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:



