UWAGA: Kończąc egzamin musisz podpisać i zwrócić tą kartkę !!!

Wydział Informatyki PB. Egzamin z projektowania systemów aplikacyjnych. 27.06.2003
Imię i nazwisko (ew. + pseudonim)

1. (40p) Obiekt zrealizowany zgodnie z protokołem COM implementuje ciąg znaków (napis). Ciąg ten przechowywany jest w prywatnej tablicy znaków własnoręcznie zaalokowanej przez obiekt. Obiekt obsługuje trzy interfejsy: **IUnknown, INapis** oraz **ITablica**, z których ostatni pozwala na traktowanie obiektu jako tablicy. Metoda *Inicjuj(ptr)* służy do inicjalizacji obiektu i może być wywoływana **wyłącznie jeden raz** podczas życia obiektu, metoda *Długość* wpisuje pod adres *pWynik* aktualną długość napisu, metoda *Dołącz(ptr)* dołącza na koniec napisu drugi napis reprezentowany przez wskaźnik *ptr*. Metody *UstawZnak(i,c)* i *PobierzZnak(i,pWynik)* pozwalają na ustawienie i pobranie i-tego znaku, przy czym w drugim wypadku pobierany znak jest zapamiętywany pod adresem *pWynik*. Zakładamy, że możliwy jest dostęp do obiektu napisu z wielu wątków jednocześnie, przy czym metody obiektu są odpowiedzialne za synchronizację. Trzy interfejsy obiektu są zaimplementowane przy pomocy pojedynczej klasy **CNapis**, która dziedziczy po klasach **INapis** i **ITablica**. Podaj deklarację i definicję wszystkich metod tej klasy. Metody interfejsów powinny **zawsze** zwracać kody błędów oraz kod S_OK gdy nie ma błędów. Implementacja obiektu klasy (IClassFactory) oraz funkcji i zmiennych globalnych biblioteki DLL nie wchodzi w zakres tego zadania.

2. (40p) Aplikacja wyświetla na ekranie punkt w kolorze czerwonym. Może być on przesuwany o 1 piksel w dół, górę, lewo, i prawo przy pomocy klawiszy strzałek. Po wciśnięciu klawisza Enter program zapamiętuje aktualne współrzędne punktu. W ten sposób przesuwając czerwony punkt i używając Enter możemy zadać zdefiniować dowolną liczbę punktów. Aplikacja zapamiętuje (zapewne w jakieś strukturze danych) wszystkie zdefiniowane punkty. Liczba zdefiniowanych punktów może być dowolna (ograniczona przez rozmiar pamięci operacyjnej). Z chwilą wciśnięcia przycisku myszy punkt, w którym znajduje się kursor myszy, zostaje połączony liniami ze wszystkimi punktami wprowadzonymi przy pomocy klawiatury. Linie te widoczne są cały czas (nawet jeżeli okno zostanie zminimalizowane, a za chwilę przywrócone do poprzedniego rozmiaru) przez 20 sekund, po czym znikają. Wciśnięcie przycisku myszy w czasie, w którym linie są widoczne jest ignorowane. Podaj kod procedury okna.

3. (20p) Okno dialogowe ma dwa pola edycji o identyfikatorach IDC_EDIT1, IDC_EDIT2, przycisk o identyfikatorze IDOK, pole typu „static text" IDC_OUT oraz dwa przyciski typu „check-box" o identyfikatorze IDC_CHECK1 oraz IDC_CHECK2. Wciśnięcie przycisku IDCOK powoduje zamknięcie okna. W polu IDC_OUT w zależności od stanu przycisków IDC_CHECK1 oraz IDC_CHECK2 powinien się znajdować: a) napis z pola IDC_EDIT1 jeżeli wyłącznie przycisk ID_CHECK1 jest zaznaczony, b) napis z pola IDC_EDIT2 jeżeli wyłącznie przycisk ID_CHECK2 jest zaznaczony, c) konkatenacja napisów z pól IDC_EDIT1i IDC_EDIT2, jeżeli oba przyciski są zaznaczone, d) napis pusty, jeżeli żaden przycisk nie jest zaznaczony. Zakładamy, że po zmianie stanu dowolnego pola edycji lub dowolnego przycisku „check-box" pole IDC_OUT jest natychmiast odświeżane. Na początku oba przyciski są zaznaczone. Podaj kod procedury okna dialogowego.

Deklaracje interfejsów COM:

```
class IUnknown {
public: virtual STDMETHODIMP QueryInterface(REFIID riid,LPVOID *ppv)=0;
virtual STDMETHODIMP_(ULONG) AddRef()=0;
virtual STDMETHODIMP_(ULONG) Release()=0;
};
class INapis : public IUnknown {
public:   virtual STDMETHODIMP Inicjuj(char *ptr)=0;
          virtual STDMETHODIMP Dołącz(INapis *ptr)=0;
          virtual STDMETHODIMP Długość(int *pWynik)=0;
};
class ITablica : public IUnknown {
public:   virtual STDMETHODIMP UstawZnak(int i, char c)=0;
public:   virtual STDMETHODIMP PobierzZnak(int i, char *pWynik)=0;
};
```

Kody błędów COM (zwracane przez **wszystkie** metody z wyjątkiem *AddRef* i *Release*, które zwracają wartość licznika odniesień): S_OK, E_FAILED, E_NOINTERFACE, E_OUTOFMEMORY

Prototypy funkcji:

```
int strlen(char *p);
int strcpy(char *dst, char *src);
int strcat(char *dst, char *src); // łączenie dwóch napisów
void EnterCriticalSection(CRITICAL_SECTION *p)
void LeaveCriticalSection(CRITICAL_SECTION *p);
void DeleteCriticalSection(CRITICAL_SECTION *p)
void InitializeCriticalSection(CRITICAL_SECTION *p);
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
UINT SetTimer(HWND hWnd ,UINT nIDEvent,UINT uElapse,TIMERPROC lpTimerFunc);
BOOL KillTimer(HWND hWnd, UINT uIDEvent);
BOOL InvalidateRect(HWND hWnd,RECT *lpRect,BOOL bErase);
HDC BeginPaint(HWND hWnd, PAINTSTRUCT *pps);
BOOL EndPaint(HWND hWnd, PAINTSTRUCT *pps);
PostQuitMessage(int ExitCode);
LRESULT DefWindowProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
void SetPixel(HDC hDC,int x,int y, COLORREF clr);
void MoveTo(int hdc,int x,int y);
void LineTo(int hdc,int x,int y);
BOOL DlgProc(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam);
BOOL EndDialog(HWND hDlg,int Result);
UINT GetDlgItemText(HWND hDlg, int Item, char *pString, int MaxChars);
UINT SetDlgItemText(HWND hDlg, int Item, char *pString);
UINT IsDlgButtonChecked(HWND hDlg, int ButtonID) //zwraca BST_CHECKED, gdy check-box zaznaczony
BOOL CheckDlgButton(HWND hDlg, int ButtonID, int State) // State == BST_CHECKED zaznacza check-box
```

Niektóre komunikaty:

```
WM_LBUTTONDOWN: xPos=LOWORD(lParam); yPos=HIWORD(lParam);
WM_KEYDOWN: nVirtKey = (int) wParam // kod klawisza
(VK_UP,VK_DOWN,VK_LEFT,VK_RIGHT,VK_RETURN)
WM_COMMAND: ID=LOWORD(wParam);          // identyfikator okna kontrolnego lub polecenia menu
                NotifyCode=HIWORD(wParam); // kod powiadomienia
```