CROSSBETS

Introdução

Este programa em C++ implementa um Cassino Virtual, onde o usuário pode escolher entre três jogos: Roleta, Roda da Fortuna e Jogo dos Dados. O jogador começa com um saldo inicial e faz apostas nos diferentes jogos. O programa realiza operações como sorteio aleatório e verificação de saldo, mantendo uma interface simples para interação contínua até o saldo acabar ou o jogador optar por sair.

Estrutura do Código

Neste projeto, utilizei bibliotecas nativas do C++ para garantir eficiência e simplicidade na implementação das funcionalidades do cassino.

- **`#include <iostream>`**
  Permite a entrada e saída de dados, como o uso de `cout` e `cin` para exibir mensagens no console e capturar entradas do usuário.
- **`#include <cstdlib>`**
  Contém funções úteis, como `rand()` para gerar números aleatórios e `srand()` para inicializar o gerador.
- **`#include <ctime>`**
  Utilizada para acessar a função `time()`, que permite definir a semente do gerador de números aleatórios com base no horário atual, garantindo maior aleatoriedade.
- **`#include <windows.h>`**
  Específica do sistema operacional Windows, permite utilizar a função `SetConsoleOutputCP()` para definir a codificação UTF-8 e exibir corretamente caracteres acentuados.

Variáveis Globais

```
using namespace std;  // Evita ter que digitar std:: toda vez que usamos funções da biblioteca padrão

void roleta();
void jogo_dos_dados();
void roda_da_fortuna();
void limparTela();
void solicitarSaldo();
void apostado();
int saldo, aposta, opcao;
char continuar;
```

Para tornar o código mais intuitivo e prático, optei por declarar as variáveis como globais, o que simplifica o gerenciamento de dados compartilhados entre funções. Além disso, organizei a lógica em funções (`voids`), o que melhora o fluxo do programa, deixando-o mais modular, legível e fácil de manter. Essa abordagem também evita repetições e facilita futuras modificações no código.

Funções do código

main()

```
int main() {
    // Configura o console para codificação UTF-8, garantindo a exibição correta de caracteres acentuados
    SetConsoleOutputCP(CP_UTF8);
    cout << "Bem-vindo(a) ao Cassino Virtual!\n";
    solicitarSaldo();  // Solicita o saldo inicial do jogador

    // Loop principal do menu do jogo
    while (true) {
        limparTela();  // Limpa a tela para uma interface mais limpa
        cout << "\nEscolha um jogo para jogar:" << endl;
        cout << "1. Roleta" << endl;
        cout << "2. Roda da Fortuna" << endl;
        cout << "3. Jogo dos Dados" << endl;
        cout << "4. Sair" << endl;
        cout << "Digite a opção desejada: ";
        cin >> opcao;

        // Verifica se a opção escolhida é válida
        if (opcao < 1 or opcao > 4) {
            limparTela();
            printf("Opção inválida. Tente novamente.\n");
            system("pause");  // Pausa até o usuário apertar uma tecla
            continue;  // Volta para o início do loop
        }

        // Se o jogador escolher sair, encerra o jogo
        if (opcao == 4) {
            limparTela();
            printf("Obrigado por jogar! Até a próxima!");
            break;  // Encerra o loop principal e o programa
        }

        limparTela();  // Limpa a tela após escolher o jogo



        // Escolhe o jogo com base na opção selecionada
        switch (opcao) {
            case 1:
                cout << "Você escolheu Roleta!" << endl;
                cout << "Um número entre 0 e 36 será gerado aleatoriamente" << endl;
                cout << "Tente adivinhar qual número foi gerado" << endl;
                roleta();  // Chama a função do jogo Roleta
                break;
            case 2:
                cout << "Você escolheu Roda da Fortuna!" << endl;
                cout << "Os prêmios possíveis são:\n{R$0, R$10, R$20, R$50, -R$10, -R$20, -R$50 e 2x}" << endl;
                cout << "Ao perder, você perde tanto o valor apostado quanto o valor do prêmio e vice-versa" << endl;
                cout << "Aposte o valor que quiser e boa sorte!\n" << endl;
                roda_da_fortuna();  // Chama a função do jogo Roda da Fortuna
                break;
            case 3:
                cout << "Você escolheu Jogo dos Dados!" << endl;
                cout << "Dois dados serão lançados aleatoriamente" << endl;
                cout << "Tente acertar a soma dos dois dados. (2 a 12)" << endl;
                jogo_dos_dados();  // Chama a função do jogo dos Dados
                break;
        }

        // Verifica se o saldo do jogador acabou e solicita novo saldo, se necessário
        if (saldo <= 0) {
            cout << "Saldo esgotado! Precisamos de um novo valor para continuar.\n";
            solicitarSaldo();
        }
    }
    return 0;  // Encerra o programa
}
```

