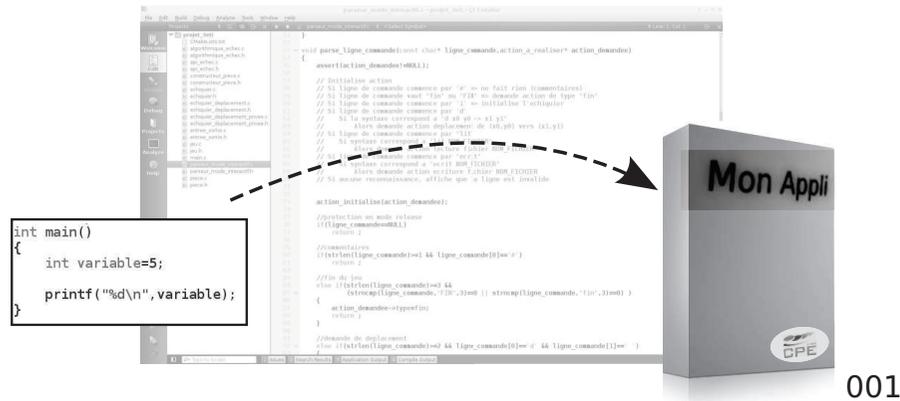


Developpement logiciel en C

Software development in C



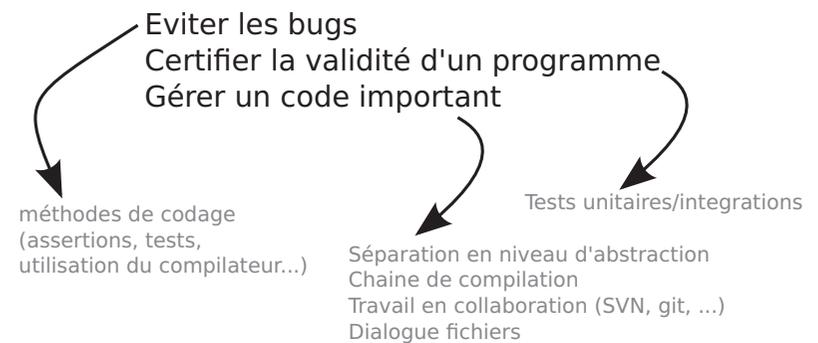
```
int main()
{
    int variable=5;
    printf("%d\n",variable);
}
```

001

But du module

Connaitre/Mettre en oeuvre:

Bonnes pratiques de codage



003

Developpement logiciel en C

- 6 Cours (2h) damien.rohmer@cpe.fr
- 6 séances projets (4h) damien.rohmer@cpe.fr
martine.breda@cpe.fr
serge.mazauric@cpe.fr
anthony.chomienne@cpe.fr
bruno.mascret@liris.cnrs.fr
richard.malgat@inria.fr
- 6 travaux autonomie (3h)
+
18h révision
(72h)

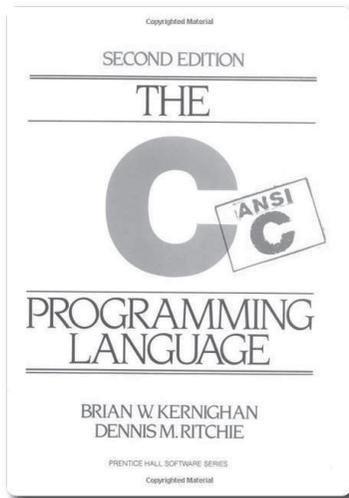
002

Acquis du module

- Acquérir une culture générale informatique (*implicite*)
 - langage du monde informatique
 - licences logiciels
 - cycles de développement
 - logiciels de debugs, version
 - ...
- Devenir indépendant face à la machine (*implicite*)
 - utiliser le compilateur et lire ses sorties debugger
 - utiliser la ligne de commande sous linux
 - réaliser des scripts minimalistes d'automatisation
 - utiliser les Makefile
 - utiliser des librairies
 - ...
- Savoir lire et s'habituer au code de vrais projets (*explicite*)
 - precompilateurs
 - contraintes des gros codes
 - méthodologie standard de développement
 - connaitre et reconnaître les bonnes pratiques
 - ...

