Обектно-ориентирано програмиране

25.03.2010

Въпрос

Въпрос

Какъв е основния проблем в разработването на софтуер?

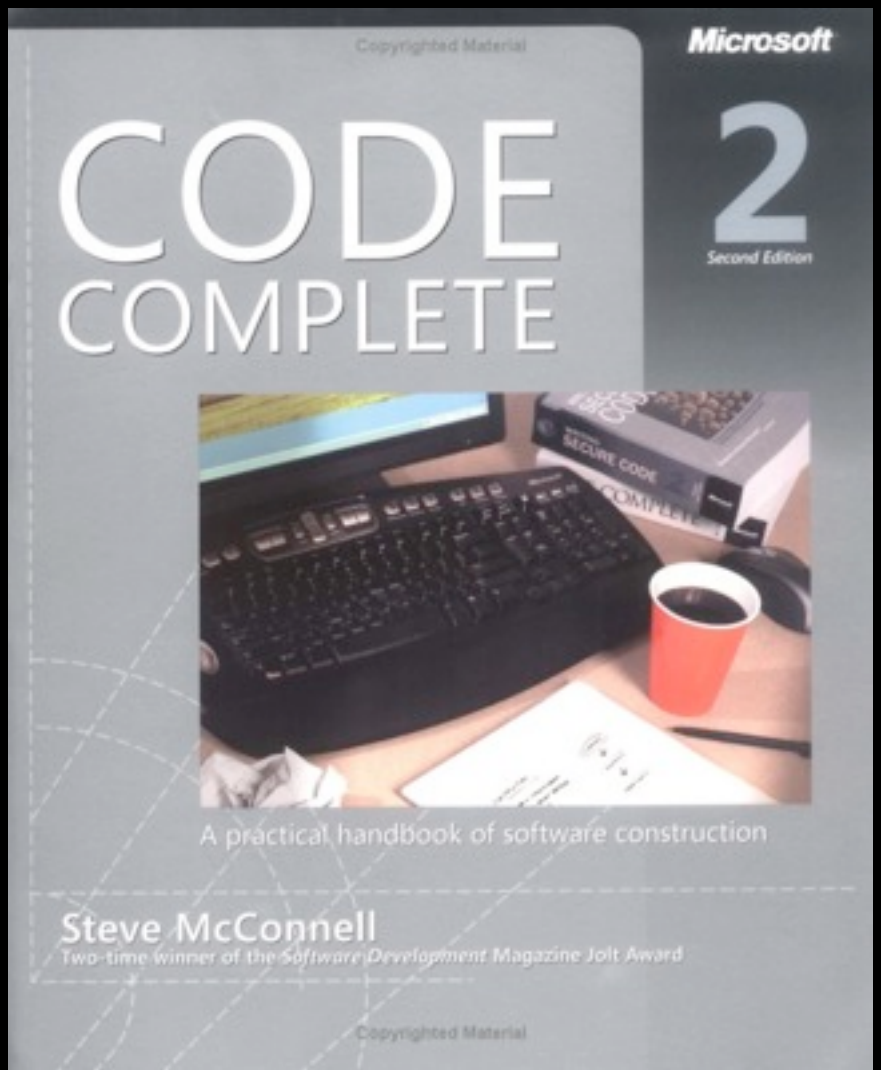
Отговор

Отговор

Сложност

Software Development's primary imperative:

