# Name: Tayyab Zain
# RegNo:4293 FBAS/BSCS4/F20

## Assignment 2:

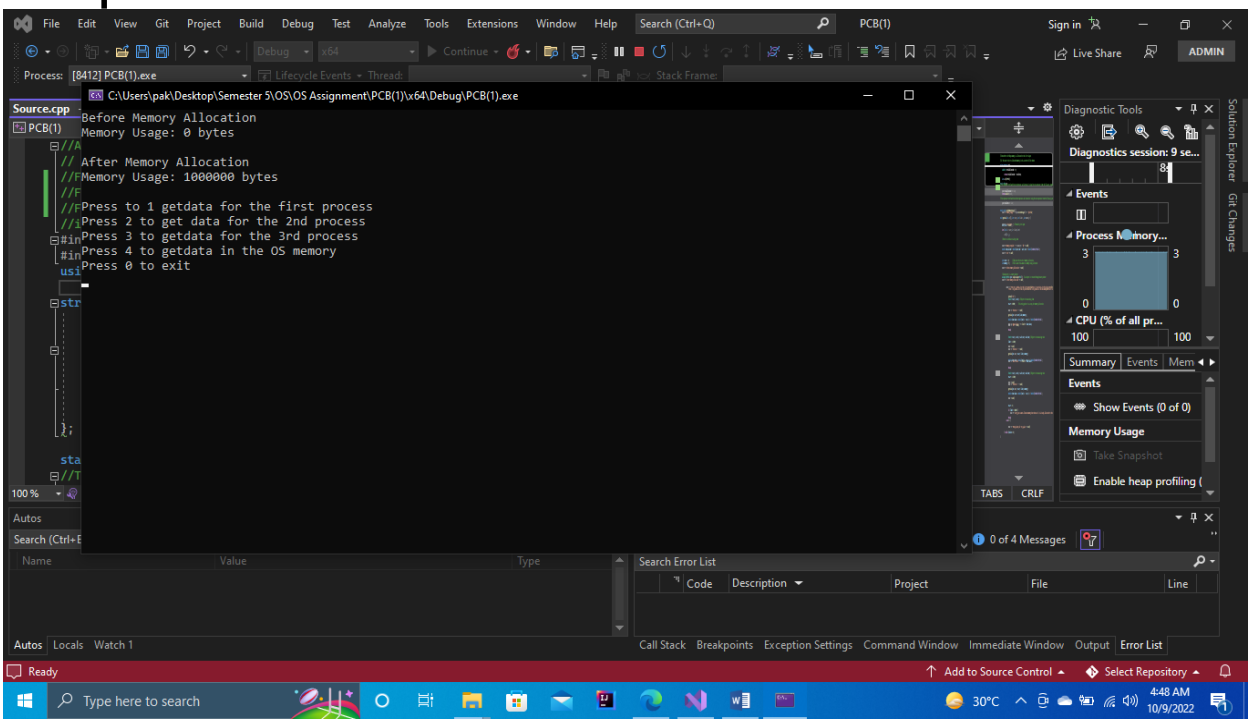
```
//A PCB Simulation
// First 200kbps are allocated for the OS
//From 200 to 370 kbps memory is allocated to the first input
//From 370 to 500 kbps memory is allocated to the second input
//From 500 to 1000 kbps memory is allocated to the third input
//if the user tries to allocated memory in OS an error will be shown
#include<iostream>
#include<memory>
using namespace std;

struct PCB {
	uint32_t totalAllocated = 0;
	uint32_t totalFree = 0;
	uint32_t currentUsage() {
		return totalAllocated - totalFree;
	};

	int arr[249998];
};

static PCB pro;
//This Operator Overloads the new Operator and instead of using the new operator
from the libarary program
// will now use the operator from here
void* operator new(size_t size) {

	pro.totalAllocated += size;

	return malloc(size);
}

//This Operator Overloads the delete Operator and instead of using the new operator
from the libarary program
// will now use the operator from here
void operator delete(void* memory, size_t size) {
	pro.totalFree += size;
	free(memory);
}

//Prints memory usage
static void printMemoryUsage() {
	cout << "Memory Usage: " << pro.currentUsage() << " bytes\n";
}

//Gets the data for the process
void getdata(int arr[], int start, int limit , int memory) {
	clock_t wait_stat, wait_end;
```

