animal 5: is eating Herbs (H)) or Meat (M)? (E) to exit : M

Animal 6: is eating Herbs (H)) or Meat (M)? (E) to exit : M

Animal 7: is eating Herbs (H)) or Meat (M)? (E) to exit : M

Animal 8: is eating Herbs (H)) or Meat (M)? (E) to exit : H

Animal 9: is eating Herbs (H)) or Meat (M)? (E) to exit : H

Animal 10: is eating Herbs (H)) or Meat (M)? (E) to exit : E

Animals report:

This Zoo has 5 Carnivores and 4 Herbivores

#include <iostream>

#include <vector>

using namespace std;

void recordAnimal(int& carnCount, int& herbCount, char food, int animalNumber);

void recordAnimal(int& carnCount, int& herbCount, char food, int animalNumber)

{

if (food == 'H')

{

cout << "H\n";

herbCount++;

}

else if (food == 'M')

{

cout << "M\n";

carnCount++;

}

else if (food == 'E')

{

cout << "\n";

}

else

{

cout << "Invalid food, please enter H, M, or E";

}

}

int main() {

int carnCount = 0;

int herbCount = 0;

int animalNumber = 0;

char food;

while (true)

{

animalNumber++;

cout << "Animal " << animalNumber << " is eating Herbs (H) or Meat(M)?";

cin >> food;

if (food == 'E')

break;

if (food != 'H' && food != 'M')

{

cout << "Invalid Food, please enter H, M, or E";

animalNumber--;

}

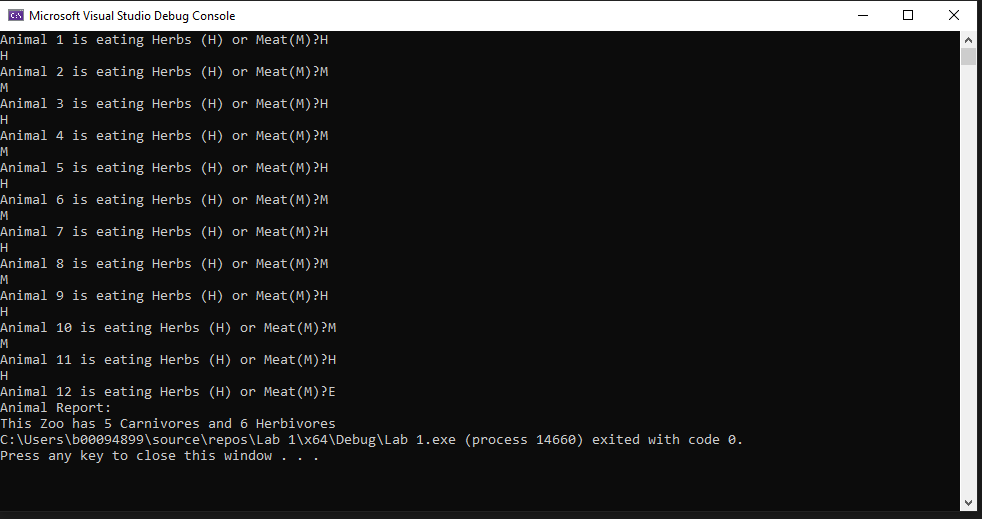
recordAnimal(carnCount, herbCount, food, animalNumber);

}

cout << "Animal Report:\nThis Zoo has " << carnCount << " Carnivores and " << herbCount << " Herbivores";

return 0;

}



**Exercise #2**

A zookeeper wants to found how much meat he needs to feed the carnivores in the zoo, each carnivore should have its own portion. Write a C++ program that allows the zookeeper to input and store the portion weight needed for every carnivore. The program should follow these steps:

1. Ask the zookeeper to input the number of carnivores in the zoo.
2. Declare a dynamic array named **portionsWeights** to store the weighs the portions.
3. Initialize the array by asking the zookeeper to input the weight of each portion.
4. Determine the weight of the lightest and heaviest portions and print them.
5. Remember to deallocate the memory used by the **portionsWeights** array.

