Programação Orientada a Objetos

Sistema de Gerenciamento para Clínica Odontológica

Prof.: Guillermo Camara Chavez

Alunos:
Arnaldo Dias
Bruno Pinto
Bernardo Emery
Roniela Gonçalves

Sumário

1. Introdução

A automação já é uma necessidade em vários estabelecimentos nos dias de hoje. A automação de vendas, controle de estoque, agendamentos entre outros, facilitam muito a vida de empreendedores e colaboradores, além de contribuir muito para a redução de custos da empresa. Uma ferramenta tão eficiente como essa, não poderia ficar de fora de clinicas e consultórios, médicos e odontológicos. Mas quais as vantagens? O que pode ser automatizado em um consultório?

Primeiramente podemos citar o aumento na produtividade da equipe, pois antes de automatizar o agendamento das consultas era necessário a contratação de um funcionário para atender ao telefone e conferir a agenda de cada um dos médicos. Com a automatização, basta que o próprio paciente escolha o horário de sua preferência, entre os que estão disponíveis no sistema. E assim, esse funcionário tem a oportunidade para dedicar a outras funções.

Também temos a vantagem da melhora da qualidade no atendimento, pois ajuda a padronizar todos os processos, desde o agendamento de consultas, até a emissão de faturas. Tudo isso representa mais qualidade ao reduzir erros humanos que poderiam comprometer o atendimento.

Outro caso que podemos citar é a facilidade na mobilidade dos colaboradores, pois com a automação de processos, uma mesma ferramenta pode ser acessada por várias pessoas ao mesmo tempo. Dessa forma, além de ter maior controle sobre o trabalho dos colaboradores, você facilita a mobilidade deles.

Mais um benefício é a possibilidade de maior segurança nos dados de pacientes, pois todas as informações ficam salvas numa mesma plataforma, não é preciso se preocupar com dados perdidos de pacientes. Nem é necessário preenchê-los novamente a cada atendimento, basta fazer uma atualização do cadastro. Esse tipo de sistema também é mais seguro por possibilitar o controle de acesso pelos colaboradores. Cada um deles pode ter uma configuração de permissões próprias.

Uma das mais importantes vantagens é a redução de custos e sustentabilidade com a menor produção de lixo do consultório, pois com a automatização da clínica o uso de pilhas de papel, tintas de impressoras e tantos outros materiais será praticamente extinta.

Com todas essas vantagens reunidas, podemos considerar outro grande benefício, que é a de reter e atrair novos pacientes, com uma clínica eficiente consegue-se agradar mais clientes, e eles por sua vez indicam novos pacientes, além de se fidelizarem à clínica.

Por esses e outros motivos, implementaremos nesse código um sistema de gerenciamento de uma clínica dentária, que pode ser aplicado para vários modelos de consultórios.

2. Descrição da Arquitetura e Implementação

A programação, como toda forma de linguagem, é um processo não-linear, de múltiplas etapas e processos, que se transformam perante cada nova mudança e desafio apresentado na razão de sua existência, tal como assim a descrevemos, assim foi a produção desse trabalho de Programação Orientada a Objetos.

Tomamos por etapa inicial a produção da UML, essa, talvez devido ao desconhecimento do time, revelou-se extremamente abstrata. Seu protótipo serviu apenas como o pistão inicial do projeto, onde a partir de sua execução, iniciamos a fase de implementação das classes.

Em seguida, guiamo-nos através da execução do menu principal, seguindo um modelo de prototipagem, onde cada iteração do código se revelava como um pequeno componente executável. Sendo assim, efetuamos o nascimento da classe principal, o menu de login, cadastro, seguido do serviço de arquivos para a permanência dos dados.

Tendo esse ponta pé inicial, produzimos os funcionários, tornamos o serviço de arquivos um processo genérico, apto a gerenciar os dados de todos os objetos da aplicação, o que se qualifica como um alto conceito de POO, pois tal serviço foi utiliza ao extremo por todos os objetos do arquivo. As classes SLT também foram muito utilizadas, seus algoritmos e containers, como vetores, e outras funções como sort, e etc.

Agenda teve alta complexidade na implementação, pois foi o recurso que mais contém objetos, além de uma lógica funcional diferente das demais. com dados variantes e operáveis em cada instância de seu objeto, porém uma vez implementada, serviu como uma base funcional para Folha de Ponto e Pagamentos, que foram facilmente implementados.

