```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "estruturas.h"

/*****************************************/
/*Samuel Anjos nº 13305                   */
/*Luis Camilo nº13987                    */
/*****************************************/

/* verifica se o cliente exite atraves de nif*/
struct listaClientes * existeClienteNif(struct listaClientes * head){

    /* campo pelo qual pretendemos pesquisa o lista ligada */
    int nif;
    /* struct que pretendemos para  editar */
    struct listaClientes * clientePretendido;
    /* perguntar ao utilizador qual o nif do funcionario */
    printf("%61s", " "); printf("Introduza o nif do Cliente:");
    scanf("%d", &nif);
    fflush(stdin);
    printf("\n");

    /* cliente prentendido para realizar a simulação*/
    clientePretendido = procuraNodeCliente(head, &nif);
    /* cliente existe */
    if (clientePretendido != NULL)
        return clientePretendido;
    else{ /* cliente não exites*/
        printf("%61s", " ");
        alteraCorTexto(12); printf("Cliente nao existe.\n"); alteraCorTexto(15);
    }

    printf("%61s", " "); system("pause");
    return clientePretendido = NULL;

}// fim função


/*****************************************************************************/
/* funçpão que vai controlar o pedir dos dados pessoais do cliente para a simulaçao*/
/*****************************************************************************/
void pedeDadosPessoais(char user[50], struct listaClientes * head, struct listaSimulacoes * simulacoesHead){
    char novoCliente;
    int flag = 1;

    /* struct que pretendemos para  editar */
    struct listaClientes * clientePretendido;
    /* struct para os dados do emprestio*/
    struct dadosPessoaisSimulacao * dadosSimulacao;

        /* verifica se o cliente ja existe atraves do nif*/
        clientePretendido = existeClienteNif(head);
        /*existe*/
        if (clientePretendido != NULL){
            /* pede os dados pessoais do cliente relevantes a simulacao*/
            dadosSimulacao = dadosPCliente(clientePretendido);
            /* cliente tem segundo propronente*/
            if (existeSegundoProponente())
                dadosSimulacao = dados2PCliente(dadosSimulacao);

            else/* cliente nao tem segurndo proponente*/
                dadosSimulacao->segundoProponente.existe = NAO;

            /* pedir os dados relevante ao emprestimo para a simulacao*/
            dadosSimulacao = dadosReferentesEmprestimo(dadosSimulacao);
            /*pedir os dados relevante ao serviços que os proponentes prentedm aderir */
            dadosSimulacao = dadosReferentesServicosProp(dadosSimulacao);

            //fim simulacao, mostra o outout da simulacao
            menuOutputSimulacao(user, dadosSimulacao, simulacoesHead);
```

```
        }// fim if

}// fim função


/* função para pedir os dados pessoais do proponente principal*/
struct dadosPessoaisSimulacao * dadosPCliente(struct listaClientes * clienteCurrente){

    int contaPoupanca;
    struct dadosPessoaisSimulacao * dadosSimulacao;

    dadosSimulacao = (struct dadosPessoaisSimulacao *) malloc(sizeof(struct dadosPessoaisSimulacao));
    if (dadosSimulacao == NULL){
        printf("Erro na allocacao de memoria para os dados pessoais da simulacao!");
        exit(1);
    }

    dadosSimulacao->cliente = clienteCurrente->cliente;
    alteraCorTexto(8);
    printf("%61s", " "); printf("Nome: %s\n", clienteCurrente->cliente.dadosClt.nomeProprio);
    printf("%61s", " "); printf("Apelido: %s\n", clienteCurrente->cliente.dadosClt.apelido);
    printf("%61s", " "); printf("NIF: %d\n", clienteCurrente->cliente.dadosClt.nif);
    printf("%61s", " "); printf("Data Nascimento: %d/%d/%d\n", clienteCurrente->cliente.dataNascimento.dia,
                                    clienteCurrente->cliente.dataNascimento.mes,
                                    clienteCurrente->cliente.dataNascimento.ano);
    printf("%61s", " "); printf("Rendimento Anual Bruto: "); alteraCorTexto(15);
    scanf("%f", &dadosSimulacao->rendAnualBruto);
    printf("%61s", " "); alteraCorTexto(8); printf("Encargo Mensal Outros Creditos: "); alteraCorTexto(15);
    scanf("%f", &dadosSimulacao->encrgMenslOutrCreditos);
    do{
        printf("%61s", " "); printf("Tem conta Poupanca-Habitacao:\n ");
