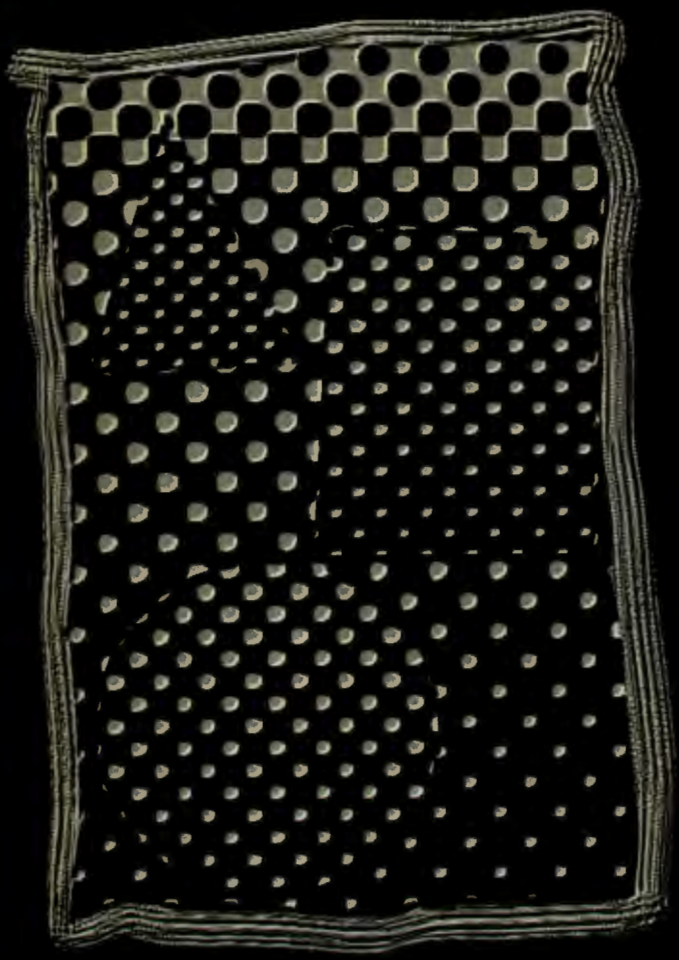
WYŻSZA SZKOŁA
INFORMATYKI STOSOWANEJ
I ZARZĄDZANIA



Henryk Spustek

ELEMENTY INFORMATYKI

WARSZAWA 2000

18.-

Seria: Skrypty WSISiZ

Skrypt zgłoszony przez
Dziekana Wydziału Zarządzania i Marketingu
dr Barbarę Mażbic-Kulmę

WYŻSZA SZKOŁA
INFORMATYKI STOSOWANEJ I ZARZĄDZANIA

Henryk Spustek

ELEMENTY INFORMATYKI

Warszawa 2000

ISBN 83-88311-17-4

Projekt graficzny okładki: Jan Młynarczyk

Druk:
Zakład Poligraficzny Jerzy Kosiński
Warszawa

6. JĘZYKI PROGRAMOWANIA

Sięgając w przeszłość – historię programowania, można wyróżnić cztery generacje języków programowania[28].

Języki pierwszej generacji to m.in. FORTRAN 1, COBOL. Charakterystyczną cechą programów użytkowych pisanych w tych językach był wspólny obszar danych do którego odwoływały się poszczególne podprogramy. Wszystkie podprogramy „widziały" całą strukturę danych. Było to niebezpieczne, ponieważ najmniejszy błąd w jednym z podprogramów miał zgubne skutki w całym programie. Struktura taka była też niewygodna w przypadku konieczności dokonywania wszelkiego rodzaju modyfikacji w programach.

W językach drugiej generacji dokonano modyfikacji podprogramów w taki sposób, że możliwym stało się przekazywanie parametrów pomiędzy nimi. Zapoczątkowało to powstanie strukturalnych metod programowania. Programowanie strukturalne to takie, gdzie oddzielnie definiuje się struktury danych oraz procedury i funkcje operujące na tych strukturach.