Casts não foram necessários, pois cada instancia opera separadamente sem necessidade de conversões, a alta modularização do programa, e a classe de chaves, permitiu referenciar cada objeto separadamente, essa decisão foi tomada também por havermos apenas um único arquivo para funcionários, usuários, agenda, etc. Na transformação de volta à memória principal, não houve uma solução lógica para converter cada usuário e funcionário a seu respectivo tipo. Para sanar essa dificuldade,

criou-se o Objeto chave, que é um objeto contido por todas as classes , que identifica cada instância de um objeto e valida seus dados perante o sistema.
Talvez, utilizando um sistema de armazenamento de dados mais complexo, como um banco de dados, ou instanciado arquivos separadamente para cada objeto derivado, pudéssemos utilizar as conversões dinâmicas, estáticas, etc. Porém devido ao rumo tomado, não foi necessário.

Devido à falta de tempo, faltou apenas a implementação de Try/Catchs para validar cada entrada de dados manipulada pelo sistema. O sistema admite uma gama de dados inválidos, apesar de não permitir a continuidade da execução em certos pontos (em que sua validação foi feita através de do/whiles e ifs), apesar da falta de tempo, temos o conhecimento da ferramenta e de seu poder na consolidação do sistema.

Também, foi pensado na utilização de MULTIMAPS e MAPS, para criar pares ordenados de objetos e organiza-los ordenadamente através de suas chaves, mas não pudemos realizar a devida implementação do recurso. Desenvolvemos uma função protótipo e genérica de ordenação, que utilizaria o recurso, sort da classe de vector para fazer a ordenação e reinserir no arquivo, mas novamente, devida a falta de tempo, não foi devidamente implementada.