        printf("%61s", " "); printf("\tNAO - 0\n");
        printf("%61s", " "); printf("\tSIM - 1\n");
        printf("%61s", " "); scanf("%d", &contaPoupanca);

    } while (contaPoupanca != NAO && contaPoupanca != SIM);

    dadosSimulacao->contaPupancaHabit = contaPoupanca;

    printf("%61s", " "); alteraCorTexto(8); printf("Agregado familiar:"); alteraCorTexto(15);
    scanf("%d", &dadosSimulacao->aggrFamiliar);

    return dadosSimulacao;

}// fim função


struct dadosPessoaisSimulacao * dados2PCliente(struct dadosPessoaisSimulacao * dadosSimulacao){
    int tamanho = 0; // index para validar numero de inteiros em um numero
    // segundo proponente existe
    dadosSimulacao->segundoProponente.existe = SIM;

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Nome proprio:"); alteraCorTexto(15);
        fflush(stdin);
    } while (!validaLetras(gets(dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.nomeProprio)));

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Apelido:"); alteraCorTexto(15);
    } while (!validaLetras(gets(dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.apelido)));

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Nif:"); alteraCorTexto(15);
        scanf("%d", &dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.nif);
        tamanho = validaNumInteiros(dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.nif);
        fflush(stdin);
    } while (tamanho != 9);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Telefone:"); alteraCorTexto(15);
        scanf("%d", &dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.telefone);
        tamanho = validaNumInteiros(dadosSimulacao->segundoProponente.dadsPessoaisSegPropont.telefone);
        fflush(stdin);
    } while (tamanho != 9);
```

```
    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Data nascimento(dia/mes/ano):"); alteraCorTexto(15);
        scanf("%d/%d/%d", &dadosSimulacao->segundoProponente.dataNascimento.dia,
            &dadosSimulacao->segundoProponente.dataNascimento.mes ,
            &dadosSimulacao->segundoProponente.dataNascimento.ano);
        fflush(stdin);
    } while (!validaData(dadosSimulacao->segundoProponente.dataNascimento.dia,
        dadosSimulacao->segundoProponente.dataNascimento.mes,
        dadosSimulacao->segundoProponente.dataNascimento.ano));

    printf("%61s", " "); printf("Rendimento Anual Bruto: "); alteraCorTexto(15);
    scanf("%f", &dadosSimulacao->segundoProponente.rendAnualBruto);

    return dadosSimulacao;

}//fim função

/* pergunta ao utilizador se existe um segundo proponente*/
int existeSegundoProponente(){
    char proponentes;

    printf("%61s", " "); printf("Existe segundo proponente: (s/n)");
    proponentes = _getche();
    printf("\n");

    if (proponentes == 's' || proponentes == 'S'){
        return 1;
    }

    else return 0;
}// fim função

/*
*   funçaco para pedir os dados referentes ao emprestimo
*/
struct dadosPessoaisSimulacao * dadosReferentesEmprestimo(struct dadosPessoaisSimulacao * dadosSimulacao){

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Finalidade do Emprestimo:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("COMPRA - 0\n");
        printf("%61s", " "); printf("CONSTRUCAO - 1\n");
        printf("%61s", " "); printf("OBRAS - 2\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.finalidEmprestimo);

    } while (dadosSimulacao->dadosEmprestimo.finalidEmprestimo != COMPRA &&
        dadosSimulacao->dadosEmprestimo.finalidEmprestimo != CONSTRUCAO &&
        dadosSimulacao->dadosEmprestimo.finalidEmprestimo != OBRAS);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Destino da Habitação:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("HAB_PROP_PERMANENTE - 0\n");
        printf("%61s", " "); printf("HAB_SECUNDARIA - 1\n");
        printf("%61s", " "); printf("HAB_PARA_ARRENDAMENTO - 2\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.destinoHabit);

    } while (dadosSimulacao->dadosEmprestimo.destinoHabit != HAB_PROP_PERMANENTE &&
        dadosSimulacao->dadosEmprestimo.destinoHabit != HAB_SECUNDARIA &&
        dadosSimulacao->dadosEmprestimo.destinoHabit != HAB_PARA_ARRENDAMENTO);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Tipo de Imovel:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("APARTAMENTO - 0\n");
