**Name: Debjyoti Sarkar Sumonta**

**Student ID: 22341019**

**CSE321 Section 5**

***C Program***

**Task 1**

#include <stdio.h>

struct Item {

int quantity;

float unit\_price;

};

int main() {

struct Item paratha, vegetable, water;

int num\_people;

float total\_bill, individual\_payment;

printf("Quantity of Paratha: ");

scanf("%d", &paratha.quantity);

printf("Unit price of Paratha: ");

scanf("%f", &paratha.unit\_price);

printf("Quantity of Vegetables: ");

scanf("%d", &vegetable.quantity);

printf("Unit price of Vegetables: ");

scanf("%f", &vegetable.unit\_price);

printf("Quantity of Mineral Water: ");

scanf("%d", &water.quantity);

printf("Unit price of Mineral Water: ");

scanf("%f", &water.unit\_price);

total\_bill = (paratha.quantity \* paratha.unit\_price) + (vegetable.quantity \* vegetable.unit\_price) + (water.quantity \* water.unit\_price);

printf("Enter the number of people: ");

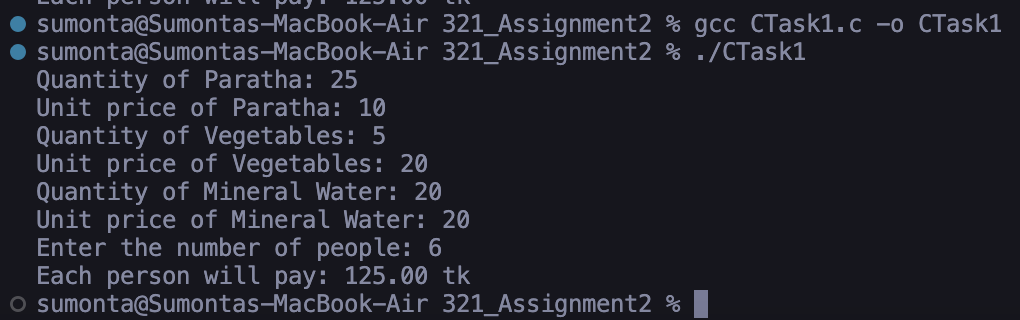
scanf("%d", &num\_people);

individual\_payment = total\_bill / num\_people;

printf("Each person will pay: %.2f tk\n", individual\_payment);

return 0;

}



**Task 2**

#include <stdio.h>

int isPerfect(int num) {

int sum = 0;

for (int i = 1; i < num; i++) {

if (num % i == 0) {

sum += i;

}

}

return sum == num;

}

int main() {

int start, end;

printf("Enter two intervals: ");

scanf("%d %d", &start, &end);

for (int i = start; i <= end; i++) {

if (isPerfect(i)) {

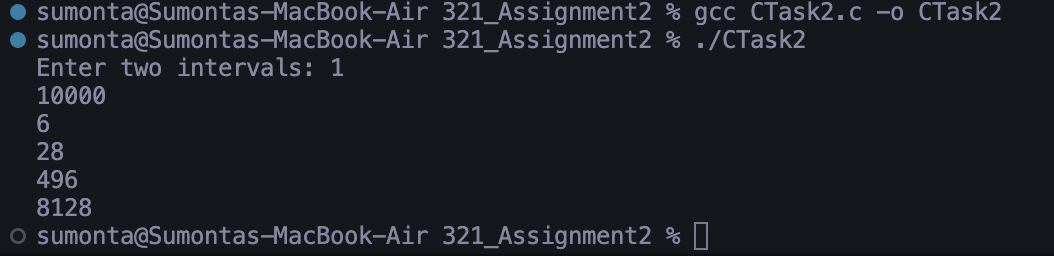
printf("%d\n", i);

}

}

return 0;

}



***System Calls***

**Task 1**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main(int argc, char \*argv[]) {

FILE \*file;

char input[100];

if (argc != 2) {

printf("Usage: %s <filename>\n", argv[0]);

return 1;

