

Activity No. 2

ARRAYS, POINTERS AND DYNAMIC MEMORY ALLOCATION

Course Code: CPE010	Program: Computer Engineering
Course Title: Data Structures and Algorithms	Date Performed: 7/31/25
Section: CPE010 – CPE21S4	Date Submitted: 7/31/25
Name(s): QUIYO, ANGELO M.	Instructor: Jimlord Quejado

6. Output

```
Handicraft activity 2.2.cpp
1 #include <iostream>
2 #include <string.h>
3
4 class Student{
5 private:
6     std::string studentName;
7     int studentAge;
8
9 public:
10
11     Student(std::string newName = "John Doe", int newAge = 18){
12         studentName = std::string(newName);
13         studentAge = newAge;
14         std::cout << "Constructor Called." << std::endl;
15     }
16
17     ~Student(){
18         std::cout << "Destructor Called." << std::endl;
19     }
20
21     //Copy Constructor
22     Student(const Student &copyStudent){
23         std::cout << "Copy Constructor Called" << std::endl;
24         studentName = copyStudent.studentName;
25         studentAge = copyStudent.studentAge;
26     }
27     //Display Attributes
28     void printDetails(){
29         std::cout << this->studentName << " " << this->studentAge << std::endl;
30     }
31
32
33     int main(){
34         Student student1("Roman", 28);
35         Student student2(student1);
36         Student student3;
37         student3 = student2;
38
39         return 0;
40
41
42
43 }
```

```
Constructor Called.
Copy Constructor Called
Constructor Called.
Destructor Called.
Destructor Called.
Destructor Called.

Process exited after 8.01534 seconds with return value 0
Press any key to continue . . .
```

Screenshot	<pre>Handsonactivity22.cpp</pre> <pre> 1 #include <iostream> 2 #include <string.h> 3 4 class Student{ 5 private: 6 std::string studentName; 7 int studentAge; 8 9 public: 10 11 Student(std::string name="John Doe", int age=18){ 12 studentName = std::string(name); 13 studentAge = age; 14 std::cout << "Constructor Called." << std::endl; 15 } 16 17 ~Student(){ 18 std::cout << "Destructor Called." << std::endl; 19 } 20 21 //Copy Constructor 22 Student(const Student &copyStudent){ 23 std::cout << "Copy Constructor Called." << std::endl; 24 studentName = copyStudent.studentName; 25 studentAge = copyStudent.studentAge; 26 } 27 //Display Attributes 28 void printDetails(){ 29 std::cout << this->studentName << " " << this->studentAge << std::endl; 30 } 31 32 int main(){ 33 const size_t j = 5; 34 Student studentList[j] = {}; 35 std::string namesList[j] = {"Carly", "Freddy", "Sam", "Zack", "Cody"}; 36 int agesList[j] = {15, 16, 18, 19, 20}; 37 38 return 0; 39 } 40 }</pre> <pre>Constructor Called. Constructor Called. Constructor Called. Constructor Called. Constructor Called. Destructor Called. Destructor Called. Destructor Called. Destructor Called. Destructor Called. Process exited after 0.01038 seconds with return value 0 Press any key to continue . . .</pre>
Observation	

Table 2-1. Initial Driver Program

Screenshot	<pre>Handsonactivity22.cpp</pre> <pre> 1 #include <iostream> 2 #include <string.h> 3 4 class Student{ 5 private: 6 std::string studentName; 7 int studentAge; 8 9 public: 10 11 Student(std::string name="John Doe", int age=18){ 12 studentName = std::string(name); 13 studentAge = age; 14 std::cout << "Constructor Called." << std::endl; 15 } 16 17 ~Student(){ 18 std::cout << "Destructor called." << std::endl; 19 } 20 21 //Copy Constructor 22 Student(const Student &copyStudent){ 23 std::cout << "Copy Constructor Called." << std::endl; 24 studentName = copyStudent.studentName; 25 studentAge = copyStudent.studentAge; 26 } 27 //Display Attributes 28 void printDetails(){ 29 std::cout << this->studentName << " " << this->studentAge << std::endl; 30 } 31 32 int main(){ 33 const size_t j = 5; 34 Student studentList[j] = {}; 35 std::string namesList[j] = {"Carly", "Freddy", "Sam", "Zack", "Cody"}; 36 int agesList[j] = {15, 16, 18, 19, 20}; 37 38 for(int i = 0; i < j; i++){ //Loop A 39 Student *ptr = new Student(namesList[i], agesList[i]); 40 studentList[i] = *ptr; 41 } 42 43 for(int i = 0; i < j; i++){ //Loop B 44 studentList[i].printDetails(); 45 } 46 47 return 0; 48 }</pre> <pre>Constructor Called. Constructor Called. Carly 15 Freddy 16 Sam 18 Zack 19 Cody 16 Destructor Called. Destructor Called. Destructor Called. Destructor Called. Destructor Called. Process exited after 0.01077 seconds with return value 0 Press any key to continue . . .</pre>
Observation	

