Coding Style

Proiectarea Algoritmilor

2012

1 Introducere

Anul acesta, 10 puncte din cele 100 de puncte ce pot fi acordate pentru o tema vor fi rezervate lizibilitatii codului scris.
Vom prezenta mai jos cateva recomandari referitoare la coding style. Recomandam sa aderati la un coding style consistent, tinand cont, desigur, de particularitatile fiecarui limbaj de programare folosit.

2 Restrictii si recomandari

2.1 Lungime linii - 2 puncte

Lungimea maxima a liniilor va fi de **80 de caractere** pentru **C/C++**, respectiv de **100 de caractere** pentru **Java**. In general, se doreste ca dimensiunea liniilor de cod sa fie sub 80 de caractere, pentru orice limbaj folosit. Extinderea la 100 pentru Java s-a facut datorita gradului de verbozitate a acestui limbaj.

2.1.1 Exceptii

* linia contine un URL sau un exemplu de comanda mai lung de 80, respectiv 100 de caractere.

2.1.2 Exemple

```
// Corect
// Pentru a evita o linie lunga, parametrii functiei au fost mutati pe linii
// noi.
BigDistributedHashTable hash_table = new BigDistributedHashTable(
    server_paths, initial_size, num_concurrent_requests,
    settings_flags);
// O alta varianta este mutarea parametrilor, cate 1 pe fiecare noua linie,
// aliniati la functia din care fac parte
bool success = ProcessInteresingStructure(interesting_structure,
                                          true, // First call
                                          NULL); // No callback
```

2.2 Indentare consecventa - 2 puncte

Indentarea codului trebuie sa fie consecventa. Nu se vor alterna spatii si taburi pentru indentare, iar blocurile vor fi corect aliniate.
Pentru a evita problemele legate de dimensiunea taburilor si, implicit, de depasirea restrictiilor privind lungimea liniilor, se pot prefera spatii in loc de taburi.

2.2.1 Exemple

```
// Gresit
if (return_value == false) {
  database.Close();
  } // indentare gresita aici
```

2.3 Evitarea codului duplicat - 1 punct

Incercati sa evitati codul duplicat prin crearea de functii relevante. In cazul in care dimensiunea codului este mica, se pot folosi functii inline.

2.4 Evitarea functiilor kilometrice - 1 punct

Dimensiunea functiilor ar trebui sa fie mai mica de 150 de linii.

2.5 Evitarea codului comentat sau mort - 1 punct

Nu trebuie sa existe linii comentate sau ce nu vor fi executate de nicio ramura a logicii programului.
Cea mai comuna sursa de cod comentat o reprezinta trecerea de la depanare la produs finit, cand liniile in care sa facea logging trebuie ignorate.

2.5.1 Sugestii

* Ar trebui sa fiti consecventi cu privire la stilul si continutul comentariilor (de exemplu, nu alternati // cu /* ... */, nu explicati secventele banale sau evidente, nu folositi ascii art)

* Pentru logging, recomandam folosirea macrourilor *LOG* din folderul *utils/*

2.6 Denumirea adecvata a claselor, a variabilelor si a functiilor/metodelor - 1 punct

Incercati sa denumiti clar variabilele pentru dizambiguizare.

2.6.1 Exemple

```
// Gresit
int nt;          // Nu are nicio semnificatie
int n_wpc_conns; // Abreviere ambigua, wpc = ?
int num_rp;      // Remote Procedures ? Requested Protocols ?
```

‘

```
// Corect
int num_tables;
int num_tcp_connections; // Majoritatea progamatorilor stiu ce inseamna TCP
```

2.7 Evitarea folosirii valorilor hard-codate - 1 punct

2.7.1 Exemple

```
// Gresit
var return_value = ProcessSomeStuff();
switch (return_value) {
  case 0:
    // We have success
    break;
  case 1:
    // Can't reach host
    break;
  case 2:
    // Connection refused
    break;
  default:
    // End of the world
    break;
}
```

‘

```
// Corect (v.1)- Folositi MACROuri
#define SERVER_CONNECTION_SUCCESS 0
#define SERVER_CONNECTION_UNREACHEABLE 1
#define SERVER_CONNECTION_REFUSED 2

// Corect (v.2) - Folositi enumeratii sau constante
class ServerConstants {
 public:
  static const int CONECTION_SUCCESS = 0;
  static const int CONNECTION_UNREACHEABLE = 1;
  static const int CONNECTION_REFUSED = 2;
};
```

2.8 Cod aerisit - 1 punct

Pentru a facilita lizibilitatea codului, folositi spatiere verticala intre blocuri logic independente.