}

file = fopen(argv[1], "a");

if (file == NULL) {

printf("Error opening file!\n");

return 1;

}

while (1) {

printf("Enter a string (-1 to stop): ");

fgets(input, 100, stdin);

input[strlen(input) - 1] = '\0';

if (strcmp(input, "-1") == 0) {

break;

}

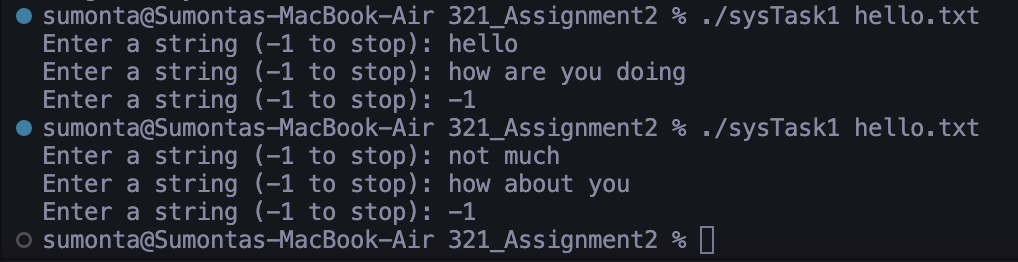
fprintf(file, "%s\n", input);

}

fclose(file);

return 0;

}



**Task 2**

#include <stdio.h>

#include <unistd.h>

#include <sys/wait.h>

int main() {

pid\_t pid1, pid2;

pid1 = fork();

if (pid1 == 0) {

pid2 = fork();

if (pid2 == 0) {

printf("I am grandchild\n");

} else {

wait(NULL);

printf("I am child\n");

}

} else {

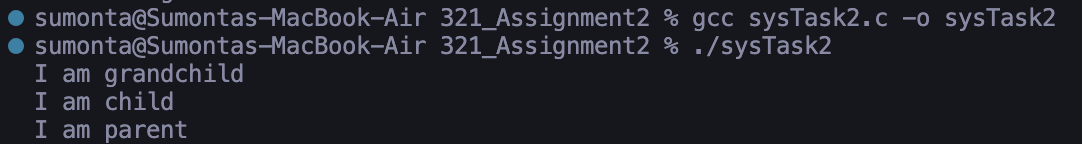
wait(NULL);

printf("I am parent\n");

}

return 0;

}



***Threads***

**Task 1**

#include <stdio.h>

#include <pthread.h>

void\* runThread(void\* arg) {

int id = \*(int\*)arg;

printf("Thread-%d running\n", id);

printf("Thread-%d closed\n", id);

pthread\_exit(NULL);

}

int main() {

pthread\_t threads[5];

int ids[5];

for (int i = 0; i < 5; i++) {

ids[i] = i + 1;

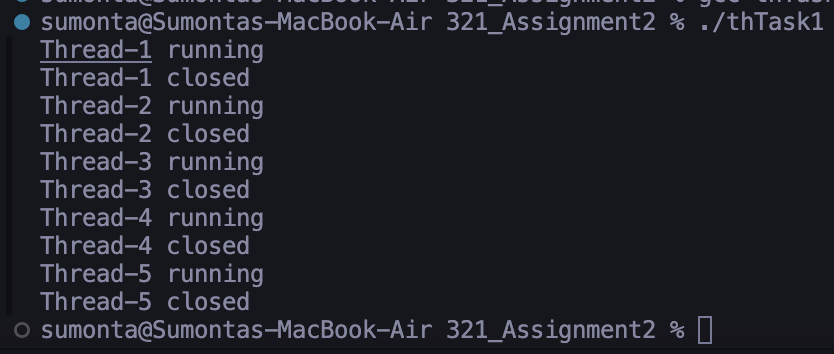
pthread\_create(&threads[i], NULL, runThread, &ids[i]);

pthread\_join(threads[i], NULL);

}

return 0;