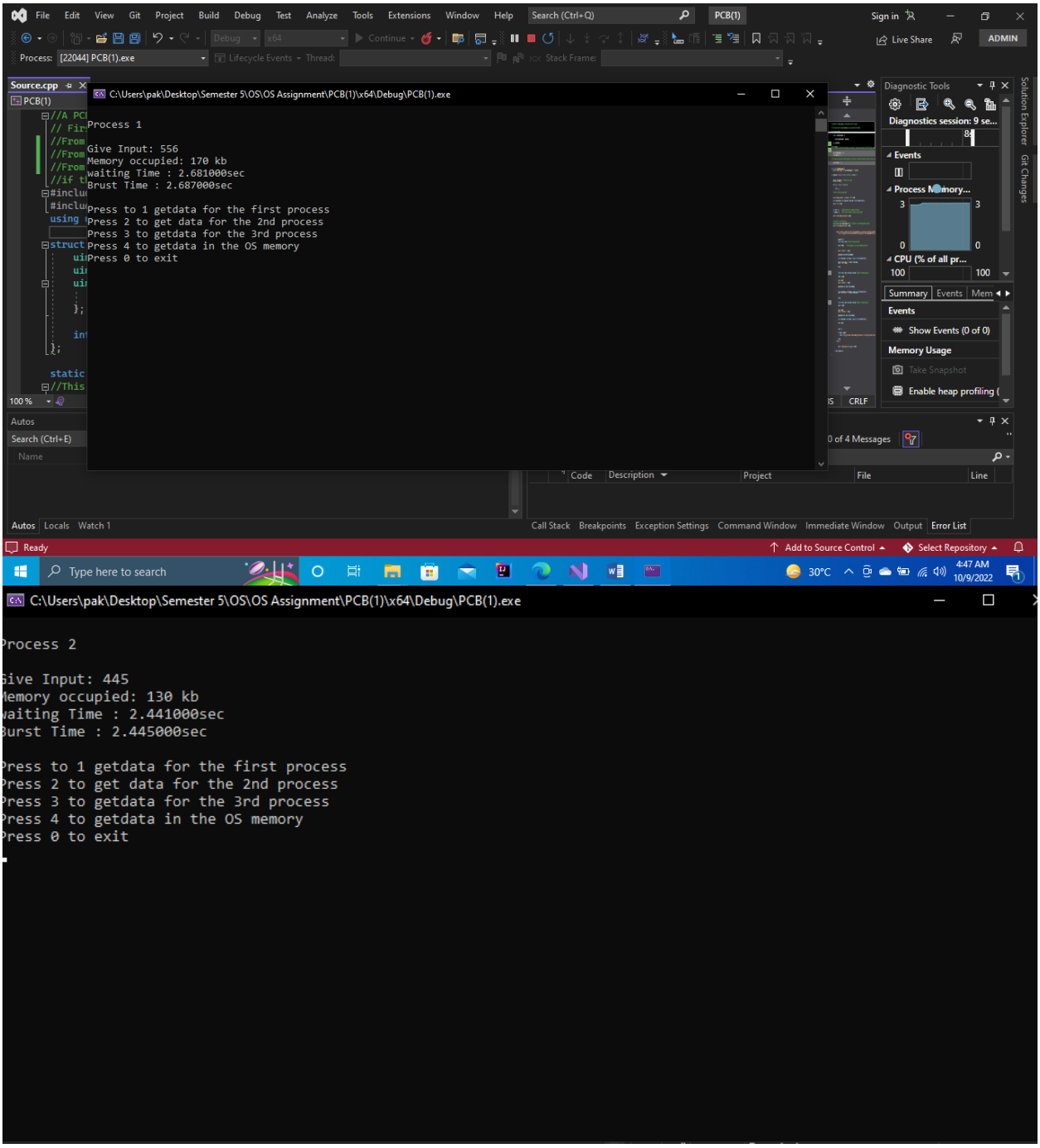
```
	cout << endl;
	int num = 0;
	cout << "Give Input: "; //Formality for input
	wait_stat = clock();
	cin >> num;
	wait_end = clock();
	for (int i = start; i < limit; i++)
	{
		int j = rand();
		arr[i] = j;

	}
	//Used to calculate waiting time
	int Count = limit - start;
	memory = sizeof(*arr) * Count;
	memory = memory / 1000;
	cout <<"Memory occupied: " << memory<< " kb" << endl;

	//Used to calculaed waiting i/o time
	double time_waited = double(wait_end - wait_stat) / double(CLOCKS_PER_SEC);
	cout << "waiting Time : " << fixed << time_waited;
	cout << "sec" << endl;
}

int main() {
	int choice = 0;			//will be usind in switch statement
	int start = 0;			//indicates the start of memory allocation
	int limit = 0;			//indicates the end of memory allocation
	int memory = 0;			//tells about the amount of memory being allocated
	int p_ID = 0;			//tels about the process id

	cout << "Before Memory Allocation" << endl;
	printMemoryUsage();
	cout << endl;

	//unique_ptr is a smart pointer
	//make_unique is helper function
	unique_ptr<PCB> pro = make_unique<PCB>();  //an object is created through
smart pointer

	cout << "After Memory Allocation" << endl ;
	printMemoryUsage();
	cout << endl;
		do
		{
			cout << "Press to 1 getdata for the first process\nPress 2 to
get data for the 2nd process\n"<<
				"Press 3 to getdata for the 3rd process\nPress 4 to
getdata in the OS memory\nPress 0 to exit"<<endl;
			cin >> choice;
			switch (choice)
			{
			case 1: {
				system("cls");
				//clock_t is a class
				clock_t stat_1, end_1; //Objects for measuring time

				cout << endl;
```