Manage Complexity



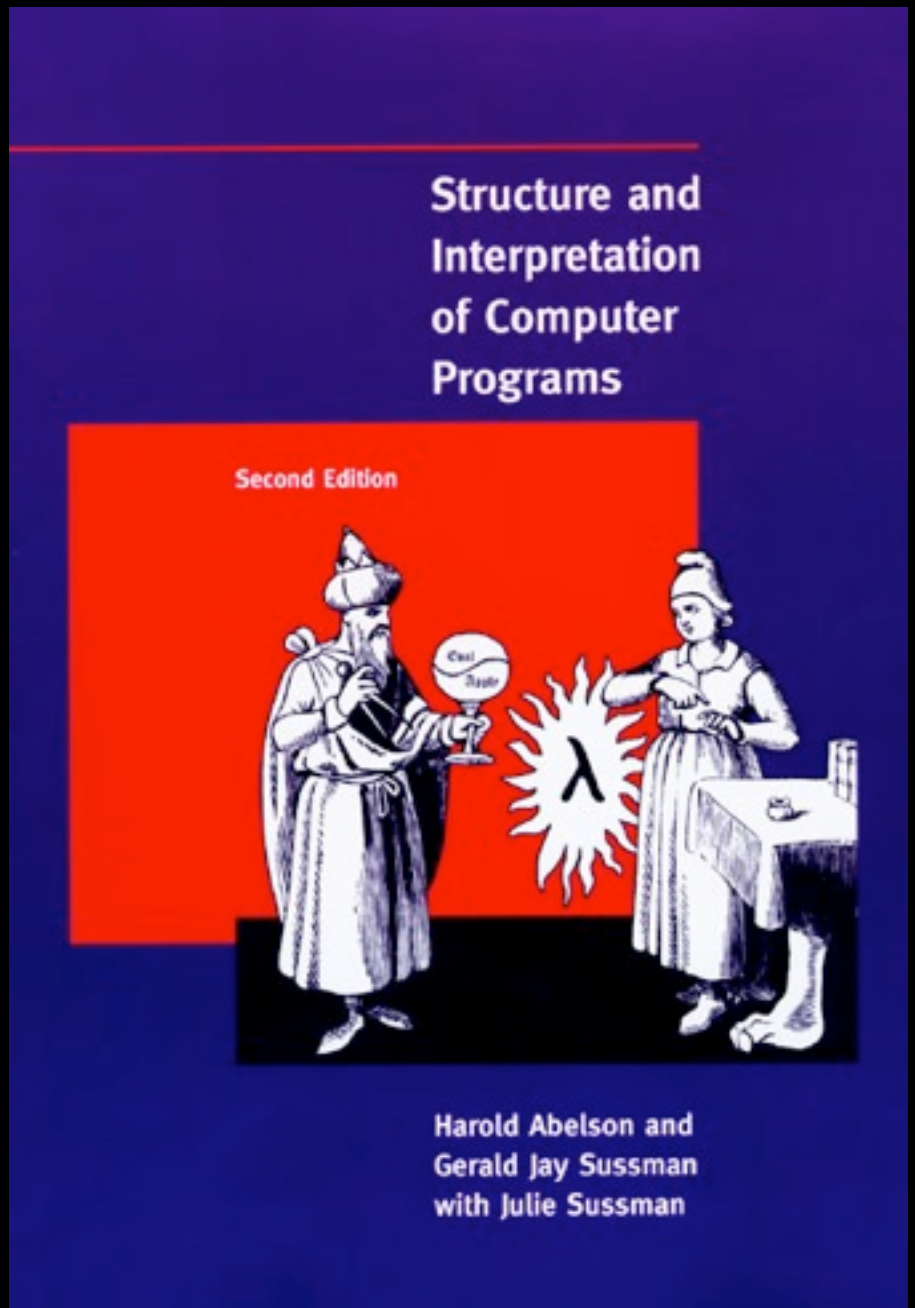
четимост
&
разбираемост

Какво правят следните програми?

```
_l1:
    mov eax,4
    mov ebx,1
    mov ecx,arr
    add ecx,esi
    mov edx,1
    int 80h

    inc esi
    cmp esi,12
    jl _l1
```

```
for (int i = 0; i < 12; i++) {
    printf(arr[i]);
}
```

```
for (int i = 0; i < MONTHS_IN_YEAR; i++) {
    printf(monthNames[i]);
}
```



Programs must be written for people to read, and only incidentally for machines to execute.

