Integrantes do grupo:

Sâmara Cristina Santos MAT0 11611EBI011

Carolina Beatriz Pereira da Silva MAT1 11711EBI022

Alana Miyai Bonitátibus MAT2 11711EBI018

---

QUESTÃO 1

KANO0=2

KANO1=3

KANO2=3

KCUR0=2

KCUR1=2

KCUR2=2

KNUM0=3

KNUM1=5

KNUM2=1

---

QUESTÃO 2

Código original

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#define N 10

#define KANO0 2 // trocar por valor devido

#define KANO1 3 // trocar por valor devido

#define KANO2 3 // trocar por valor devido

#define KCUR0 2 // trocar por valor devido

#define KCUR1 2 // trocar por valor devido

#define KCUR2 2 // trocar por valor devido

#define KNUM0 3 // trocar por valor devido

#define KNUM1 5 // trocar por valor devido

#define KNUM2 1 // trocar por valor devido

float media\_de\_aleatorios(int ID)

{

int \* p = (int \*) malloc(N\*sizeof(int));

int i;

float media = 0;

for(i = 0; i < N; i++)

{

p[i] = rand()%9 + 1;

media += p[i];

}

return media;

}

int main()

{

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

srand(ID0\*100+ID1\*10+ID2);

printf("1o: %f\n", media\_de\_aleatorios(ID0));

printf("2o: %f\n", media\_de\_aleatorios(ID1));

printf("3o: %f\n", media\_de\_aleatorios(ID2));

return EXIT\_SUCCESS;

}

1o: 53.000000

2o: 46.000000

3o: 61.000000

Código com alterações

#include <conio.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define N 10

#define KANO0 2

#define KANO1 3

#define KANO2 3

#define KCUR0 2

#define KCUR1 2

#define KCUR2 2

#define KNUM0 3

#define KNUM1 5

#define KNUM2 1

float media\_de\_aleatorios(int ID)

{

int \* p = (int \*) malloc(N\*sizeof(int));

int i;

float media = 0;

for(i = 0; i < N ; i++)

{

p[i] = rand()%9 + 1;

media += p[i];

}

return media;

free(media\_de\_aleatorios);

}

int main()

{

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

srand(time(NULL));

printf("1o: %f\n", media\_de\_aleatorios(ID0));

printf("2o: %f\n", media\_de\_aleatorios(ID1));

printf("3o: %f\n", media\_de\_aleatorios(ID2));

getche();

return EXIT\_SUCCESS;

}

O problema da memória estava no acúmulo de números gravados na função media\_de\_aleatorios, resolvido utilizando free(media\_de\_aleatorios), que evita o vazamento de memória. Já o bug do código estava na inclusão da biblioteca time.h e utilização de srand(time(NULL)) para permitir a geração de números aleatórios, diferente das sequências iniciais que demonstravam sempre os mesmos valores.

Além disso a função intitula-se média dos valores quando na verdade ela realiza um somatório, logo o somatório deveria ser dividido por 10, e os parâmetros ID enviados a ela não são utilizados para seu cálculo.

---

QUESTÃO 3

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define KANO0 2

#define KANO1 3

#define KANO2 3

#define KCUR0 2

#define KCUR1 2

#define KCUR2 2

#define KNUM0 3

#define KNUM1 5

#define KNUM2 1

double f(double x) {

double y = 0.0;

double PI = 4.0\*atan(1.0);

int ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

switch(ID2) {

case 0: y = x\*x-5\*x+6; break;

case 1: y = 2.0\*PI\*x; break;

case 2: y = PI\*x\*x; break;

case 3: y = 6\*x/PI; break;

case 4: y = x\*x\*2.0\*PI; break;

case 5: y = -x\*x+5\*x-6; break;

case 6: y = 3.5\*x-2.0; break;

case 7: y = PI\*x/2.0; break;

case 8: y = -PI\*x+0.1\*x; break;

case 9: y = 2.0\*x-3.0\*PI; break;

default: y = 0.0;

}

return y;

}

int main() {

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

srand(ID0\*100+ID1\*10+ID2);

double x, y; int i;

