Conhecimentos Básicos (Trainee e Junior)

1 - A linguagem AdvPL é usada para desenvolver para qual ERP?

A) SAP Business One
B) TOTVS Protheus
C) Oracle ERP Cloud
D) Microsoft Dynamics ERP
E) TOTVS Datasul

R: Alternativa B - A Linguagem AdvPL é usada para criar customizações no ERP TOTVS Protheus, antigo Microsiga

2 - Atualmente qual é o IDE mais usado pela comunidade para desenvolver códigos em AdvPL e qual é o comando para compilar um arquivo .prw?

A) VSCode e Ctrl + F8
B) TDS (Eclipse) e Ctrl + F8
C) VSCode e Ctrl + F9
D) TDS (Eclipse) e Ctrl + F9
E) Dev Studio e Ctrl + F9

R: Alternativa C - O Dev Studio já foi descontinuado há muito tempo, e o TDS (baseado em Eclipse) irá ser descontinuado em 2020.

O VSCode vem ganhando cada vez mais apoio da comunidade e mais usuários.

O comando padrão para fazer a compilação / re-compilação de fontes é Ctrl + F9

3 - Quais são os dois tipos mais comuns de funções que programadores podem criar em AdvPL (com exceção de analistas com chave de compilação)?

A) User Function e Static Function
B) Local Function e Static Function
C) User Function e Private Function
D) User Function e Public Function
E) User Function e Function

R: Alternativa A - As funções sem prefixo (apenas Function), somente os analistas com chave de acesso podem crar e compilar.

Já para outros analistas, os dois tipos comuns são User Function (que são executadas com u_ antes do nome) e as Static Function (chamadas apenas dentro do próprio fonte)

4 – Cite e exemplifique pelo menos 3 tipos de valores em AdvPL?

R: A linguagem AdvPL possui vários tipos de valores para as variáveis. Abaixo segue uma lista das mais comuns:

A - Array - Conjunto de valores podendo ser como vetor ou matriz (unidimensional ou multidimensional)

C - Char / Character - Valor com conteúdo textual, podendo armazenar desde uma única letra, até inúmeros parágrafos

B - Block / CodeBlock - Blocos de código, que são executados por funções externas como eVal aeVal, Processa, etc

D - Date - Valor do tipo Data, no padrão DD/MM/YY ou DD/MM/YYYY (dependendo do profile do usuário), mas quando é salvo no banco de dados é transformado em caracter

L - Logical - Valor do tipo Lógico, podendo ser verdadeiro (.T. / .Y.) ou falso (.F. / .N.)

N - Numeric - Valor do tipo Numérico, podendo ser desde um valor inteiro (sem valor decimal) como valor com ponto flutuante (com valor decimal)

O - Object - Objeto instanciado de uma Classe, como uma tela ou um relatório

U - Undefinied - Valor indefinido, com o conteúdo nulo (Nil)

5 - A linguagem AdvPL possui tipagem de dados? Justifique sua resposta.

R: Antes de dezembro de 2014, não existia uma tipagem definida de dados, então uma mesma variável em AdvPL poderia ter tanto o conteúdo numérico como caracter.

Depois foi feito a alteração e incorporado isso nas novas versões do Protheus, dessa forma, hoje em dia existe a Tipagem de Dados em AdvPL. Para utilizá-la é bem simples, basta ao criar a variável, colocar o tipo na frente, por exemplo:

```
Local nIdade AS NUMERIC
```

Para saber mais, leia - https://tdn.totvs.com/display/tec/Tipagem+de+Dados

Conhecimentos Intermediários (Pleno)

1 - O que as variáveis cFilAnt e dDataBase tem em comum?

A) São constantes
B) São variáveis públicas
C) São variáveis privadas
D) São variáveis estáticas
E) São variáveis locais

R: Alternativa B - São variáveis públicas carregadas na inicialização do Protheus ao fazer login com algum usuário, sendo que o cFilAnt contém a Filial e dDataBase contém a Data do Sistema

2 - O que são MV_PAR01 a MV_PAR60?