още един пример

```
float* something(float *a, float *b, float *c) {
    float v1[3], v2[3];
    float *pr = new float[3];

    for (int i = 0; i < 3; i++) {
        v1[i] = b[i] - a[i];
        v2[i] = c[i] - b[i];
    }

    pr[0] = v1[1] * v2[2] - v1[2] * v2[1];
    pr[1] = v1[2] * v2[0] - v1[0] * v2[1];
    pr[2] = v2[0] * v2[1] - v1[1] * v2[0];

    float len = sqrt(pr[0] * pr[0] + pr[1] * pr[1] + pr[2] * pr[2]);

    for (int i = 0; i < 3; i++) pr[i] /= len;

    return pr;
}
```

```
typedef float point[3];
typedef float vector[3];

vector *make_vector(point *a, point *b) {
    vector *result = new vector;
    for (int i = 0; i < 3; i++) result[i] = b[i] - a[i];
    return result;
}

vector *cross_product(vector *a, vector *b) {
    vector *result = new vector;
    result[0] = a[1] * b[2] - a[2] * b[1];
    result[1] = a[2] * b[0] - a[0] * b[2];
    result[2] = a[0] * b[1] - a[1] * b[0];
    return result;
}

void normalize_vector(vector *v) {
    float len = sqrt(v[0] * v[0] + v[1] * v[1] + v[2] * v[2]);
    for (int i = 0; i < 3; i++) vector[i] /= len;
}
```

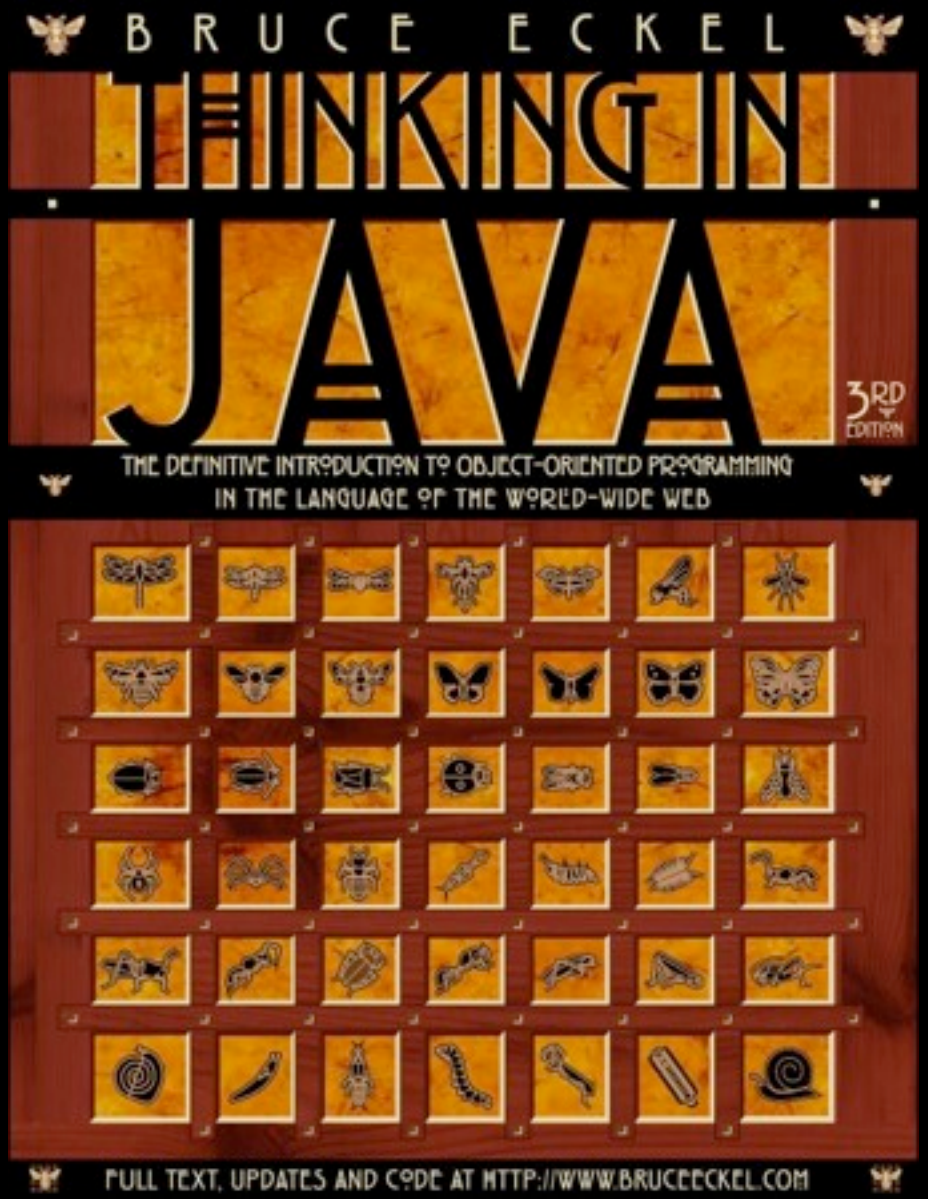
```
vector *something(point *a, point *b, point *c) {
    vector* v1 = make_vector(a, b);
    vector* v2 = make_vector(b, c);
    vector* result = cross_product(v1, v2);

    normalize_vector(result);

    delete[] v1;
    delete[] v2;

    return result;
}
```