W trzeciej generacji języków programowania zostały wprowadzone moduły grupujące dane i podprogramy pozostające ze sobą w związku logicznym. Było to duże osiągnięcie w porównaniu do dwóch poprzednich generacji jako, że mniej uciążliwym stały się ewentualne modyfikacje kodu programów, jak również możliwe było tworzenie zespołów projektowych (w przypadkach tworzenia skomplikowanych programów) zajmujących się programowaniem określonych zagadnień.

Zupełnie inną jakość stanowią języki programowania czwartej generacji. Języki te nazywa się językami obiektowymi. Typowi przedstawiciele tej grupy to: Turbo Pascal for Windows i jego rozwinięcie w postaci DELPHI, Turbo C++ for Windows, Visual C, Visual Basic. Pierwszym językiem zawierającym elementy programowania obiektowego był SmallTalk, opracowany już w końcu lat sześćdziesiątych. Jednakże, dopiero w 1983 r. rozpowszechniły się metody programowania obiektowego, wraz z powstaniem języka C++. W językach obiektowych rozszerzono pojęcie struktury danych o jeszcze jeden element który nazwano obiektem.

---

[28] M.Kliszewski Inżynieria oprogramowania obiektowego, cz.1 Analiza obiektowa, Wydwanictwo Respekt Tomaszów Mazowiecki 1994

Zmieniło się całkowicie podejście do programowania. Strukturę programu obiektowego tworzą kolekcje powiązanych ze sobą obiektów. Na obiekt składają się dane i metody (algorytmy) działania na danych. Moduły zawierają zatem nie podprogramy, tak jak miało to miejsce w językach wcześniejszych generacji, a kolekcje obiektów. Programowanie zorientowane obiektowo lepiej odzwierciedla rzeczywistość. W życiu codziennym spotykamy się z konkretnymi rzeczami, których wygląd i zachowanie są postrzegane jako całość. Przykładem może być telewizor i jego obsługa. Nie wiedząc nic o budowie telewizora jesteśmy w stanie skorzystać z jego działania znając jedynie obsługę tego urządzenia.

Do podstawowych pojęć programowania obiektowego należą:

- klasa,
- obiekt,
- składowa,
- metoda,
- kapsułkowanie,
- dziedziczenie,
- polimorfizm,
- metody wirtualne.

Objaśnimy wyżej wymienione pojęcia[29].

Klasa – jest pełnym opisem pewnego abstrakcyjnego tworu, zawierającego określone cechy oraz procedury i funkcje umożliwiające manipulowanie nimi. Cechy (atrybuty) nazywamy *składowymi*, natomiast procedury i funkcje – *metodami* danej klasy. Dobrze skonstruowana klasa powinna umożliwiać dokonywanie wszystkich operacji na danych wyłącznie za pośrednictwem metod.

Obiekt – jest przedstawicielem klasy. O ile klasa jest tworem abstrakcyjnym, to obiekt nie. Wszystkie obiekty tej samej klasy zachowują wszystkie cechy i są identyczne względem siebie.

---

[29] Definicje o których mowa można znaleźć w J.Kierko Turbo Vision Programowanie obiektowe dla każdego, Wydawnictwo Mikom Warszawa 1994

Kapsułkowanie – proces ukrywania szczegółów konstrukcyjnych klasy, udostępniający programiście jedynie opis działania poszczególnych metod. Programista ma natomiast możliwość uzupełniania swojego obiektu o dodatkowe możliwości, poprzez dodanie nowych elementów (danych i metod) do opisu klasy wyjściowej, tworząc w ten sposób nową klasę. Stanowi to fundamentalną zasadę programowania obiektowego mówiącą o tym, że korzystając z klas uprzednio zdefiniowanych można tworzyć własne.

Dziedziczenie - mechanizm dzięki któremu możliwe jest apsułkowanie omówione wyżej. Mechanizm dziedziczenia powoduje, że każda następna klasa wywodząca się od danej klasy będzie dysponowała jej wszystkimi cechami.