LoopA	<pre> int main() { const size_t j = 5; Student studentList[j] = {}; std::string namesList[j] = {"Carly", "Freddy", "Sam", "Zack", "Cody"}; int ageList[j] = {15, 16, 18, 19, 16}; for(int i = 0; i < j; i++){ //loop A Student *ptr = new Student(namesList[i], ageList[i]); studentList[i] = *ptr; } </pre>
Observation	
LoopB	<pre> for(int i = 0; i < j; i++){ //loop B studentList[i].printDetails(); } return 0; } </pre>
Observation	
Output	<p>Constructor Called. Constructor Called. Carly 15 Freddy 16 Sam 18 Zack 19 Cody 16 Destructor Called. Destructor Called. Destructor Called. Destructor Called. Destructor Called.</p> <p>==== Code Execution Successful ===</p>
Observation	

Table 2-2. Modified Driver Program with Student Lists

7. Supplementary Activity

ILO C: Solve programming problems using dynamic memory allocation, arrays and pointers

Jenna wants to buy the following fruits and vegetables for her daily consumption. However, she needs to distinguish between fruit and vegetable, as well as calculate the sum of prices that she has to pay in total.

Problem 1: Create a class for the fruit and the vegetable classes. Each class must have a constructor, deconstructor, copy constructor and copy assignment operator. They must also have all relevant attributes (such as name, price and quantity) and functions (such as calculate sum) as presented in the problem description above.

See how a CS professor is using our compiler for class assignment. Try Programiz PRO for Educators!

Programiz
C++ Online Compiler

kfc.com.ph | KFC PH App

main.cpp

```
1 #include <iostream>
2 #include <string>
3
4
5 class Fruit {
6 public:
7     std::string name;
8     double price;
9     int quantity;
10
11
12     Fruit(std::string n, double p, int q) {
13         name = n;
14         price = p;
15         quantity = q;
16     }
17
18
19     void display() {
20         std::cout << "Fruit: " << name << ", Price: PHP " << price << ", "
21         Quantity: " << quantity << std::endl;
22     }
23
24
25 class Vegetable {
```

Run

Output

Fruit: Apple, Price: PHP 10, Quantity: 7
Vegetable: Broccoli, Price: PHP 60, Quantity: 12

==== Code Execution Successful ===

Programiz PRO >

KFC CHOOSE & MATCH

STEP 1: CHOOSE YOUR FAVE 'N' STEP 2: MATCH WITH A SIDE

KFC PH APP

COMBO A - ₱80.00

COMBO B - ₱95.00

Coca-Cola

Programiz PRO >

KFC FLAVOR SHOTS DEAL

FLAVOR SHOTS & MASHED POTATO BUNDLE

ONLY ₱80

SAME AS MUCH AS POSS

Available 2pm to 5pm only.
Available for Dine-In,
Take Out and
Drive Thru orders.

Problem 2: Create an array `GroceryList` in the driver code that will contain all items in Jenna's Grocery List. You must then access each saved instance and display all details about the items.

Dashboard Online C++ Compile DATA STRUCTURE PROCEDURE AUGUS get.microsoft.com

programiz.com/cpp-programming/online-compiler/

See how a CS professor is using our compiler for class assignment. Try Programiz PRO for Educators!

Programiz
C++ Online Compiler

KFC CHOOSE & MATCH
kfc.com.ph | KFC PH App

Programiz PRO

main.cpp

```

4
5
6- struct GroceryItem {
7    std::string name;
8    double price;
9    int quantity;
10 };
11
12- int main() {
13
14     std::vector<GroceryItem> jennaGroceryList;
15
16
17
18     GroceryItem apple;
19     apple.name = "Apple";
20     apple.price = 10.0;
21     apple.quantity = 7;
22     jennaGroceryList.push_back(apple);
23
24
25     GroceryItem banana;
26     banana.name = "Banana";
27     banana.price = 10.0;
28     banana.quantity = 8;
29     jennaGroceryList.push_back(banana);

```

Run

Output

```

--- Here is Jenna's full grocery list: ---
Item: Apple
Price: PHP 10
Quantity: 7
-----
Item: Banana
Price: PHP 10
Quantity: 8
-----
Item: Broccoli
Price: PHP 60
Quantity: 12
-----
Item: Lettuce
Price: PHP 50
Quantity: 10
-----
==== Code Execution Successful ====

```

KFC FLAVOR SHOTS DEAL
FLAVOR SHOTS & MASHED POTATO BUNDLE
ONLY ₱80
Available 2pm to 5pm only.
Available for Dine-In, Take Out and Drive Thru orders.