обекти
&
съобщения

Абстракция

Енкапсулация

Модулярност



Абстракция

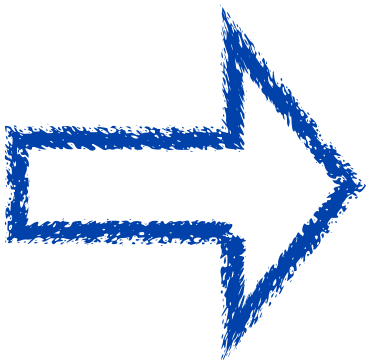
Абстракция

Абстракцията е процесът или резултатът на генерализация чрез намаляването на информационното съдържание на концепция или наблюдаемо явление, обикновено за да се остави само информацията, която е значима за конкретна цел.

Абстракция

Абстракцията е процесът или резултатът на генерализация чрез намаляването на информационното съдържание на концепция или наблюдаемо явление, обикновено за да се остави само информацията, която е значима за конкретна цел.

Specific house

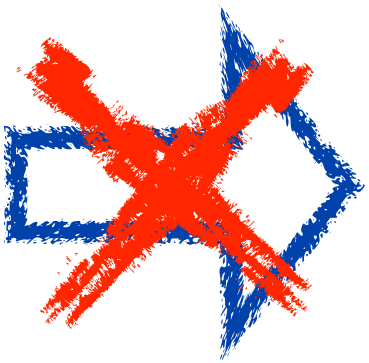


Abstract house

енкапсулация

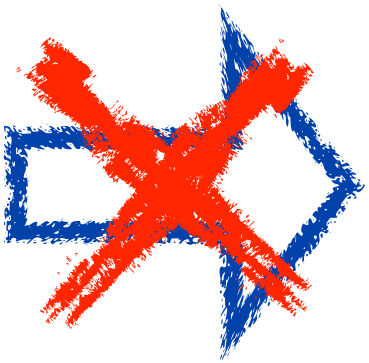
(== яицкартсба)

Abstract house



Doctor House

Abstract house



Specific house

модуляризация

```
vector *something(point *a, point *b, point *c) {
    vector* v1 = make_vector(a, b);
    vector* v2 = make_vector(b, c);
    vector* result = cross_product(v1, v2);

    normalize_vector(result);

    delete[] v1;
    delete[] v2;

    return result;
}
```

```
vector something(point a, point b, point c) {
  vector v1 = new vector(a, b);
  vector v2 = new vector(b, c);

  return v1.product(v2).normalized();
}
```