Polimorfizm – polega na zdefiniowaniu wspólnych metod dla wszystkich obiektów potomnych i wskazaniu kompilatorowi do którego obiektu mają się odnosić używane metody. To dynamiczne powiązanie metod z obiektami realizowane jest przez metody *wirtualne*.

6.1 PROGRAMOWANIE W WINDOWS

Na początek, krótka historia Windows. W listopadzie 1983 r. firma Microsoft zapowiedziała utworzenie pakietu programów o nazwie Windows. Dwa lata później pojawiła się pierwsza wersja Windows opatrzona numerem 1.01, a już w dwa lata później na rynku oprogramowania pojawiła się kolejna wersja 2.0. Później powstały kolejno wersje Windows/386 (2.1) i w 1990 r. bardzo udana wersja 3.0. Kolejne ulepszenia Windows od wersji 1.01 do 3.0 polegały między innymi na tym, że możliwa stała się praca kilku programów przedstawionych w oddzielnych oknach (zastosowano tryb wirtualny procesora 386). Znacznie wygodniej można było posługiwać się myszą, powstały kolorowe ikony, proporcjonalne wielkości czcionek i przede wszystkim możliwym stał się dostęp do 16 MB pamięci. W komputerze z procesorem 386 można już było korzystać z wirtualnej przestrzeni adresowej. Zapewniało to wykorzystanie przestrzeni adresowej o wielkości cztery razy większej niż zainstalowana w komputerze pamięć fizyczna. Kolejną wersją systemu Windows była wersja 3.1 i jej rozwinięcie 3.11 umożliwiająca tworzenie lokalnych sieci komputerowych. W pełni 32 bitowy system operacyjny powstał w 1994 r. pod nazwą Windows NT. Rok 1995 – Microsoft wypuszcza na rynek system Wndows 95, który przynosi firmie

niezaprzeczalny sukces, poprzedzony zresztą szeroką akcją reklamową. Do głównych zalet tego systemu należy zaliczyć:

- wyższy poziom aplikacji wchodzących w skład pakietu,
- pojawienie się programów ukierunkowanych na korzystanie z sieci komputerowych,
- ulepszono realizację wielozadaniowości (jednoczesnego wykonywania kilku programów jednocześnie),
- wprowadzono możliwość wielowątkowości wykonywanych zadań (polega to na tym, że kilka modułów tego samego programu może być wykonywanych jednocześnie),
- efektywne wykorzystanie pamięci,
- możliwość wykorzystania przez różne aplikacje procedur zawartych w licznych bibliotekach dynamicznych (DLL),
- możliwość korzystania przez aplikacje z ponad 600 różnych funkcji systemowych zwanych funkcjami API[30].

Windows 95, tak jak i wszystkie poprzednie wersje systemu, nie był też pozbawiony wad. W porównaniu do poprzednich wersji, wymagania sprzętowe znacznie wzrosły. Uwidoczniło się też wiele błędów i niedoróbek. Początkowo mogło się wydawać, że jest to ostatnia wersja systemu Windows, a tymczasem .... Dzisiaj mamy już wersję 2000.

Niezaprzeczalnie, Windows stał się najpopularniejszym środowiskiem graficznym komputerów działających w oparciu o MS – DOS.

[30] S.Osiak, Programowanie w Windows, Wydwanictwo Mokom Warszawa 1997

6.1.1 PROGRAMOWANIE PRZEZ DUŻE P ... W WINDOWS

Programowanie w Windows stanowi zupełnie inną jakość w porównaniu z programowaniem w Dos. Programując w systemie Dos, do komunikowania się programu z systemem używa się odpowiednich przerwań (patrz przykłady programów w języku asembler rozdz. 8.1), zaś w Windows do tego celu wykorzystuje się zbiór funkcji, tworzących tzw. interfejs programowy aplikacji (ang. Application Programming Interface – API). Interfejs ten składa się z kilkuset funkcji podzielonych na grupy tematyczne, m.in. GDI – dotyczące grafiki, zarządzania oknami, itp. Tworzenie aplikacji multimedialnych stało się prostsze dzięki poleceniom MCI (ang. Media Control Interface), umożliwiające odtwarzanie dźwięku oraz filmów video.