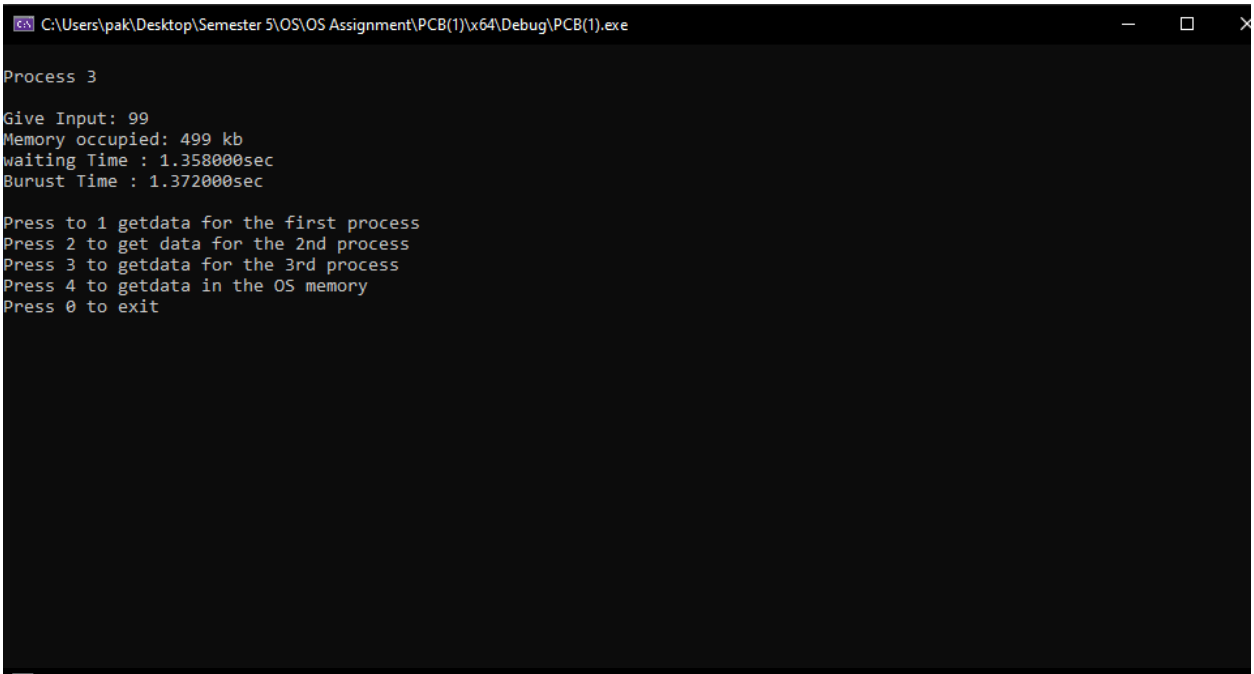
```
                        start = 50000;            //Starting point of an array for
memory allocation
                        limit = 92500;            //End point of an array for
memory allocation
                        p_ID = 1;

                        cout << "Process 1 " << endl;

                        stat_1 = clock();
                        getdata(pro->arr,start,limit,memory);
                        end_1 = clock();
                        //Used to calculate Burust time
                        double time_taken = double(end_1 - stat_1) /
double(CLOCKS_PER_SEC);


                        cout << "Brust Time : " << fixed << time_taken;
                        cout << "sec" << endl;
                        cout << endl;

                        break;

                }
                case 2: {
                        system("cls");
                        clock_t stat_1, end_1, wait_stat, wait_end; //Objects for
measuring time

                        start = 92500;
                        limit = 125000;
                        p_ID = 2;

                        cout << endl;

                        cout << "Process 2 " << endl;

                        stat_1 = clock();
                        getdata(pro->arr, start, limit, memory);
                        end_1 = clock();


                        double time_taken = double(end_1 - stat_1) /
double(CLOCKS_PER_SEC);
                        cout << "Burst Time : " << fixed << time_taken ;
                        cout << "sec" << endl;
                        cout << endl;

                        break;
                }
                case 3: {
                        system("cls");
                        clock_t stat_1, end_1, wait_stat, wait_end; //Objects for
measuring time

                        start = 125000;
                        limit = 249998;
                        p_ID = 3;
```

