Principles of Programming Language

Assignment 7

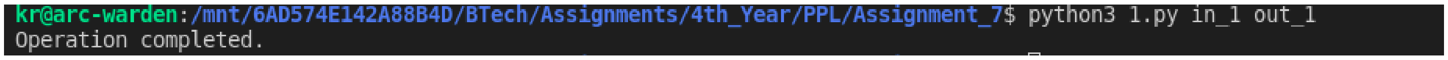
Student Details

Name: Krunal Rank
Adm No: U18CO081

1

```
import sys

class ERRORS:
   INVALID_ARGS = """Invalid Arguments.\nPlease enter 2 Arguments which are Input
File Path and Output File Path respectively."""

def perform_operation(input_file_path, output_file_path):
   """
   Reads input file, replaces consecutive spaces to a single space and writes to
output file.
   """
   with open(input_file_path, 'r+') as input_file:
       with open(output_file_path, 'w') as output_file:
           output_file.write(" ".join(input_file.read().split()))

if __name__ == "__main__":
   if len(sys.argv) != 3:
       raise Exception(ERRORS.INVALID_ARGS)

   perform_operation(sys.argv[1], sys.argv[2])

   print("Operation completed.")
```

```
Input:
This   is some  text   and has    some    input    consecutive spaces.
Output:
This is some text and has some input consecutive spaces.
```



```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ python3 1.py in_1 out_1
Operation completed.
```

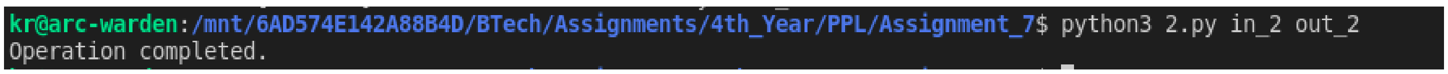
2

```
import sys
```

```python
class ERRORS:
    INVALID_ARGS = """Invalid Arguments.\nPlease enter 2 Arguments which are Input 
File Path and Output File Path respectively."""

def perform_operation(input_file_path, output_file_path):
    """
    Reads input file, and copies content to output file
    """
    with open(input_file_path, 'rb') as input_file:
        with open(output_file_path, 'wb') as output_file:
            output_file.write(input_file.read())

if __name__ == "__main__":
    if len(sys.argv) != 3:
        raise Exception(ERRORS.INVALID_ARGS)

    perform_operation(sys.argv[1], sys.argv[2])

    print("Operation completed.")
```

Input File:
Input Text with some values
1
2
3
4

Output File:
Input Text with some values
1
2
3
4

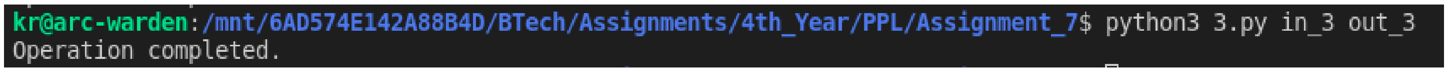


```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ python3 2.py in_2 out_2
Operation completed.
```

3

```python
import sys

class ERRORS:
   INVALID_ARGS = """Invalid Arguments.\nPlease enter 2 Arguments which are Input File Path and Output File Path respectively."""

def perform_operation(input_file_path, output_file_path):
   """
   Reads input file, and copies content to output file switching the case of each character.
   """
   with open(input_file_path, 'r') as input_file:
       with open(output_file_path, 'w') as output_file:
           output_file.write(input_file.read().swapcase())

if __name__ == "__main__":
   if len(sys.argv) != 3:
       raise Exception(ERRORS.INVALID_ARGS)

   perform_operation(sys.argv[1], sys.argv[2])

   print("Operation completed.")
```

```
Input File:
This is some RanDom text With manY cases.

Output File:
tHIS IS SOME rANdOM TEXT wITH MANy CASES.
```

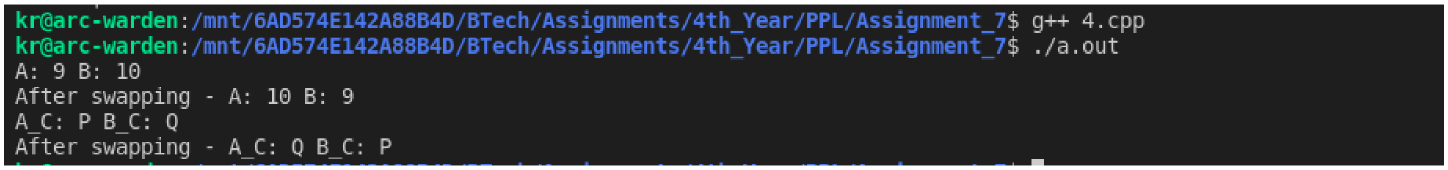


```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ python3 3.py in_3 out_3
Operation completed.
```