A) Variáveis públicas
B) Parâmetros do Banco de Dados (SX1)
C) Constantes
D) Variáveis privadas
E) Nenhuma das alternativas anteriores

R: Alternativa A - São variáveis públicas que ficam na memória do Protheus, e são atualizadas ao chamar a função Pergunte() ou ParamBox()

3 - Levando em consideração o seguinte cenário, possuo um Array muldimensional onde a primeira coluna é a matrícula do funcionário, a segunda coluna é o nome e a terceira coluna é a data de nascimento. Agora, pensando nesse caso, eu quero pesquisar pelo nome DANIEL, qual dos comandos abaixo estaria correto?

A) aScan(aDados, {|x| AllTrim(Upper(x[1])) == "DANIEL"})
B) aScan(aDados, {|x| AllTrim(Upper(x[1])) == "daniel"})
C) aScan(aDados, {|x| AllTrim(Upper(x[1])) == "Daniel"})
D) aScan(aDados, {|x| AllTrim(x[1]) == "DANIEL"})
E) Nenhuma das alternativas anteriores

R: Alternativa E - Pois o nome é a coluna número 2, e em todas as alternativas é usado a coluna 1. O comando mais correto possível para o cenário descrito na pergunta, seria:
aScan(aDados, {|x| AllTrim(Upper(x[2])) == "DANIEL"})

4 - Explique com suas palavras o que é um APO e o que é um RPO.

R: Quando um código fonte desenvolvido em AdvPL (como arquivo .prw) é compilado, ele é transformado em um pequeno binário, como se fosse um executável, esse binário é conhecido como APO (Advanced Protheus Objects).

Já um RPO, é repositório desses objetos, então cada fonte desenvolvido se transforma num artefato dentro do repositório, e quando acessamos alguma rotina, ela procura dentro desse RPO pelo APO correspondente.

Referência: https://tdn.totvs.com/display/tec/AdvPL

5 - Interprete o trecho abaixo, explicando linha a linha o que está sendo feito:

```
aPergs := {}
aAdd(aPergs, {1, "Produto De",  cPar0,  "", ".T.", "SB1", ".T.", 80,  .F.})
aAdd(aPergs, {1, "Produto Ate", cPar1,  "", ".T.", "SB1", ".T.", 80,  .T.})
aAdd(aPergs, {1, "Data De",     dPar2,  "", ".T.", "", ".T.", 80,  .F.})
aAdd(aPergs, {1, "Data Ate",    dPar3,  "", ".T.", "", ".T.", 80,  .T.})

If ParamBox(aPergs, "Informe os parametros")
      oReport := fReportDef()
      oReport:PrintDialog()
EndIf
```

R: Cria um array vazio chamado aPergs
Adiciona um conteúdo nesse array, com o texto Produto De, com uma variável chamada cPar0, com a consulta padrão da SB1 e com uma largura de 80
Adiciona um conteúdo nesse array, com o texto Produto Ate, com uma variável chamada cPar1, com a consulta padrão da SB1, com uma largura de 80 sendo obrigatório
Adiciona um conteúdo nesse array, com o texto Data De, com uma variável chamada dPar2 e com uma largura de 80
Adiciona um conteúdo nesse array, com o texto Data Ate, com uma variável chamada dPar3 com uma largura de 80 sendo obrigatório

Chama a função ParamBox passando esse array e um texto Informe os parâmetros
Se a funcao acima for confirmada, cria um objeto chamado oReport, chamando uma funcao chamada fReportDef()
Chama o metodo PrintDialog() nesse objeto oReport
Finaliza o Se

Conhecimentos Avançados (Sênior e Especialista)

1 - Explique sobre Orientação a Objetos em AdvPL, dando um exemplo de Classe, Método e Atributo

R: Orientação a Objetos é um paradigma de programação, onde eu consigo criar objetos e manipular eles com comandos específicos.