FILE \* arq;

remove("dados.dat");

arq = fopen("dados.dat","ab");

for(i = 0; i < 100; i++) {

x = (double) rand()/RAND\_MAX;

y = f(x);

fwrite(&y,sizeof(double),1,arq);

}

printf("Codigo: %d%d%d\n",ID0,ID1,ID2);

fclose(arq);

return EXIT\_SUCCESS;

}

Resultado obtido após rodar o programa:

Código 971

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#define KANO0 2

#define KANO1 3

#define KANO2 3

#define KCUR0 2

#define KCUR1 2

#define KCUR2 2

#define KNUM0 3

#define KNUM1 5

#define KNUM2 1

double media(double a, double b, double c) {

return (a+b+c)/3;

}

int main() {

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

FILE \* arq;

int idA, idB, idC;

double nA, nB, nC;

arq = fopen("dados.dat","rb");

if(arq == NULL) {

fprintf(stderr,"Arquivo inexistente!\n");

return EXIT\_FAILURE;

}

switch(ID2) {

case 1: idA = 13; idB = 14; idC = 64; break;

case 2: idA = 21; idB = 42; idC = 84; break;

case 3: idA = 23; idB = 37; idC = 46; break;

case 4: idA = 16; idB = 55; idC = 82; break;

case 5: idA = 9; idB = 33; idC = 76; break;

case 6: idA = 0; idB = 39; idC = 99; break;

case 7: idA = 10; idB = 86; idC = 92; break;

case 8: idA = 17; idB = 61; idC = 92; break;

case 9: idA = 11; idB = 24; idC = 77; break;

case 10: idA = 5; idB = 53; idC = 65; break;

default: idA = idB = idC = 0;

}

int ler;

for (ler=0;ler<idA;ler++)

{

fseek(arq,ler\*sizeof(double),SEEK\_SET);

fread(&nA, sizeof(double),1,arq);

}

for (ler=0;ler<idA;ler++)

{

fseek(arq,ler\*sizeof(double),SEEK\_SET);

fread(&nA, sizeof(double),1,arq);

}

for (ler=0;ler<idA;ler++)

{

fseek(arq,ler\*sizeof(double),SEEK\_SET);

fread(&nA, sizeof(double),1,arq);

}

fclose(arq);

printf("Matricula: %d%d%d\n",ID0,ID1,ID2);

printf("Media [%lf %lf %lf] = %lf\n",nA,nB,nC,media(nA,nB,nC));

getche ();

return 0;

}

Resultado obtido após rodar o programa:

Matrícula: 971

Media [0.418598 0.418598 0.418598] = 0.418598

A função adicionada ao programa contempla três opções de resultados, para idA, idB e idC. Em cada for é utilizado fseek (FILE \*fp,long numbytes,int origem) de modo a encontrar a posição de idA, idB ou idC no ponteiro, e em seguida o fread(void \*buffer, int numero\_de\_bytes, int count, FILE \*fp) encarrega-se de ler e guardar este valor em nA, nB e nC para que as médias possam ser calculadas no main.

QUESTÃO 4

Código original

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define KANO0 2

#define KANO1 3

#define KANO2 3

#define KCUR0 2

#define KCUR1 2

#define KCUR2 2

#define KNUM0 3

#define KNUM1 5

#define KNUM2 1

typedef

struct Aluno

{

char nome[256];

int matricula;

unsigned int idade;

}

Aluno;

void mostrar(Aluno aluno)

{

printf("Nome: %s\nMAT: %03d\nIdade: %u\n", aluno.nome, aluno.matricula,

aluno.idade);

}

void gravar(Aluno aluno)

{

FILE \* arq;

arq = fopen("registro.txt","w");

fwrite(&(aluno.nome),256,1,arq);

fwrite(&(aluno.matricula),sizeof(int),1,arq);

fwrite(&(aluno.idade),sizeof(unsigned int),1,arq);

fclose(arq);

}

int ler(FILE \* arq, Aluno \* paluno, unsigned int id)

{

fseek(arq,id\*sizeof(Aluno),SEEK\_SET);

int ok = fread(&(paluno->nome),256,1,arq);

fread(&(paluno->idade),sizeof(unsigned int),1,arq);

fread(&(paluno->matricula),sizeof(int),1,arq);

return ok;