Exibe o menu principal e solicita o saldo inicial e permite que o jogador escolha entre três jogos ou encerre o programa. Além de realizar verificações de saldo após cada jogada, há sempre verificação para conferir se a opção digitada é válida.

apostado()

```
void apostado(){
    do {
        cout << "Digite o valor da aposta: ";
        cin >> aposta;

        // Verifica se a aposta é válida em relação ao saldo
        if (aposta > saldo) {
            limparTela();
            printf("Aposta inválida! Seu saldo atual é: R$%d\n", saldo);
            cout << "Tente novamente.\n";
        }
    } while (aposta > saldo);  // Repete até o jogador digitar uma aposta válida
}
```

Verifica se o valor apostado é válido (ou seja, não excede o saldo) e solicita uma nova aposta até que seja informado um valor válido.

limparTela()

Limpa a tela do console utilizando o comando `system("CLS")`.

```
// Função para limpar a tela
void limparTela() {
    system("CLS");  // Comando para limpar a tela no Windows
}
```

solicitarSaldo()

```
// Função para solicitar o saldo inicial do jogador
void solicitarSaldo() {
    cout << "Digite seu saldo inicial: ";
    cin >> saldo;
}
```

Solicita ao jogador um saldo inicial para continuar jogando.

Jogos

O saldo é mantido entre os jogos, sendo solicitado novamente apenas quando é zerado. A lógica principal foi implementada utilizando uma estrutura `do-while`, cuidadosamente escolhida para permitir que o jogador decida se deseja continuar jogando. Essa abordagem garante que o jogo seja reiniciado automaticamente com a geração de novos números aleatórios, mantendo a experiência fluida e intuitiva. Caso o jogador opte por encerrar, o ciclo é interrompido de forma natural, retornando ao menu principal ou finalizando o programa.

roleta()

```cpp
// Função do jogo Roleta
void roleta() {
    do {
        srand(time(0));  // Inicializa o gerador aleatório com base no tempo
        int numero = rand() % 37;  // Gera um número aleatório entre 0 e 36
        int apostaNumero;
        apostado();
        limparTela();

        cout << "Aposte em um número entre 0 e 36: ";
        cin >> apostaNumero;

        // Verifica se o número apostado está dentro do intervalo permitido
        if (apostaNumero < 0 or apostaNumero > 36) {
            printf("Aposta inválida! Tente novamente: ");
            cin >> apostaNumero;
        }

        limparTela();  // Limpa a tela para exibir o resultado
        printf("O número sorteado foi: %d\n", numero);

        // Verifica se o jogador acertou o número sorteado
        if (apostaNumero == numero) {
            printf("Parabéns! Você venceu!\n");
            saldo += aposta;  // Adiciona o valor da aposta ao saldo
        } else {
            printf("Você perdeu!\n");
            saldo -= aposta;  // Subtrai o valor da aposta do saldo
        }

        printf("Seu saldo atual é: R$%d\n", saldo);

        // Se o saldo acabar, solicita novo saldo
        if (saldo <= 0) {
            cout << "Saldo esgotado! Precisamos de um novo valor para continuar.\n";
            solicitarSaldo();
        }

        cout << "Deseja jogar novamente? (s/n): ";
        cin >> continuar;
        limparTela();
    }
    while (continuar == 's' or continuar == 'S');
```

Um número entre 0 e 36 é gerado aleatoriamente, e o jogador tenta adivinhar qual foi sorteado. Se acertar, o valor apostado é adicionado ao saldo; se errar, o valor é subtraído. Após cada rodada, o saldo atualizado é exibido. A função `srand()` é utilizada para garantir que os números gerados sejam aleatórios em cada execução do programa. A seguir, são realizadas verificações para validar a aposta e compará-la ao número sorteado, determinando se o jogador ganhou ou perdeu.

