Introduction to programming

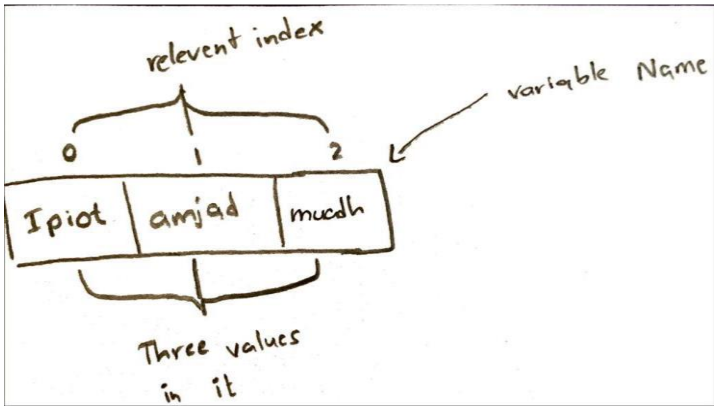
Assignment 9.1C

Student id: - BSCP|CS|63|136

Name: - Iflal Iqbal

Arrays

- Arrays are the basic data structure that allows storing multiple elements of the same data type all together. The disadvantage is that arrays have a fixed size, that fixed size of the array must be declared in the initial declaration. The elements in an array are stored sequentially in memory and that can be accessed by the index, starting from 0. Arrays are much better to store a list of known values. Fast lookup times based on indexes is a major advantage of array. It is easy to loop through elements using a for or while loop. Below is a simple sketch and a code block to understand it.



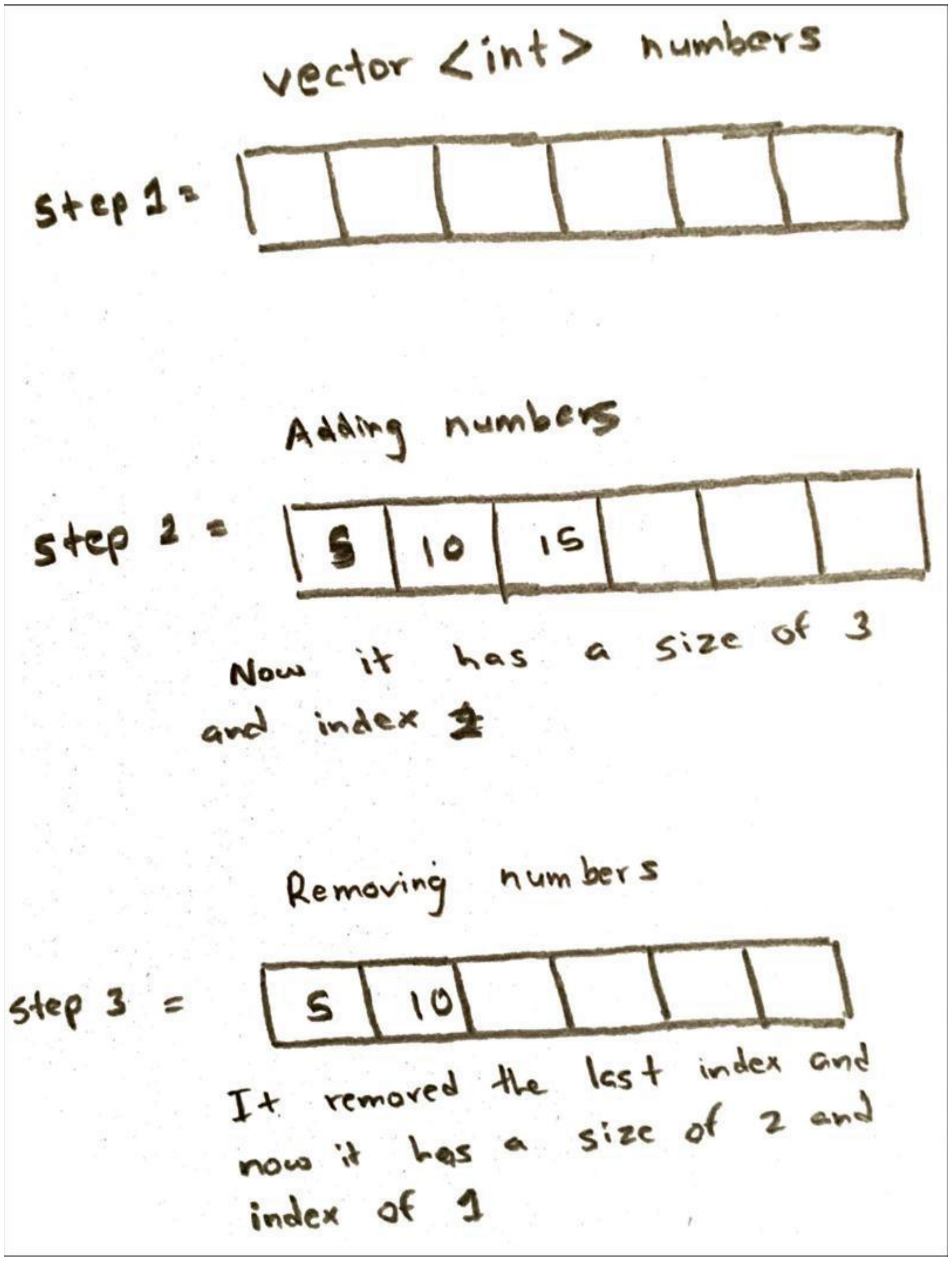
```
String name[3] ={“ipiot”, “amjad”, “muadh”}:

Write_line(name[1]);
```