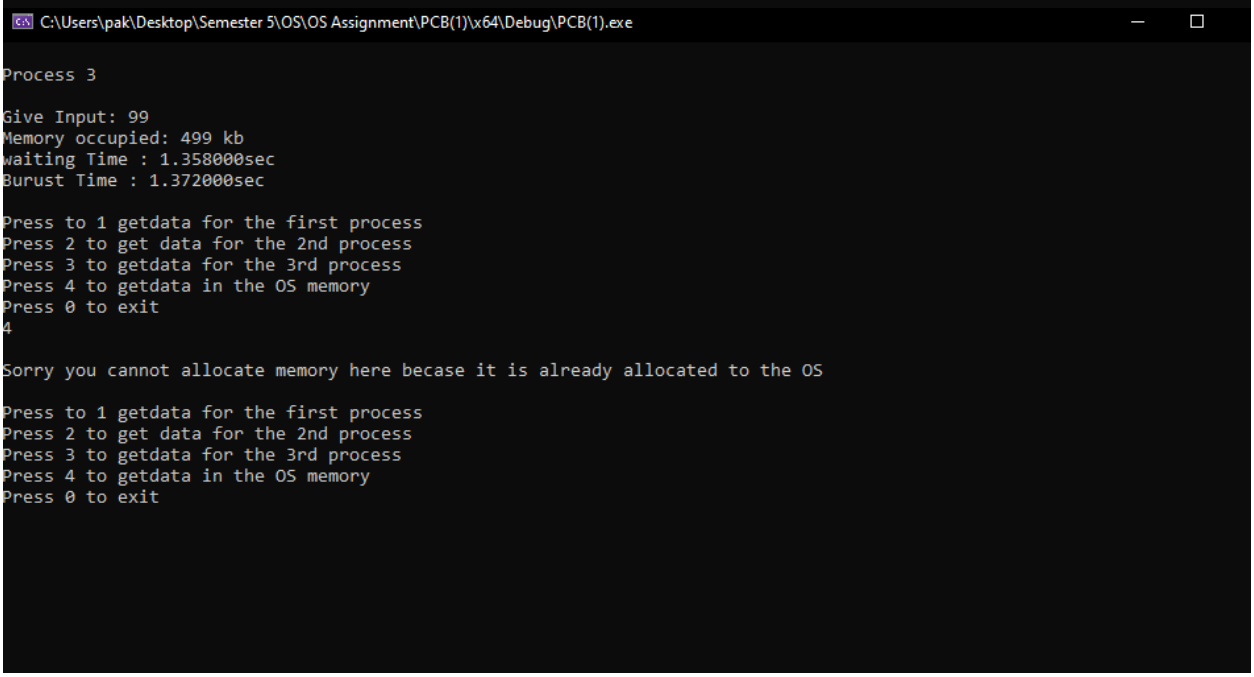
```
                        cout << endl;
                        cout << "Process 3 " << endl;

                        stat_1 = clock();
                        getdata(pro->arr, start, limit, memory);
                        end_1 = clock();

                        double time_taken = double(end_1 - stat_1) /
double(CLOCKS_PER_SEC);
                        cout << "Burust Time : " << fixed << time_taken;
                        cout << "sec" << endl;
                        cout << endl;

                        break;
                }
                case 4: {
                        start = 0;
                        limit = 0;
                        p_ID = 4;
                        if (limit < 50000) {
                                cout << endl;
                                cout << "Sorry you cannot allocate memory here
becase it is already allocated to the OS" << endl;
                                cout << endl;
                        }
                        break;
                }
                case 0: {
                        exit(0);
                }
                default:
                        cout << "Wrong input plz try again" << endl;
                        break;
                }

        } while (choice!=0);


}
```