}

void inicia()

{

remove("registro.txt");

Aluno aluno;

strncpy(aluno.nome,"Oswald",256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%11 + 17;

gravar(aluno);

strncpy(aluno.nome,"Samara", 256);

aluno.matricula = 011;

aluno.idade = 22;

gravar(aluno);

strncpy(aluno.nome,"Carolina", 256);

aluno.matricula = 022;

aluno.idade = 22;

strncpy(aluno.nome,"Alana", 256);

aluno.matricula = 016;

aluno.idade = 18;

gravar(aluno);

strncpy(aluno.nome,"Silvia", 256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%15 + 17;

gravar(aluno);

strncpy(aluno.nome,"Mickey", 256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%9 + 17;

gravar(aluno);

}

int main()

{

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

srand(ID0\*100+ID1\*10+ID2);

Aluno aluno;

FILE \* arq;

unsigned int i;

inicia();

arq = fopen("registro.txt","r");

i = 0;

while(!feof(arq))

{

if(ler(arq, &aluno, i))

mostrar(aluno);

i++;

}

fclose(arq);

getche();

return EXIT\_SUCCESS;

}

Resultado:

Nome: Mickey

Matricula: 019

Idade: 98

Código com alterações

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define KANO0 2

#define KANO1 3

#define KANO2 3

#define KCUR0 2

#define KCUR1 2

#define KCUR2 2

#define KNUM0 3

#define KNUM1 5

#define KNUM2 1

typedef

struct Aluno {

char nome[256];

int matricula;

unsigned int idade;

}

Aluno;

void mostrar(Aluno aluno) {

printf("> %s: MAT %03d\n: %d anos;\n", aluno.nome, aluno.matricula,aluno.idade);

}

void gravar(Aluno aluno) {

FILE \* arq;

arq = fopen("registro.txt","a");

fwrite(&(aluno.nome),256,1,arq);

fwrite(&(aluno.matricula),sizeof(int),1,arq);

fwrite(&(aluno.idade),sizeof(unsigned int),1,arq);

fclose(arq);

}

int ler(FILE \* arq, Aluno \* paluno, unsigned int id) {

arq = fopen("registro.txt","r");

fseek(arq,id\*sizeof(Aluno),SEEK\_SET);

int ok = fread(&(paluno->nome),256,1,arq);

fread(&(paluno->matricula),sizeof(int),1,arq);

fread(&(paluno->idade),sizeof(unsigned int),1,arq);

return ok;

}

void inicia() {

remove("registro.txt");

Aluno aluno;

strncpy(aluno.nome,"Oswald",256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%11 + 17;

gravar(aluno);

strncpy(aluno.nome,"Samara", 256);

aluno.matricula = 11;

aluno.idade = 22;

gravar(aluno);

strncpy(aluno.nome,"Carolina", 256);

aluno.matricula = 22;

aluno.idade = 22;

gravar(aluno);

strncpy(aluno.nome,"Alana", 256);

aluno.matricula = 18;

aluno.idade = 18;

gravar(aluno);

strncpy(aluno.nome,"Silvia", 256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%15 + 17;

gravar(aluno);

strncpy(aluno.nome,"Mickey", 256);

aluno.matricula = rand()%999 + 1;

aluno.idade = rand()%9 + 17;

gravar(aluno);

}

int main() {

int ID0 = (KANO0+KANO1+KANO2)%9 + 1,

ID1 = (KCUR0+KCUR1+KCUR2)%9 + 1,

ID2 = (KNUM0+KNUM1+KNUM2)%9 + 1;

srand(ID0\*100+ID1\*10+ID2);

Aluno aluno;

FILE \* arq;

unsigned int i;

inicia();

arq = fopen("registro.txt","r");

i = 0;

while(!feof(arq)) {

if(ler(arq,&aluno, i))

mostrar(aluno);

i++;

}

fclose(arq);

getche();

return EXIT\_SUCCESS;

}

No caso da função gravar, ao abrirmos um arquivo do tipo "w", toda vez que ele

é criado ocorre uma sobreposição em relação ao próximo arquivo gravado. Logo,

alterou-se essa parte do código para um arquivo do tipo "a".