}



**Task 2**

#include <stdio.h>

#include <pthread.h>

void\* printNumbers(void\* arg) {

int id = \*(int\*)arg;

for (int i = id \* 5 + 1; i <= id \* 5 + 5; i++) {

printf("Thread %d prints %d\n", id, i);

}

pthread\_exit(NULL);

}

int main() {

pthread\_t threads[5];

int ids[5];

for (int i = 0; i < 5; i++) {

ids[i] = i;

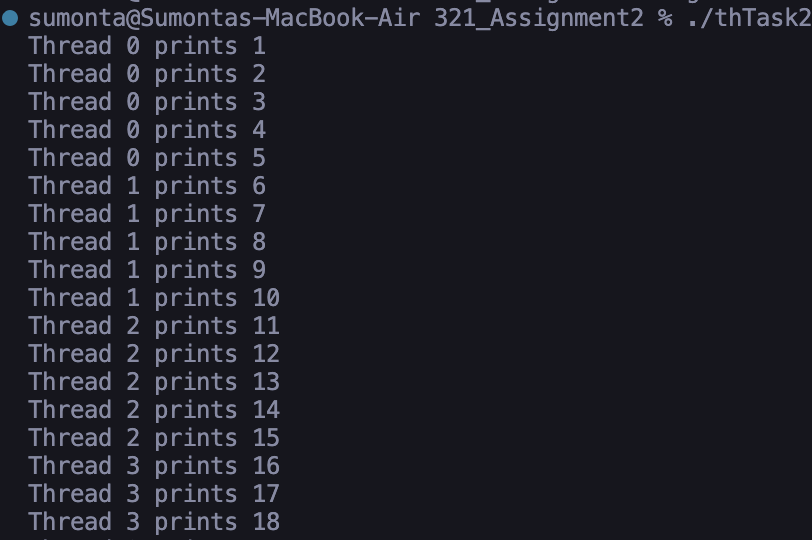
pthread\_create(&threads[i], NULL, printNumbers, &ids[i]);

pthread\_join(threads[i], NULL);

}

return 0;

}



**Task 3**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <string.h>

int ascii\_sum[3];

void\* calculate\_ascii\_sum(void\* arg) {

char\* name = (char\*)arg;

int sum = 0;

for (int i = 0; i < strlen(name); i++) {

sum += name[i];

}

int\* result = (int\*)malloc(sizeof(int));

\*result = sum;

pthread\_exit(result);

}

void\* compare\_sums(void\* arg) {

if (ascii\_sum[0] == ascii\_sum[1] && ascii\_sum[1] == ascii\_sum[2]) {

printf("Youreka\n");

} else if (ascii\_sum[0] == ascii\_sum[1] || ascii\_sum[1] == ascii\_sum[2] || ascii\_sum[0] == ascii\_sum[2]) {

printf("Miracle\n");

} else {

printf("Hasta la vista\n");

}

pthread\_exit(NULL);

}

int main() {

pthread\_t threads[4];

char\* usernames[3] = {"Jamal", "Hakim", "Noor"};

for (int i = 0; i < 3; i++) {

pthread\_create(&threads[i], NULL, calculate\_ascii\_sum, usernames[i]);

}

for (int i = 0; i < 3; i++) {

int\* result;

pthread\_join(threads[i], (void\*\*)&result);

ascii\_sum[i] = \*result;

free(result);

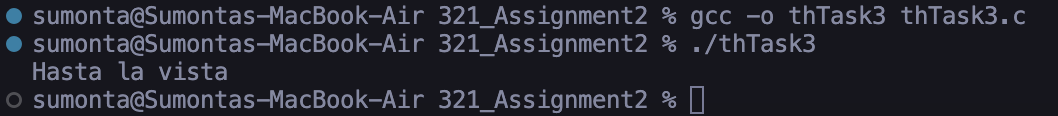
}

pthread\_create(&threads[3], NULL, compare\_sums, NULL);

pthread\_join(threads[3], NULL);

return 0;

}

****