roda_da_fortuna()

```cpp
// Função da Roda da Fortuna
void roda_da_fortuna() {
    do {
        const int NUM_SECOES = 9; // Total de seções na roda
        int premios[NUM_SECOES] = {0, 10, 20, 50, -10, -20, -50, 2}; // Prêmios correspondentes às seções (ganhos e perdas)
        srand(time(0));
        apostado();
        limparTela();
        cout << "Girando a roda...\n";

        for (int i = 3; i > 0; --i) {
            cout << i << "...\n";
            Sleep(1000);
        }
        limparTela();
        int resultado = rand() % NUM_SECOES;
        if (premios[resultado] > 0 && premios[resultado] != 2) {
            int ganho = aposta + premios[resultado]; // Ganho total
            printf("Parabéns! Você ganhou R$%d, seu saldo atual é: R$%d\n", premios[resultado], saldo + ganho);
            saldo += ganho; // Atualiza o saldo com o ganho total
        }
        if (premios[resultado] < 0) {
            int perda = aposta + (-premios[resultado]); // Perda total
            printf("Perdeu tudo! Você perdeu R$%d e sua aposta, seu saldo atual é: R$%d\n", -premios[resultado], saldo - perda);
            saldo -= perda; // Atualiza o saldo com a perda total
        }
        if (premios[resultado] == 0) {
            printf("Você não ganhou nada desta vez. Seu saldo permanece: R$%d\n", saldo); // Não perdeu nem ganhou
        }
        if (premios[resultado]==2){
            printf("Parabéns! Você dobrou o seu valor, seu saldo atual é: R$%d\n", saldo); // dobra o valor da aposta
            saldo += aposta*2;
        }

        // Checa se o saldo ficou negativo
        if (saldo <= 0) {
            printf("Você não tem mais saldo para jogar. Jogo encerrado.\n");
            solicitarSaldo();
        }
        cout << "Deseja jogar novamente? (s/n): ";
        cin >> continuar;
        limparTela();
    }
    while (continuar == 's' || continuar == 'S');
```

O jogador gira a roda, que pode dar diferentes prêmios:
Ganho positivo (+10, +20, +50), Perda (-10, -20, -50), Sem ganho (0), Multiplicação por 2.
O saldo é atualizado com base no resultado da rodada, porém com um detalhe, caso perca algum valor, perde o valor sorteado e a aposta, o mesmo para os ganhos, se ganhar algum valor, leva o valor sorteado mais a aposta. E o resultado sem ganho é auto-explicativo, nem ganha, nem perde, o saldo continua o mesmo. Sempre verificando se o saldo ficou negativo e solicita novo saldo, se necessário.

Uma constante "`const int NUM_secoes`" define o número total de sessões da roda, garantindo a estrutura fixa do sorteio. Assim, ao invés de depender de valores dinâmicos, o programa utiliza essa constante para limitar o sorteio aos prêmios predefinidos. A função `rand()` é aplicada para selecionar aleatoriamente uma das opções de prêmios.

Para aumentar a emoção, implementei uma contagem regressiva antes da revelação do prêmio. Um loop `for` exibe números de 3 a 1, utilizando a função `sleep` para pausar a execução por 1 segundo entre cada número. Isso cria suspense e torna a experiência de jogo mais envolvente.

jogo_dos_dados()

```
// Função do jogo dos Dados
void jogo_dos_dados() {
    do {
        srand(time(0));
        int dado1 = rand() % 7;  // Gera um número entre 1 e 6
        int dado2 = rand() % 7;
        int resultado = dado1 + dado2;
        int apostaDados;
        apostado();

        limparTela();
        cout << "Aposte em um número entre 2 e 12: ";
        cin >> apostaDados;

        if (apostaDados < 2 or apostaDados > 12) {
            printf("Aposta inválida! Tente novamente: ");
            cin >> apostaDados;
        }

        limparTela();
        cout << "Dados lançados! Dado 1: " << dado1 << " + Dado 2: " << dado2 << " = " << resultado << endl;

        if (resultado == apostaDados) {
            printf("Parabéns! Você ganhou!\n");
            saldo += aposta;
        } else {
            printf("Você perdeu!\n");
            saldo -= aposta;
        }
        printf("Seu saldo atual é: R$%d\n", saldo);

        cout << "Deseja jogar novamente? (s/n): ";
        cin >> continuar;
        limparTela();
```

Dois dados são lançados, e o jogador aposta na soma deles. Se acertar, o saldo aumenta com o valor apostado; caso contrário, o valor é subtraído do saldo. Ao final de cada rodada, o resultado dos dados e o saldo atualizado são exibidos.

A função `srand()` é utilizada para gerar dois números aleatórios, e a soma deles é armazenada na variável `resultado`. Em seguida, verificações garantem que a aposta digitada seja válida. Se for, a aposta é comparada com o resultado para determinar se o jogador ganhou ou perdeu.

Conclusão

O Cassino Virtual foi desenvolvido aplicando conceitos fundamentais de C++, como o uso de variáveis globais, funções modulares e a estrutura `do-while` para permitir que o usuário decida se quer continuar jogando, reiniciando o sorteio de números aleatórios. O sistema foi projetado para rodar em plataformas Windows, exigindo apenas um compilador compatível com C++ (como **GCC** ou **MinGW**) e suporte para a função `system()` para limpeza e pausa da tela. Possíveis melhorias incluem: Compatibilidade multiplataforma, Interface gráfica e armazenamento de dados.