# Outputs:


C:\Users\pak\Desktop\Semester 5\OS\OS Assignment\PCB(1)\x64\Debug\PCB(1).exe
Before Memory Allocation
Memory Usage: 0 bytes
// After Memory Allocation
//Memory Usage: 1000000 bytes
//Press to 1 getdata for the first process
//Press 2 to get data for the 2nd process
//Press 3 to getdata for the 3rd process
//Press 4 to getdata in the OS memory
//Press 0 to exit


C:\Users\pak\Desktop\Semester 5\OS\OS Assignment\PCB(1)\x64\Debug\PCB(1).exe
Process 1
Give Input: 556
Memory occupied: 170 kb
Waiting Time : 2.681000sec
Brust Time : 2.687000sec
Press to 1 getdata for the first process
Press 2 to get data for the 2nd process
Press 3 to getdata for the 3rd process
Press 4 to getdata in the OS memory
Press 0 to exit
C:\Users\pak\Desktop\Semester 5\OS\OS Assignment\PCB(1)\x64\Debug\PCB(1).exe
Process 2
Give Input: 445
Memory occupied: 130 kb
Waiting Time : 2.441000sec
Burst Time : 2.445000sec
Press to 1 getdata for the first process
Press 2 to get data for the 2nd process
Press 3 to getdata for the 3rd process
Press 4 to getdata in the OS memory
Press 0 to exit


C:\Users\pak\Desktop\Semester 5\OS\OS Assignment\PCB(1)\x64\Debug\PCB(1).exe
Process 3
Give Input: 99
Memory occupied: 499 kb
waiting Time : 1.358000sec
Burust Time : 1.372000sec
Press to 1 getdata for the first process
Press 2 to get data for the 2nd process
Press 3 to getdata for the 3rd process
Press 4 to getdata in the OS memory
Press 0 to exit


C:\Users\pak\Desktop\Semester 5\OS\OS Assignment\PCB(1)\x64\Debug\PCB(1).exe
Process 3
Give Input: 99
Memory occupied: 499 kb
waiting Time : 1.358000sec
Burust Time : 1.372000sec
Press to 1 getdata for the first process
Press 2 to get data for the 2nd process
Press 3 to getdata for the 3rd process
Press 4 to getdata in the OS memory
Press 0 to exit
4
Sorry you cannot allocate memory here becase it is already allocated to the OS
Press to 1 getdata for the first process
Press 2 to get data for the 2nd process
Press 3 to getdata for the 3rd process
Press 4 to getdata in the OS memory
Press 0 to exit