10:11 AM 8/4/2025

Problem 3: Create a function TotalSum that will calculate the sum of all objects listed in Jenna's Grocery List.

Dashboard Online C++ Compile DATA STRUCTURE PROCEDURE AUGUS get.microsoft.com

programiz.com/cpp-programming/online-compiler/

See how a CS professor is using our compiler for class assignment. Try Programiz PRO for Educators!

Programiz
C++ Online Compiler

KFC CHOOSE & MATCH
kfc.com.ph | KFC PH App

Programiz PRO

main.cpp

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4
5
6- struct GroceryItem {
7    std::string name;
8    double price;
9    int quantity;
10 };
11
12- double TotalSum(const std::vector<GroceryItem>& list) {
13     double total = 0.0;
14
15
16     for (int i = 0; i < list.size(); i++) {
17         total += list[i].price * list[i].quantity;
18     }
19
20     return total;
21 }
22
23- int main() {
24     std::vector<GroceryItem> jennaGroceryList;
25
26

```

Run

Output

```

Jenna's total grocery cost is: PHP 1370
==== Code Execution Successful ====

```

KFC FLAVOR SHOTS DEAL
FLAVOR SHOTS & MASHED POTATO BUNDLE
ONLY ₱80
Available 2pm to 5pm only.
Available for Dine-In, Take Out and Drive Thru orders.

10:13 AM 8/4/2025

Problem 4: Delete the Lettuce from Jenna's GroceryList list and de-allocate the memory assigned.

The screenshot shows a browser window with several tabs open at the top: Dashboard, Online C++ Compile, DATA STRUCTURE, PROCEDURE AUGUS, get.microsoft.com, and others. The main content area is the Programiz C++ Online Compiler. On the left, there's a sidebar with icons for various file types. The code editor contains a C++ program named main.cpp:

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4
5
6 struct GroceryItem {
7     std::string name;
8     double price;
9     int quantity;
10 };
11
12
13 double calculateTotalSum(const std::vector<GroceryItem>& myList) {
14     double total = 0.0;
15
16     for (int i = 0; i < myList.size(); i++) {
17         total = total + (myList[i].price * myList[i].quantity);
18     }
19
20     return total;
21 }
22
23
24
25 int main() {
```

The output window shows the execution results:

```
Here is the starting grocery list:
Item: Apple, Quantity: 7, Price: 10
Item: Banana, Quantity: 8, Price: 10
Item: Broccoli, Quantity: 12, Price: 60
Item: Lettuce, Quantity: 10, Price: 50
The total cost for the groceries is: PHP 1370

Okay, now we're going to remove the Lettuce from the list.
The updated grocery list is:
Item: Apple, Quantity: 7, Price: 10
Item: Banana, Quantity: 8, Price: 10
Item: Broccoli, Quantity: 12, Price: 60
The new total cost is: PHP 870

==== Code Execution Successful ====
```

On the right side of the compiler interface, there are two KFC promotional banners: "KFC CHOOSE & MATCH" and "KFC FLAVOR SHOTS DEAL". The bottom right corner of the screen shows the system status bar with the time (10:16 AM) and date (8/4/2025).

8. Conclusion

Making a basic grocery list program allowed us to get a great deal of information about C++. First of all, we determined how many we needed and how to make a struct—which is similar to a custom container—to store an item's name and price. Storing all those items in a vector makes it incredibly convenient as it may grow or shrink as we add or remove items.

We also designed a feature for total expense computation. That was rather neat since it demonstrated how you might create a distinct piece of code to execute a certain task. At last, we learnt how to search for and remove an item, such as lettuce, from our list. This was a great lesson on how to handle our data.

This exercise showed you well how you may approach a daily issue using basic programming ideas. For anyone studying C++, this is a good beginning. Though it wasn't too challenging, it certainly demonstrated how events relate.

9. Assessment Rubric