3. Testes de Execução

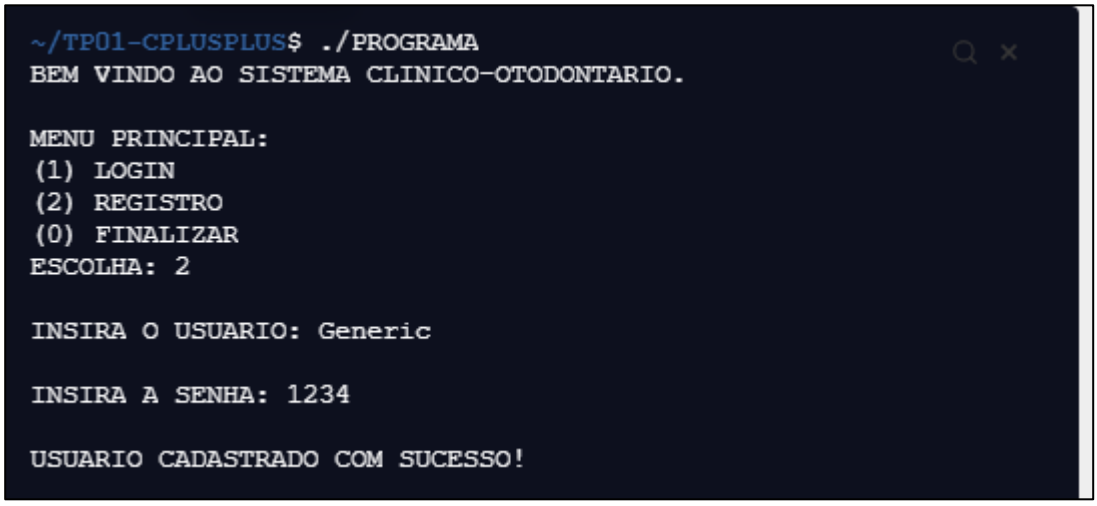

Figura 1. Teste de Registro

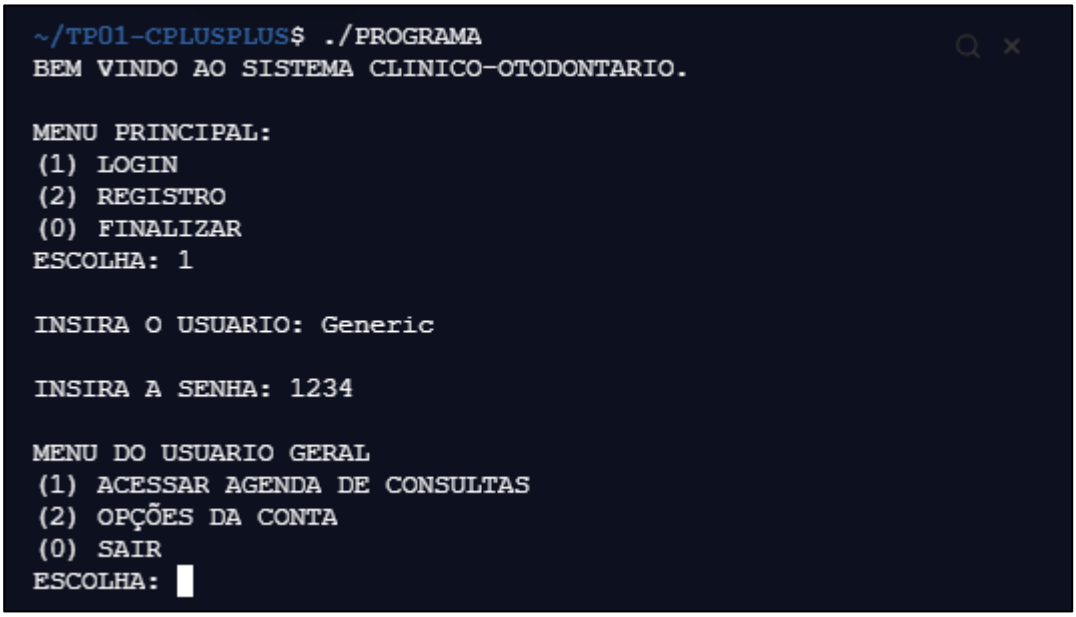

Figura 2 - Teste de Login.

```
~/TP01-CPLUSPLUS$ ./PROGRAMA
BEM VINDO AO SISTEMA CLINICO-OTODONTARIO.

MENU PRINCIPAL:
(1) LOGIN
(2) REGISTRO
(0) FINALIZAR
ESCOLHA: 0

FINALIZANDO A EXECUÇÃO.~/TP01-CPLUSPLUS$
```

Figura 3 - Teste de saída da primeira aplicação.

```
~/TP01-CPLUSPLUS$ ./PROGRAMA
BEM VINDO AO SISTEMA CLINICO-OTODONTARIO.

MENU PRINCIPAL:
(1) LOGIN
(2) REGISTRO
(0) FINALIZAR
ESCOLHA: 1

INSIRA O USUARIO: Generic

INSIRA A SENHA: 1234

MENU DO USUARIO GERAL
(1) ACESSAR AGENDA DE CONSULTAS
(2) OPÇÕES DA CONTA
(0) SAIR
ESCOLHA: 1

MENU DE AGENDA DE CONSULTAS.
(1) AGENDAR UMA CONSULTA
(2) LISTAR AGENDA
(3) DESMARCAR UMA CONSULTA
(4) HISTORICO DE CONSULTAS
(0) SAIR
ESCOLHA:
```

Figura 4 - Aplicações para o usuário e agenda de Consultas.

```
MENU DE AGENDA DE CONSULTAS.
 (1) AGENDAR UMA CONSULTA
 (2) LISTAR AGENDA
 (3) DESMARCAR UMA CONSULTA
 (4) HISTORICO DE CONSULTAS
 (0) SAIR
ESCOLHA: 1

|       NOME |     CDOMD|
INSIRA O CDOMD DE UM DOS ESPECIALISTAS:
```

Figura 5 – Menu agenda de consulta.

```
MENU DE AGENDA DE CONSULTAS.
 (1) AGENDAR UMA CONSULTA
 (2) LISTAR AGENDA
 (3) DESMARCAR UMA CONSULTA
 (4) HISTORICO DE CONSULTAS
 (0) SAIR
ESCOLHA: 1

|       NOME |     CDOMD|
|GenericEspecialista01|        01|
|GenericEspecialista01|        02|
|GenericEspecialista03|        03|
INSIRA O CDOMD DE UM DOS ESPECIALISTAS: 01
INSIRA O MES DE 2021 QUE DESEJA ACESSAR
 (1) Janeiro
 (2) Fevereiro
 (3) Março
 (4) Abril
 (5) Maio
 (6) Junho
 (7) Julho
 (8) Agosto
 (9) Setembro
 (10) Outubro
 (11) Novembro
 (12) Dezembro
ESCOLHA:
```

Figura 6 – Menu agenda de consulta.