        printf("%61s", " "); printf("MORADIA - 1\n");
        printf("%61s", " "); printf("TERRENO - 2\n");
        printf("%61s", " "); printf("ARMAZEM - 3\n");
        printf("%61s", " "); printf("GARAGEM - 4\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.tipoImovl);

    } while (dadosSimulacao->dadosEmprestimo.tipoImovl != APARTAMENTO &&
        dadosSimulacao->dadosEmprestimo.tipoImovl != MORADIA &&
        dadosSimulacao->dadosEmprestimo.tipoImovl != TERRENO &&
        dadosSimulacao->dadosEmprestimo.tipoImovl != ARMAZEM &&
        dadosSimulacao->dadosEmprestimo.tipoImovl != GARAGEM );
```

```
    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Tem Garagem:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.garagem);

    } while (dadosSimulacao->dadosEmprestimo.garagem != NAO &&
        dadosSimulacao->dadosEmprestimo.garagem != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Localizacao do Imovel:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("CONTINENTE - 0\n");
        printf("%61s", " "); printf("REGIAO_AUTONOMA - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.localDoImovel);

    } while (dadosSimulacao->dadosEmprestimo.localDoImovel != CONTINENTE &&
        dadosSimulacao->dadosEmprestimo.localDoImovel != REGIAO_AUTONOMA);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Taxa de Juro:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("INDX_EURIBOR_3MES - 0\n");
        printf("%61s", " "); printf("INDX_EURIBOR_6MES - 1\n");
        printf("%61s", " "); printf("TAXA_FIXA - 2\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.taxaDeJuro);

    } while (dadosSimulacao->dadosEmprestimo.taxaDeJuro != INDX_EURIBOR_3MES &&
        dadosSimulacao->dadosEmprestimo.taxaDeJuro != INDX_EURIBOR_6MES &&
        dadosSimulacao->dadosEmprestimo.taxaDeJuro != TAXA_FIXA);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Credito para condicao de deficiente:\n");
    alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.creditoDeficiente);

    } while (dadosSimulacao->dadosEmprestimo.creditoDeficiente != NAO &&
        dadosSimulacao->dadosEmprestimo.creditoDeficiente != SIM);


    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Imovel do Banco:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.imovelBanco);

    } while (dadosSimulacao->dadosEmprestimo.imovelBanco != NAO &&
        dadosSimulacao->dadosEmprestimo.imovelBanco != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Cliente com Protocolo:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->dadosEmprestimo.clienteProtocol);

    } while (dadosSimulacao->dadosEmprestimo.clienteProtocol != NAO &&
        dadosSimulacao->dadosEmprestimo.clienteProtocol != SIM);

    printf("%61s", " "); alteraCorTexto(8); printf("Montanate solicitado:"); alteraCorTexto(15);
     scanf("%d", &dadosSimulacao->dadosEmprestimo.montanteSolicitado);
    printf("\n");

    printf("%61s", " "); alteraCorTexto(8); printf("Valor da escritura:"); alteraCorTexto(15);
    scanf("%d", &dadosSimulacao->dadosEmprestimo.valorEscritura);
    printf("\n");

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Prazo (anos):"); alteraCorTexto(15);
        scanf("%d", &dadosSimulacao->dadosEmprestimo.prazoEmprest);
        printf("\n");
    } while (!validaPrazoEmprestimo(dadosSimulacao->dadosEmprestimo.prazoEmprest));
```

```
    return dadosSimulacao;
}// fim função

/*
*   função para validar prazo do emprestimo
*/
int validaPrazoEmprestimo(int numAnos){

    switch (numAnos){
        case 2:
        case 3:
        case 5:
        case 10:
        case 15:
        case 20:
        case 25:
        case 30:
        case 35:
        case 40:
        case 45:
        case 50: return 1;
        default: return 0;
    }// fim switch

}//fim função

/*
*   funçaco para pedir os dados referentes ao servicos que os proponentes vao adqurir
*/
struct dadosPessoaisSimulacao * dadosReferentesServicosProp(struct dadosPessoaisSimulacao * dadosSimulacao){

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Subscricao a Seguro de Vida para 1 ou 2 pessoas:");
    alteraCorTexto(15);
        scanf("%d", &dadosSimulacao->servicosProponentes.quantSegurVida);
    } while (dadosSimulacao->servicosProponentes.quantSegurVida != 1 && dadosSimulacao->servicosProponentes.