Wszystkie, działające w Windows, aplikacje posiadają identyczny wygląd tzn. posiadają identyczny graficzny interfejs użytkownika (GUI). Wyświetlane na ekranie dane wyglądają identycznie na ewentualnym, późniejszym wydruku – technika WYSIWYG.

Często zachodzi konieczność bezpośredniego komunikowania się z poszczególnymi elementami sprzętu zainstalowanymi w komputerze IBM – PC. Taki, bezpośredni dostęp uzyskujemy poprzez wykorzystanie komponentów bibliotek DirectX. W skład DirectX wchodzą między innymi biblioteki zapewniające obsługę kart graficznych i dźwiękowych, obsługę myszy, joysticka, itp. Dostęp do procedur zawartych w DirectX odbywa się z poziomu języka C/C++[31].

Niezmiernie pożyteczny jest dostępny w Windows mechanizm OLE (ang. Object Linking and Embedding – łączenie i osadzanie obiektów). Od momentu powstania OLE jego zakres znacznie się poszerzył i obecnie OLE jest technologią, która pozwala na tworzenie komponentów, nadających się do wielokrotnego użycia na różnych platformach[32].

Przykładowo, Visual Basic umożliwia tworzenie aplikacji, które mogą udostępniać swoje obiekty innym aplikacjom. Są to tzw. Serwery OLE. Wykorzystując mechanizm OLE można umieścić wykresy bądź całe arkusze pochodzące z Excel wewnątrz aplikacji Visual Basica. Visual Basic używa

---

[31] A.Ślosarski, DirectX w przykładach, Wydawnictwo Mikom Warszawa 1999

[32] J.Karmański, Praktyczny kurs programowania w Windows 95, Wydawnictwo Helion Gliwice 1997

tym samym Excela jako serwera OLE. Mechanizm ten pozwala na wykorzystanie potęgi obiektów Excela bez konieczności pisania kodu dla wszystkich arkuszy czy wykresów[33]. Inny przykład, to wykorzystanie w aplikacji stworzonej na platformie Delphi silnego narzędzia jakim jest w arkuszu kalkulacyjnym Excel – Solver. Możliwym jest wykonanie aplikacji umożliwiającej wykorzystanie Solvera do zagadnień programowania liniowego w ten sposób, że z poziomu Delphi wywoływany jest arkusz kalkulacyjny Excel w którym wykonywane są wszelkie obliczenia, natomiast wyniki przekazywane zostają z powrotem do aplikacji głównej. W ten sposób użytkownik nie znając działania Solvera, może skorzystać z jego usług. Dostęp do Solvera odbywa się poprzez odpowiednie makro, zaś wywołanie arkusza Excel – zapewnia mechanizm OLE.

6.1.2 PROGRAMOWANIE PRZEZ MNIEJSZE P ... W WINDOWS

Obecnie nie wystarczy jedynie sprawna obsługa pakietu biurowego. Trzeba nauczyć się programować, po to aby zautomatyzować wybrane czynności[34]. W przypadku Worda i Excela użytkownik ma możliwość programowania w Visual Basicu, którego przodkami były starsze wersje języka Basic. VB dla aplikacji Worda czy też Excela nie jest językiem w pełni obiektowym, tłumaczy to tytuł podrozdziału[35]. Jednakże, nie jest to żadną przeszkodą w sprawnym wykorzystaniu niewątpliwego dobrodziejstwa jakim jest wbudowany język programowania VB. Użytkownik ma możliwość skorzystania z obiektów, właściwości i metod, których dokładny opis wraz z przykładami można znaleźć w pliku pomocy. W celu sprawnego posługiwania się dostępnymi obiektami należy dokładnie zapoznać się z ich opisem. W drugiej części skryptu znajdują się ćwiczenia, gdzie wykorzystano VB: makro To-To napisane dla aplikacji Word'97, funkcje użytkownika w Excelu, zegar analogowy – ćwiczenie w Excel.