4.

```
#include <bits/stdc++.h>
using namespace std;

template<class T>

void custom_swap(T& a,T& b){
    T p = a;
    a = b;
    b = p;
}

int main(){
    int a = 9, b = 10;
```

```cpp
    cout<<"A: "<<a<<" B: "<<b<<endl;
    custom_swap(a,b);
    cout<<"After swapping - A: "<<a<<" B: "<<b<<endl;
    char a_c = 'P',b_c='Q';
    cout<<"A_C: "<<a_c<<" B_C: "<<b_c<<endl;
    custom_swap(a_c,b_c);
    cout<<"After swapping - A_C: "<<a_c<<" B_C: "<<b_c<<endl;


}
```

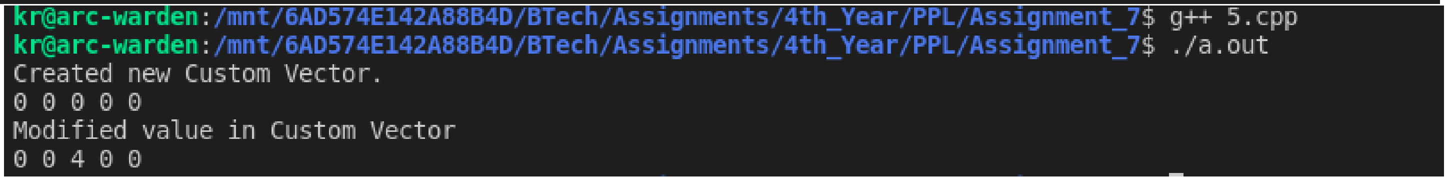


```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ g++ 4.cpp
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ ./a.out
A: 9 B: 10
After swapping - A: 10 B: 9
A_C: P B_C: Q
After swapping - A_C: Q B_C: P
```

5

```cpp
#include <bits/stdc++.h>

using namespace std;

template<typename T>

class CustomVector{
    public:
        T* ptr;
        int s;
        CustomVector(int size){
            ptr = new T[s];
            s = size;
        }
        CustomVector(int size, T d){
            ptr = new T[s];
            s = size;
            for(int i = 0;i<s;i++) ptr[i] = d;
        }
        void modify(int p,T v){
            if(!(p>=0 && p<s)) return;
            ptr[p] = v;
        }
        int size(){
            return s;
        }
};
```

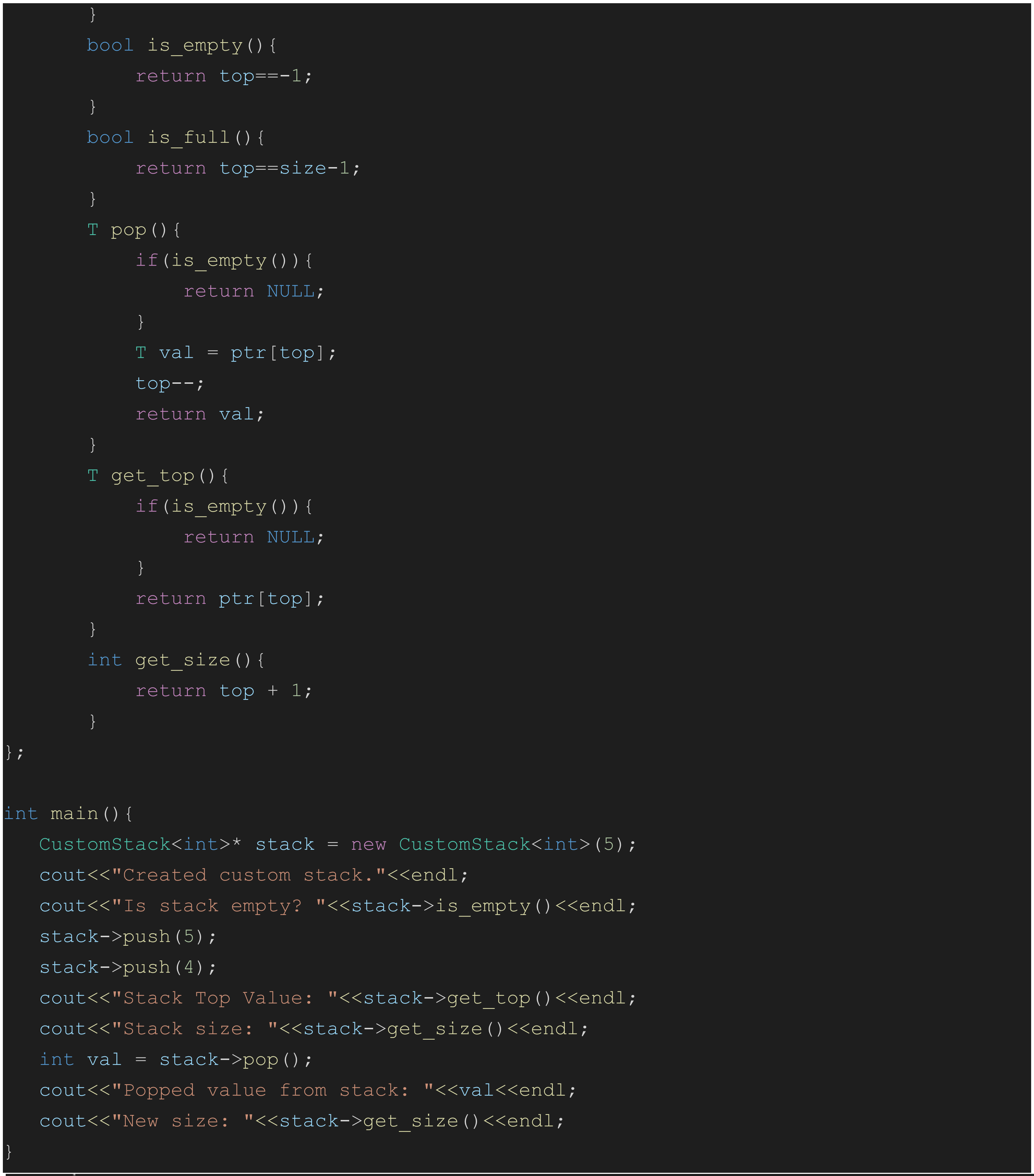
```
int main(){
   CustomVector<int>* new_vec = new CustomVector<int>(5,0);

   cout<<"Created new Custom Vector."<<endl;
   for(int i = 0;i<5;i++) cout<<new_vec->ptr[i]<<" ";
   cout<<endl;

   new_vec->modify(2,4);
   cout<<"Modified value in Custom Vector"<<endl;
   for(int i = 0;i<5;i++) cout<<new_vec->ptr[i]<<" ";
   cout<<endl;




}
```