    quantSegurVida != 2);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Subscricao do Seguro Multirriscos:\n");
    alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.seguroMultirrisc);

    } while (dadosSimulacao->servicosProponentes.seguroMultirrisc!= NAO &&
        dadosSimulacao->servicosProponentes.seguroMultirrisc != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Domiciliacao Rendimentos:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.domicilRendimentos);

    } while (dadosSimulacao->servicosProponentes.domicilRendimentos != NAO &&
        dadosSimulacao->servicosProponentes.domicilRendimentos != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Servico - Cartao de Credito:\n"); alteraCorTexto(15)
    ;
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.produtosServicos.
    cartaoCredito);

    } while (dadosSimulacao->servicosProponentes.produtosServicos.cartaoCredito != NAO &&
        dadosSimulacao->servicosProponentes.produtosServicos.cartaoCredito != SIM);


    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Servico - Cartao de Debito:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.produtosServicos.cartaoDebito)
```

```
    ;

    } while (dadosSimulacao->servicosProponentes.produtosServicos.cartaoDebito!= NAO &&
        dadosSimulacao->servicosProponentes.produtosServicos.cartaoDebito != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Servico - Banco ONLINE:\n"); alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.produtosServicos.bancoOnline);

    } while (dadosSimulacao->servicosProponentes.produtosServicos.bancoOnline != NAO &&
        dadosSimulacao->servicosProponentes.produtosServicos.bancoOnline != SIM);

    do{
        printf("%61s", " "); alteraCorTexto(8); printf("Servico - Domicilio pagamentos periodicos:\n");
    alteraCorTexto(15);
        printf("%61s", " "); printf("NAO - 0\n");
        printf("%61s", " "); printf("SIM - 1\n");
        printf("%61s", " "); scanf("%d", &dadosSimulacao->servicosProponentes.produtosServicos.
    domiciPagmtPerid);

    } while (dadosSimulacao->servicosProponentes.produtosServicos.domiciPagmtPerid!= NAO &&
        dadosSimulacao->servicosProponentes.produtosServicos.domiciPagmtPerid != SIM);

    printf("%61s", " "); system("pause");
    return dadosSimulacao;
}// fim função


void encargosMensais(struct dadosPessoaisSimulacao * dadosSimulacao){

    //struct encargoMensal ENCARGO_MENSAL;

    int comissaoDossier;
    int comissaoAvaliacao;
    int registoProvisAquesisao;
    int registoProvisHipoteca;
    int IMT;
    int custoAteContrato;
    int despesasEscritura;
    int impostoSelo;
    int conversaoRegisto;
    int custosContratacao;
    int custoTotal;

    /* calculo comissao do dossier e avaliacao */
    if (dadosSimulacao->contaPupancaHabit == SIM){
        comissaoDossier = 0;
        comissaoAvaliacao = 230 / 2;

    }
    else{
        comissaoDossier = 275;
        comissaoAvaliacao = 230;
    }
    /* tabela precos registo provisorio*/
    registoProvisAquesisao = 165;
    registoProvisHipoteca = 178;
    conversaoRegisto = 48;
    IMT = tabelaIMT(dadosSimulacao->dadosEmprestimo.valorEscritura);
    impostoSelo = (0.0008 * dadosSimulacao->dadosEmprestimo.valorEscritura) + (0.0006 * dadosSimulacao->
    dadosEmprestimo.valorEscritura);

    despesasEscritura = 297 + (0.008 * dadosSimulacao->dadosEmprestimo.valorEscritura) + (dadosSimulacao->
    dadosEmprestimo.montanteSolicitado * 0.006);
    /* se existir conta habitacao*/