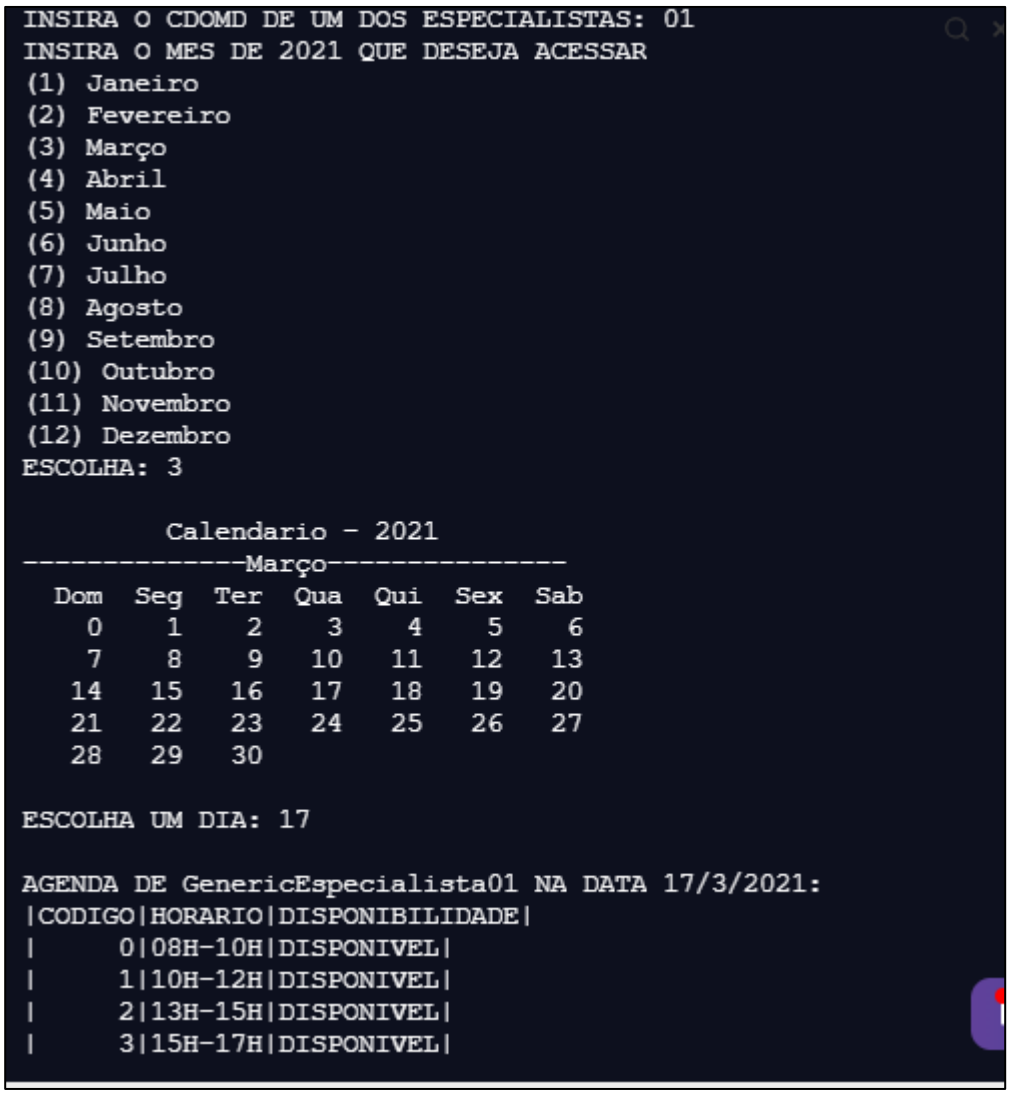

Figura 7 - Agenda de Consultas com o calendário.

```
AGENDA DE GenericEspecialista01 NA DATA 17/3/2021:
|CODIGO|HORARIO|DISPONIBILIDADE|
|     0|08H-10H|DISPONIVEL|
|     1|10H-12H|DISPONIVEL|
|     2|13H-15H|DISPONIVEL|
|     3|15H-17H|DISPONIVEL|
|     4|17H-19H|DISPONIVEL|

INSIRA O CODIGO DO HORARIO PARA AGENDARUMA CONSULTA MO DIA 17/3
/2021
(9) CANCELAR
ESCOLHA: 0

DIGITE O NOME DO PACIENTE: GenericPaciente01

CONSULTA AGENDADA COM SUCESSO!
```