Output
amjad

In this example I have declared a variable name which holds an array value of three as (ipiot, Amjad and muadh). As we discussed above index is the best friend of array, here in this case we have three value which means we have two index (index is being counted from 0). In the second line in our code we are asking to print the index of in the array name so that it gives the output as “amjad” where ipiot is in index 0 and muadh is in index 2.

Dynamic arrays (vectors in C++)

- A dynamic array is implemented using vectors in C++. Here the disadvantage of the fixed size array is been eliminated by implementing dynamic arrays. In dynamic arrays the elements can be added or removed without effecting the other elements inside the array that's what we implement vectors. The values will be added to the vector class using push_back() and values will be removed using pop_back(). In the case of dynamic arrays, we do not need to declare the size of data at the initial stage. Vectors are very useful to store data when we do not know the exact size of it. Below is a code block of mine,


vector <int> numbers
step 1 =
Adding numbers
step 2 =
5
10
15
Now it has a size of 3 and index 2
Removing numbers
step 3 =
5
10
It removed the last index and now it has a size of 2 and index of 1

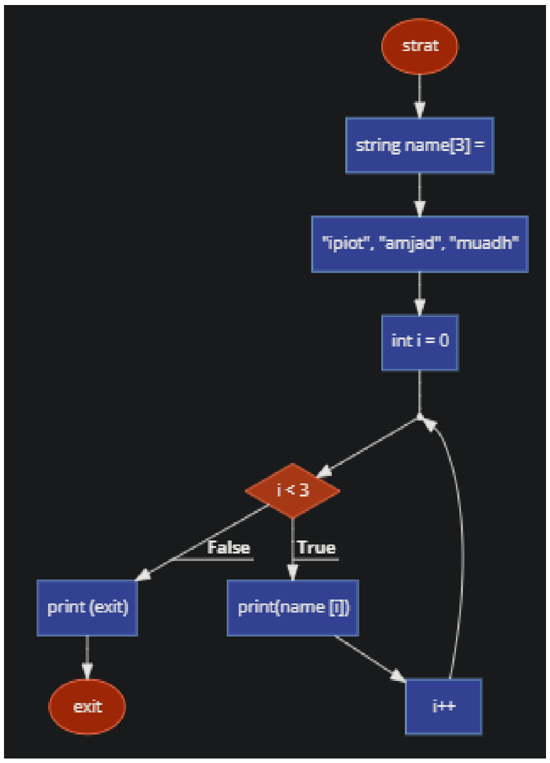
This is a simple visualization as what happen is the below code block, remember whatever being added or removed to the array it will not affect it. It will just resize and compress.