    if (dadosSimulacao->contaPupancaHabit == SIM)
        despesasEscritura = despesasEscritura / 2;

    alteraCorTexto(8); printf("Comissao do Dossier: %d\n", comissaoDossier); alteraCorTexto(15);

    alteraCorTexto(8); printf("Comissao de Avaliacao: %d\n", comissaoAvaliacao); alteraCorTexto(15);
    alteraCorTexto(8); printf("Registo Provisorio Aquesicao: %d\n", registoProvisAquesisao); alteraCorTexto
```

```
    (15);
    alteraCorTexto(8); printf("Registo Provisorio Hipoteca: %d\n", registoProvisHipoteca); alteraCorTexto
    (15);
    alteraCorTexto(8); printf("IMT:%d \n", IMT); alteraCorTexto(15);
    printf("Custos associados ate contrato: %d\n", custoAteContrato=comissaoAvaliacao + comissaoDossier +
    registoProvisAquesisao + registoProvisHipoteca + IMT);
    printf("\n");
    alteraCorTexto(8); printf("Despesas Escritura: %d\n", despesasEscritura); alteraCorTexto(15);
    alteraCorTexto(8); printf("Imposto Selo: %d\n",impostoSelo); alteraCorTexto(15);
    alteraCorTexto(8); printf("Conversao de Registo (IVA incluido): %d\n", conversaoRegisto); alteraCorTexto
    (15);
    printf("Custos de Contratacao: %d\n", custosContratacao = despesasEscritura + impostoSelo +
    conversaoRegisto);
    printf("\n");
    alteraCorTexto(8); printf("Custos Totais: %d \n", custoTotal = custoAteContrato + custosContratacao);
    alteraCorTexto(15);
    printf("\n");

    system("pause");

}// fim funçao

int tabelaIMT(int precoImovel){

    if (precoImovel <= 92407)
        return 0;
    if (precoImovel > 92407 && precoImovel <= 126403)
        return 1848;
    if (precoImovel > 126403 && precoImovel <= 172348)
        return 5640;
    if (precoImovel > 172348 && precoImovel <= 287213)
        return 9087;
    if (precoImovel > 287213 && precoImovel <= 574323)
        return 11959;
    else
        return (.06 * 574323 );

}

/*
*   carregamento de funcionarios de ficheiro binarios
*/
struct listaSimulacoes * carregaSimulacoes (struct listaSimulacoes * head) {

    FILE *fp;
    char tempString [100];
    struct listaSimulacoes  *temp;
    struct listaSimulacoes *current;

    fp = abreFicheiroBinario("simul.txt", "r");

    /* percorre o ficheio até ao fim */
    while (!feof(fp)){
        //current = (struct funcionario *) malloc (sizeof (struct funcionario));
        if ((current = (struct listaSimulacoes *) malloc(sizeof(struct listaSimulacoes))) == NULL){
            printf("Erro: reservad de memoria carrega funcionarios.");
            exit(1);
        }
        /* devolve 0 qunado não tem sucesso */

        fscanf(fp, "%s," ,tempString[0] );
            current->next = NULL;
            temp = head;

            /* preencher o primeiro node */
            if (head == NULL){
                head = current;
            }
            /* preencher o resto dos nodes */
            else{

                while (temp->next != NULL)
                    temp = temp->next;
```

```
                temp->next = current;
            }


    } // fim while

    fclose(fp);

    return head;
}// fim função



void guardarSimulacaoFicheiro(struct dadosPessoaisSimulacao * dadosSimulacao, struct listaSimulacoes *
    simulacaoHead){
    FILE *fp;

    struct listaSimulacoes *current;
    /* chama a função para abri o ficheiro */
    fp = abreFicheiroBinario("simu.txt", "w");

    current = simulacaoHead;
    /* escreve no ficheiro um node de cada vez da lista */
    while (current != NULL){


        fprintf(fp, "%s", current->simulacao.dadosCliente.nomeProprio);


    }

    fclose(fp);

}
```