**Sample output**

Enter the number of carnivores : 6

Enter the portions weights for 6 carnivores:

12.3 33.2 1.2 25.1 6.1 0.8

The weight for the heaviest portion is 33.2 and the weight of the lightest portion is 0.8

#include <iostream>

using namespace std;

int main() {

int carnnumber;

cout << "Enter the number of carnivores: ";

cin >> carnnumber;

double\* portionsWeights = new double[carnnumber];

cout << "Enter the portions weight for " << carnnumber << " carnivores: ";

for (int i = 0; i < carnnumber; i++)

{

cin >> portionsWeights[i];

}

double lightestWeight = portionsWeights[0];

double heaviestWeight = portionsWeights[0];

for (int i = 1; i < carnnumber; i++)

{

if (portionsWeights[i] < lightestWeight)

lightestWeight = portionsWeights[i];

}

for (int i = 1; i < carnnumber; i++)

{

if (portionsWeights[i] > heaviestWeight)

heaviestWeight = portionsWeights[i];

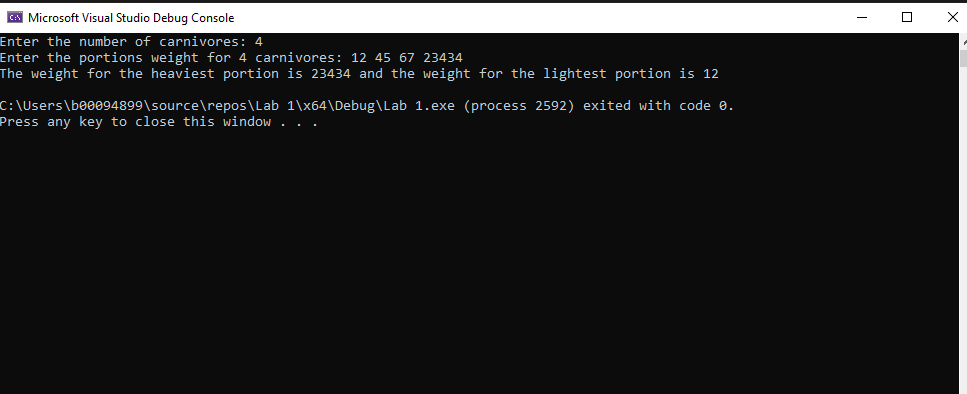
}

cout << "The weight for the heaviest portion is " << heaviestWeight << " and the weight for the lightest portion is " << lightestWeight << "\n";

delete[] portionsWeights;

return 0;

}



**Exercise #3**

Redo Exercise #2, but use a vector instead of the dynamic array.

#include <iostream>

#include <vector>

using namespace std;

int main() {

int carnnumber;

double weight;

cout << "Enter the number of carnivores: ";

cin >> carnnumber;

vector<double> portionsWeights;

cout << "Enter the portions weight for " << carnnumber << " carnivores: ";

for (int i = 0; i < carnnumber; i++)

{

cin >> weight;

portionsWeights.push\_back(weight);

}

double lightestWeight = portionsWeights[0];

double heaviestWeight = portionsWeights[0];

for (int i = 1; i < carnnumber; i++)

{

if (portionsWeights[i] < lightestWeight)

lightestWeight = portionsWeights[i];

if (portionsWeights[i] > heaviestWeight)

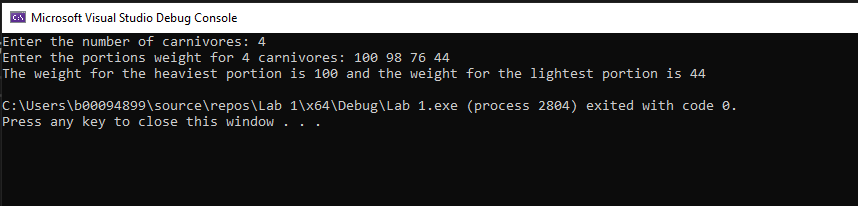
heaviestWeight = portionsWeights[i];

}

cout << "The weight for the heaviest portion is " << heaviestWeight << " and the weight for the lightest portion is " << lightestWeight << "\n";

return 0;

}

****