Example: -

```
int main() {
    // Create a vector of integers
vector<int> numbers;

    // Add few numbers to it
numbers.push_back(5);
numbers.push_back(10);
numbers.push_back(15);

    // Print the vector values
write_line("vector values: ");
    for (int i = 0; i < numbers.size(); ++i) {
write_line(numbers[i]);
 if (i < numbers.size() - 1) {
            write_line("");
        }
}
    write_line();

    //checking the condition to remove the last value
if (!numbers.empty()) {
    numbers.pop_back();
    }
    //printing the updated values
write_line("updated vector values: ");
for (int i = 0; i < numbers.size(); i++) {
write_line(numbers[i]);         if (i <
numbers.size() - 1) {
            write_line("");
        }
}
    return 0;
}
//simply what's done here is I have created a vector to store integer values and I am putting three values
into it (5, 10, 15) and then I call to print all the elemets inside the vector to be printed. And then I create a
if condition as if the "!numbers.empty" this means if the numbers not empty the code block inside it will
execute as it will remove the last value in the vector and I am asking to print the updated value again.
```

For loops

- For loops are a type of control structure that allows a block of code to be executed again and again reputedly for a few numbers of time until it meets the condition. Each of the loop includes an initialized value, a condition to check and an increment. Using for loops we could implement iterations and eliminate repetitions in the code again and again without collision.



```
int main() {
 string name[3] = {"ipiot", "amjad", "muadh"};
   for (int i = 0; i < 3; i++)
    {
          write_line(name [i]);
    }
}
```

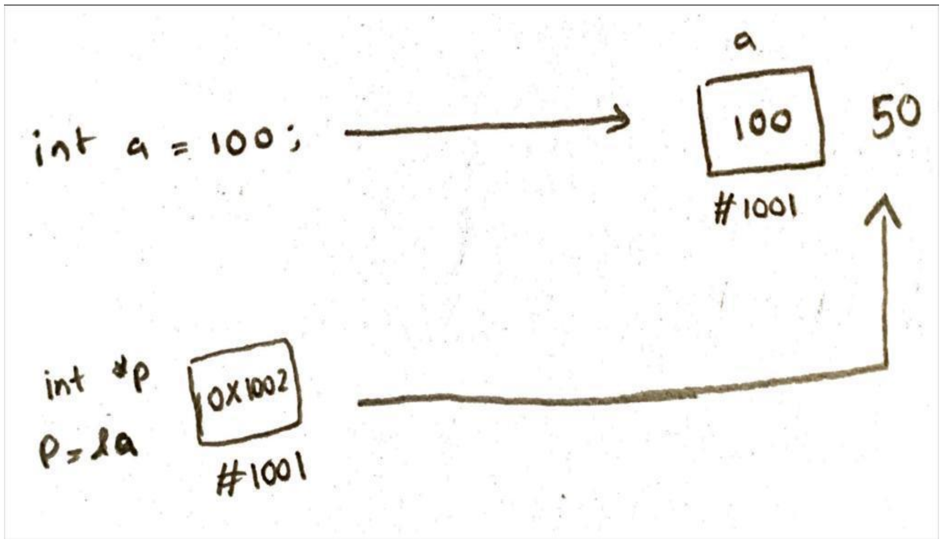
Here is a simple code of mine where I have used for loop instead of printing each value separately. First, I have a string name variable which is an array holding three value (ipiot, Amjad and muadh), and having the index as two. Then I have created a for loop declared as i=0 (considering i as index) and asking if the i is smaller than 3 the do then below statement, so in the first case it is lower than 3 so its true and it will print the name ipiot there and again back to the loop condition as this it repeat itself till the i is equal to 3 if it is equal it get out of the loop. In this loop case it gives an output as,

Output

```
Ipiot

Amjad

Muadh
```

Pointers

- Pointers are variables that store memory addresses instead of storing values. They allow control of memory contents by using memory addresses. To get the actual value kept at the address pointers use the '*' operator to get there. This allow memory access at a low level. Pointers also allow dynamic memory allocation. **('write_line' is simple saying to print).** Below is a image to explain the simple code block below.



```
int main() {
int a = 100;
int *p;
 p = &a;
write_line(a);
a = 50;
write_line(*p);
return 0;
}
```

Here is simpler explanation of the above code, here I had declared a integer a and a value of 100 is assigned to it. And then I had declared a integer pointer variable p without any initialization for it. In the next the pointer p is assigned the memory address of the variable a. now the pointer p will direct to the memory location where the value of a is stored. And then I am asking to print a value and there will be a output of 100 for that. And then again, I had assigned the a with a value of 50 this will change the value of a. when I ask to print it again, I will get the output of 50.

Output

100

50