Figura 8 - Horários para agendamento de consultas.

```
~/TP01-CPLUSPLUS$ ./PROGRAMA
BEM VINDO AO SISTEMA CLINICO-OTODONTARIO.

MENU PRINCIPAL:
(1) LOGIN
(2) REGISTRO
(0) FINALIZAR
ESCOLHA: 1

INSIRA O USUARIO: admin

INSIRA A SENHA: admin

MENU DO ADMINISTRADOR
(1) ADMINISTRAR USUARIOS
(2) ADMNISTRAR FUNCIONARIOS
(3) AGENDA
(4) FOLHA DE PONTO
(5) PAGAMENTOS
(0) SAIR
ESCOLHA:
```

Figura 9 - Funções para o Administrador

```
MENU ADMINISTRATIVO DE FUNCIONARIOS
(1) setFileR UM NOVO FUNCIONARIO
(2) LISTAR FUNCIONARIOS
(3) ALTERAR DADOS DE UM FUNCIONARIO
(4) REMOVER UM FUNCIONARIO
(0) SAIR DO MENU
ESCOLHA: 1

INSIRA O NOME: GenericEspecialista01

INSIRA O CPF: 00000000000

(1) ESPECIALISTA
(2) ASSISTENTE
(3) RECEPCIONISTA
ESCOLHA: 1

INSIRA O CDOMD DO ESPECIALISTA: 01

FUNCIONARIO CADASTRADO COM SUCESSO!
```

Figura 10 - Menu administrativo.

```
MENU DO ADMINISTRADOR
(1) ADMINISTRAR USUARIOS
(2) ADMNISTRAR FUNCIONARIOS
(3) AGENDA
(4) FOLHA DE PONTO
(5) PAGAMENTOS
(0) SAIR
ESCOLHA: 5

MENU DE PAGAMENTOS
(1) ADICIONAR UM PAGAMENTO
(2) LISTAR PAGAMENTOS
(0) SAIR
ESCOLHA: 1
INSIRA O TIPO DE PAGAMENTO:
(1) AGUA
(2) LUZ
(3) ALUGUEL
(4) TELEFONE
(5) PRODUTOS DE LIMPEZA
(6) MANUTENÇÃO DE EQUIPAMENTOS
(7) MATERIAL DE ESCRITORIO
(8) MATERIAL DE ATENDIMENTO
ESCOLHA: 8
INSIRA A DESCRIÇÃO DO PAGAMENTO DE Material de Atendimento:
```

Figura 11 - Funções para Administrador e Realização de Pagamentos Diversos.

4. Observações

Link do repositorio no github:

https://github.com/bruno-augusto-pinto/TP01-CPLUSPLUS

Caso o projeto não compile em Windows, utilize a plataforma Replit para compilar online:

https://replit.com

Link do projeto no Replit (Também pode ser importado diretamente através do Github):

https://replit.com/@BernardoEmery/TP01-CPLUSPLUS-1

5. Conclusão

Observou-se que em trabalhos de códigos de grande porte, como foi o trabalho aqui apresentado, a programação orientada a objetos se faz indispensável. Com ela foi possível entender e organizar o código de forma eficiente e assim conseguir um resultado satisfatório no desenvolvimento do mesmo. A abstração nos ajudou a compreender o funcionamento do sistema de forma mais simples, representando os objetos como algo real ou virtual e assim, relacionando o programa com algo já conhecido. A divisão em classes nos ajudou a entender como cada etapa do programa se dividia e quais funcionalidades e características seriam atribuídas. Sendo assim, mesmo com a grande complexidade de funções exigidas no sistema, pode-se construir um sistema capaz de satisfazer todos requisitos pedidos de forma competente. Pode se dizer também que mesmo sem sua interface gráfica, poderia facilmente ser utilizado para desenvolver sistemas completos para aplicações práticas.

6. Referências

https://stackoverflow.com/

https://www.cplusplus.com/reference/

https://docs.microsoft.com/pt-br/cpp/?view=msvc-160

https://www.simdoctor.com.br/blog/automacao-de-processos-em-clinicas-e-consultorios/