Em AdvPL, também existe esse paradigma, onde uma Classe pode ser instanciada em uma variável do tipo Objeto, e essa variável pode sofrer ações através de Métodos, assim como ter definições alteradas dos seus Atributos.

Pensando em um cenário hipotético, eu posso ter a classe Pessoa, com métodos como Andar e Comer, e atributos como Altura e Peso.

Em AdvPL, a maioria dos objetos são usados para criação de telas e relatórios. Um exemplo de classe que podemos citar é a TReport que é usada para criação de relatórios. Dentre os vários métodos dela, podemos citar o SetPortrait que é para definir por padrão a orientação como Retrato. E entre os atributos, podemos citar o nFontBody, que define o tamanho da fonte do texto.

Todo método usado em AdvPL, é precedido por : e sempre possui parênteses, então no exemplo acima, supondo que nosso TReport seja instanciado em uma variável chamada oReport, para chamar/executar o método usamos oReport:SetPortrait().

Já o Atributo, é similar ao método, mas ele pode ser alterado ou acesso diretamente, então no nosso exemplo, se quiséssemos alterar a fonte do relatório para o tamanho 12, o comando ficaria, oReport:nFontBody := 12.

2 - Explique sobre o conceito de MVC e como ele é utilizado em AdvPL

R: MVC é um conceito usado há tempos em programação, onde você separa as camadas do seu software, uma para ser responsável pelo visual (View), outra pela modelagem dos dados (Model), e outra pelo controle de ações (Controller).

Em AdvPL, o Controller é o próprio AppServer, ele que irá ser o responsável pelo fluxo de ações. Já o Model, nele ficará as tabelas definidas, assim como operações de relacionamento entre mais de uma tabela. O View, é responsável pela montagem visual dos dados do Model, dessa forma sendo possível até mesmo colocar campos totalizadores, e separar em blocos a visualização.

O Model, nós devemos criar uma função estática chamada de ModelDef. E o View, nós devemos criar uma função estática chamada de ViewDef. Com isso o nosso Controller será encarregado de fazer o resto.

Além de deixar o código mais "limpo", fica com uma melhor manutenção com essas separações, e inclusive para Pontos de Entrada, onde a mesma User Function, serve para vários

cenários diferentes, testando o ID de cada operação (por exemplo, antes de efetuar o Commit da operação, e após efetuar o Commit da operação).

3 - Explique sobre os dois tipos de relatórios mais comuns em AdvPL e como é o uso deles (TReport e FWMSPrinter)

R: A classe TReport é uma classe padrão, onde ela faz o controle de impressões de maneira automática (como quebra de páginas e definições de colunas). Através de classes auxiliares como TRFunction, TRCell, TRBreak, entre outras é possível parametrizar um relatório que pode ser gerado via spool, via pdf, via planilha, via html, etc.

Já a classe FWMSPrinter é para a criação de relatórios gráficos, com vários métodos (como SayBitmap e SayAlign), apesar de ela ser mais apresentável que o TReport, todo o controle dela deve ser feito por programação, então por exemplo, ao quebrar uma página, deve ser feito o controle de linhas em variáveis e a utilização de métodos como EndPage() e StartPage().

4 - Explique sobre a arquitetura de pastas do ERP (Protheus e Protheus Data), exemplificando como elas funcionam com a linguagem AdvPL

R: Basicamente na pasta Protheus, ela fica encarregada por todos os executáveis e binários do sistema, sendo que dentro dela, possui a pasta de APO com os repositórios, e ao compilar um fonte em AdvPL, ele é transformado em um pequeno executável que fica dentro desse repositório. Temos também as pastas de binários, sendo que temos 2 principais, o AppServer, que é o serviço encarregado por executar nosso ERP, fazendo a comunicação com o DbAccess (e consequentemente o Banco de Dados). Já o outro binário é o SmartClient, que é o executável que o usuário irá abrir e irá utilizar funções que estão dentro do RPO, então se houver alguma customização, ao clicar no menu via SmartClient, ele vai mandar um comando de execução para o AppServer, que irá buscar o objeto no repositório, e se necessário fará comunicação com o banco de dados via DbAccess.