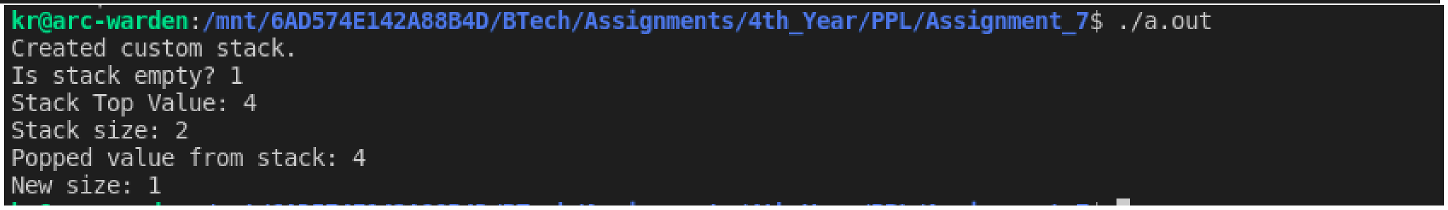
```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ g++ 5.cpp
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ ./a.out
Created new Custom Vector.
0 0 0 0 0
Modified value in Custom Vector
0 0 4 0 0
```

6

```
#include <bits/stdc++.h>

using namespace std;

template<typename T>

class CustomStack{
   private:
       T* ptr;
       int size;
       int top = -1;
   public:
       CustomStack(int s){
           ptr = new T[s];
           size = s;
       }
       void push(T v){
           if(is_full()){
               return;
           }
           top++;
           ptr[top] = v;
```



```cpp
        }
        bool is_empty(){
            return top==-1;
        }
        bool is_full(){
            return top==size-1;
        }
        T pop(){
            if(is_empty()){
                return NULL;
            }
            T val = ptr[top];
            top--;
            return val;
        }
        T get_top(){
            if(is_empty()){
                return NULL;
            }
            return ptr[top];
        }
        int get_size(){
            return top + 1;
        }
};

int main(){
   CustomStack<int>* stack = new CustomStack<int>(5);
   cout<<"Created custom stack."<<endl;
   cout<<"Is stack empty? "<<stack->is_empty()<<endl;
   stack->push(5);
   stack->push(4);
   cout<<"Stack Top Value: "<<stack->get_top()<<endl;
   cout<<"Stack size: "<<stack->get_size()<<endl;
   int val = stack->pop();
   cout<<"Popped value from stack: "<<val<<endl;
   cout<<"New size: "<<stack->get_size()<<endl;
}
```



```
kr@arc-warden:/mnt/6AD574E142A88B4D/BTech/Assignments/4th_Year/PPL/Assignment_7$ ./a.out
Created custom stack.
Is stack empty? 1
Stack Top Value: 4
Stack size: 2
Popped value from stack: 4
New size: 1
```