---

[33] D.Kurata, Programowanie obiektowe w Visual Basicu, Wydawnictwo LT&P Warszawa 1997

[34] J.Korol, Visual Basic w Excelu 97, Wydawnictwo Mikom Warszawa 1998

[35] Dowód na prawdziwość tezy o niepełnej „obiektowości" tego języka można znaleźć w książce W.Leonhard, L.Hudspeth i T.J.Lee Excel '97 Dokuczliwości, Wydawnictwo READ ME Łódź 1998

Dotychczasowe wydawnictwa
WYŻSZEJ SZKOŁY INFORMATYKI STOSOWANEJ I ZARZĄDZANIA

- Z. Stachowiak: *Ekonomia. Zarys podstawowych problemów.* 1998; Wyd. 2. 2000.
- Z. Mikołejko: *Elementy filozofii.* 1998; Wyd. 2 popr. i rozsz. 1998; Wyd. 3 popr. i rozsz. 1999.
- W. Arczewska: *Bazy danych Oracle* 1998; Wyd. 2 popr. i rozsz. 1999; Wyd. 3 popr. 1999.
- S. Bożek, P. Cholajda, G. Szkatuła: *Wstęp do bazy danych MS Access dla Windows 95.* 1998.
- T. Łuba: *Podstawy układów logicznych.* 1998; Wyd. 2 popr. 1999.
- G. Szkatuła, A. Pogorzelec: *Ćwiczenia z bazy danych Microsoft Access 97.* 1999; Wyd. 2 rozsz. 1999.
- A. Żochowski: *L E M Laboratorium eksperymentów matematycznych.* 1999; Wyd. 2. popr. 2000.
- J. Hołubiec, red.: *Analiza systemowa w finansach i zarządzaniu. Wybrane problemy.* 1999.
- M. Doros: *Przetwarzanie obrazów. Materiały pomocnicze.* Cz.1, 2. 1999; Wyd. 2 popr. 1999.
- L. Oleksyn: *Istota, zakres i cechy rachunku kosztów.* 1999.
- L. Oleksyn: *Zadania rachunku kosztów w zarządzaniu.* 1999.
- L. Oleksyn: *Ekonomia - zarys wykładu.* 1999.
- Z. Nahorski: *Metoda najmniejszych kwadratów.* Cz. 1, 2. 1999.
- O. Hryniewicz: *Wykłady ze statystyki.* 1999.
- P. Cholajda: *Systemy informatyczne w MS ACCESS 97 PL.* 1999.
- K. Liderman: *Bezpieczeństwo informacji w systemach informatycznych.* 2000.
- M. Barszczewski: *Zarządzanie sieciami telekomunikacyjnymi.* 2000.
- J. Borkowski, M. Dyrda, L. Kanarski, B. Rokicki: *Wybrane problemy psychologii organizacji. O konflikcie i negocjacjach.* 2000.
- J. Jarmakiewicz: *Sieci teleinformatyczne.* Cz. 1, 2. 2000.
- T. Łuba: *Synteza układów logicznych.* 2000.
- H. Spustek: *Elementy informatyki.* 2000.

**WYŻSZA SZKOŁA
INFORMATYKI STOSOWANEJ
I ZARZĄDZANIA**

pod auspicjami
Polskiej Akademii Nauk

ZAŁOŻYCIELEM
Wyższej Szkoły Informatyki Stosowanej i Zarządzania
jest
Fundacja Krzewienia Nauk Systemowych
powołana z inicjatywy
Prezesa
POLSKIEJ AKADEMII NAUK

FUNDATOREM
Fundacji Krzewienia Nauk Systemowych
jest
POLSKA AKADEMIA NAUK

ORGANEM
sprawującym nadzór jest
MINISTERSTWO EDUKACJI NARODOWEJ

Wyższa Szkoła Informatyki Stosowanej i Zarządzania
prowadzi studia wyższe na kierunkach:
INFORMATYKA
ZARZĄDZANIE I MARKETING

SIEDZIBA
Instytut Badań Systemowych
Polskiej Akademii Nauk
ul. Newelska 6, 01-447 Warszawa

ISBN 83-88311-17-4