Quanto a Protheus Data, ela que fica encarregada da inteligência do sistema. Antigamente, nela ficavam os menus, dicionários, profiles, etc... Hoje ela fica encarregada, em processos de atualização (UpdDistr) ou em uso de customizações também, onde é possível gerar arquivos e deixar armazenado em alguma subpasta para uso, via customização AdvPL.

5 - Cite 3 tabelas do dicionário de dados no Protheus e 3 tabelas padrão do sistema. Agora escolha 1 das tabelas do dicionário e desenvolva um exemplo acessando a tabela do dicionário (levando em consideração que o dicionário esteja no Banco de Dados), e desenvolva um exemplo acessando alguma tabela padrão do sistema (pode ser via query ou via comandos).

R: As tabelas mais conhecidas do Dicionário de Dados são:
SX1 - Grupo de Perguntas
SX2 - Tabelas do Protheus
SX3 - Campos do Protheus
SX6 - Parâmetros do Sistema
SX7 - Gatilhos do Protheus
SXA - Pastas / Agrupamentos de campos
SXB - Consulta Padrão
SIX - Índices das Tabelas

Para acessar uma tabela do Dicionário, nós podemos usar a função OpenSX, conforme exemplo abaixo:

```
cEmpresa := "02"
cAliasTmp := "SX3TST"
cFiltro   := "X3_ARQUIVO == 'AIB'"
OpenSXs(Nil, Nil, Nil, Nil, cEmpresa, cAliasTmp, "SX3", Nil, .F.)
(cAliasTmp)->(DbSetFilter({|| &(cFiltro)}, cFiltro))
(cAliasTmp)->(DbGoTop())
Alert(&("(cAliasTmp)->X3_CAMPO"))
(cAliasTmp)->(DbCloseArea())
```

Já as tabelas mais comuns do sistema (existem mais de 8.000), podemos destacar:
SB1 - Cadastro de Produtos
SB2 - Saldos em Estoque
SA1 - Cadastro de Clientes
SA2 - Cadastro de Fornecedores
SA3 - Cadastro de Vendedores
SA4 - Cadastro de Transportadoras
SC5 - Cabeçalho do Pedido de Venda
SC6 - Itens do Pedido de Venda
SC7 - Pedido de Compra

Para acessar via comando, é bem simples, conforme o exemplo:

```
cPedido := '012345'
DbSelectArea('SC5')
SC5->(DbSetOrder(1)) //Filial + Pedido
SC5->(DbGoTop())
If SC5->(DbSeek(FWxFilial('SC5') + cPedido))
      Alert("Estou no pedido " + SC5->C5_NUM )
EndIf
```

Agora, via cláusula SQL, posso executar com Embedded SQL, com a função PLSQuery ou até mesmo de modos mais antigos como TCQuery, abaixo um exemplo usando PLSQuery:

```
cPedido := '012345'
cQrySC5 := " SELECT " + CRLF
cQrySC5 += "      C5_NUM,  " + CRLF
cQrySC5 += "      C5_EMISSAO SC5  " + CRLF
cQrySC5 += " FROM  " + CRLF
cQrySC5 += "      " + RetSQLName('SC5') + " SC5 " + CRLF
cQrySC5 += " WHERE  " + CRLF
cQrySC5 += "      C5_FILIAL = '" + FWxFilial('SC5') + "'  " + CRLF
cQrySC5 += "      AND C5_NUM = '" + cPedido + "'  " + CRLF
cQrySC5 += "      AND SC5.D_E_L_E_T_ = ' ' " + CRLF
PLSQuery(cQrySC5, "QRY_SC5")

If ! QRY_SC5->(EoF())
      Alert("Estou no pedido " + QRY_SC5->C5_NUM )
EndIf
```