**Lab\_LPVOID\_1**

Na różnicę od poprzedniego zadania, teraz struktura obiektu MY\_POINT zawiera wskaźnik

struct MY\_POINT

{

double \*crd; //crd[0] - x, crd[1] - y

};

Oznacza to, że pamięć dla *crd* powinna być alokowana dynamicznie, a później – zwalniana. Dla tego powstają funkcji:

MY\_POINT \*CreateVertex(double x, double y);

void MY\_POINT\_Free(MY\_POINT \*\*ob);

void MY\_POINT\_FreeCoord(MY\_POINT \*ob);

Funkcja *CreateVertex* stosuje się przy dynamicznym tworzeniu obiektu MY\_POINT: dynamicznie alokuje pamięć dla całego obiektu MY\_POINT, a dalej – dla tablicy współrzędnych *crd* i inicjuje tą tablicę wartościami *x*, *y*.

Funkcja MY\_POINT\_Free zwalnia pamięć dla obiektu MY\_POINT, który powstał dynamicznie.

Jeśli obiekt MY\_POINT powstał statycznie (na przykład, w strukturze Triangle – MY\_POINT *vert*[3] każdy z elementów tablicy *vert* powstaje statycznie), wskaźnik *crd* jest nieinicjowany. Dla tego funkcja TriangInit inicjuje wskaźnik *crd* dla każdego wierzchołku. Alokowanie pamięci dla tablicy współrzędnych odbywa się w funkcji MY\_POINT\_Equel, a dla zwolnienia tej pamięci jest używana funkcja MY\_POINT\_FreeCoord.

Dla zwolnienia pamięci kontener MY\_ARRAY\_ELEM używa wskaźnik do funkcji FreeObject ptr\_fun\_free. Każdy obiekt graficzny (Triangle, Quadrate, Circle) powinien mieć funkcje, która zwalnia pamięć alokowaną dynamicznie (tu trzeba wyraźnie przedstawić, jakie to obiekty byli alokowane dynamicznie, a jakie – statycznie) i ma prototyp void (\*FreeObject)(void \*\* pdat); Wtedy każda taka funkcja może być wywołana przez wskaźnik do funkcji typu FreeObject . **Taka technika nadaje możliwość uniezależnić kod funkcji z obsługi kontenera MY\_ARRAY\_ELEM od typu danych.**

Zadaniem jest wprowadzenie podobnych zmian do innych graficznych obiektów. Dla obiektu Circle ilość wierzchołków powinna być przedstawiona przez zmienną, podawaną przez użytkownika. Oznacza to, że tablica MY\_POINT \**vert* powinna być alokowana dynamicznie.