004

Bibliographie supplémentaire



Syntaxe C complète

005

Bibliographie supplémentaire

Les sites web:

Wikipedia
<http://www.wikipedia.org/>



+ Google France

Developpez.com
<http://www.developpez.com/>



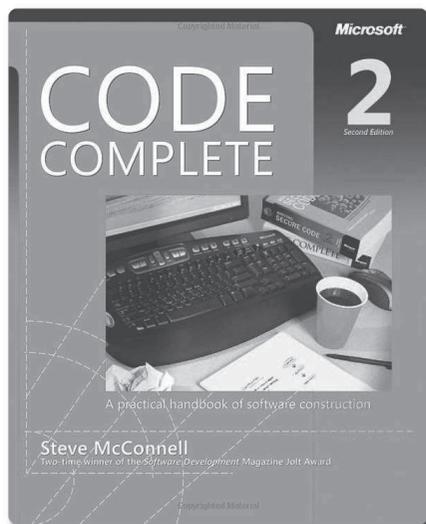
OpenClassrooms
<http://fr.openclassrooms.com>



Servez-vous en pour pratiquer, progresser
Ne le cachez pas!

007

Bibliographie supplémentaire



Bonnes pratiques de codage (générique)

006

Pratiquez

Informatique = pratique
= expérience

Pratiquez !

+ Pratique => meilleur emploi

recherchez ce que vous savez faire!
pas vos notes!



008

Pratiquez

Informatique= pratique
= expérience

Pratiquez !

Outils à votre disposition:

PC Personnel+  = confort

Pensez aux solutions pas chères
ex. leboncoin:



120 euros



180 euros



85 euros

+
Salles infos CPE
ouvertes jusqu'à
21h !

009

Qualité du code

Qu'est ce qu'un bon code?

011

Rappel: Types de bases (*built-in*)

```
printf("char      (%d)\n",sizeof(char));  
printf("short    (%d)\n",sizeof(short));  
printf("int       (%d)\n",sizeof(int));  
printf("long int  (%d)\n",sizeof(long int));  
printf("long long int (%d)\n",sizeof(long long int));  
printf("float     (%d)\n",sizeof(float));  
printf("double    (%d)\n",sizeof(double));  
printf("long double (%d)\n",sizeof(long double));  
printf("void*     (%d)\n",sizeof(void*));
```

char	(1)
short	(2)
int	(4)
long int	(8)
long long int	(8)

entiers signés

unsigned char
unsigned int
unsigned short
unsigned long int
unsigned long long int

entiers positifs

float	(4)
double	(8)
long double	(16)

flottants

char*	(8)
int*	(8)
...	
void*	(8)

pointeurs

Sous Linux, compilé avec gcc, x86, 64bits

010

Qualité du code

Qu'est ce qu'un bon code?

un code en un minimum de lignes ?
un code avec un maximum d'optimisations ?
un code qui n'utilise que des pointeurs ?

un code qui se lit simplement ?
un code qui s'écrit simplement ?

un code qui montre sa complexité ?
un code qui cache sa complexité ?

012

Qualité du code

Un code lisible est un code:

Réutilisable

- Qui se lit et se comprend facilement
- Qui est sa propre documentation
- Représenter le plus possible des entités réelles
- S'organiser en structures représentatives



Peu de bugs

Maintenable

Facile à débiter

Optimisation plus aisée

013

Qualité du code



Remarque 1

Utiliser les **enum** dès que possible

- + Expressivité d'une chaîne de caractère
- + Simplicité d'un entier

```
enum type_animal {poule,lapin,cheval,chevre};
enum type_nourriture {mais,carotte,foin,herbe};

int main()
{
    enum type_animal mon_animal=lapin;
    enum type_nourriture nourriture;
    nourriture=foin;

    return 0;
}
```

Diagram annotations:

- nom de l'enum: points to 'type_animal' and 'type_nourriture' in the enum declarations.
- valeurs possibles: points to 'poule,lapin,cheval,chevre' and 'mais,carotte,foin,herbe'.
- automatique: points to the enum declarations.
- affectation, manipulation comme un entier: points to the assignments in the main function.

Legend:

- poule : 0
- lapin : 1
- cheval : 2
- chevre : 3

015

Qualité du code

Cas des enum

014

Enum

Exemple

```
enum type_animal {poule,lapin,cheval,chevre};
enum type_nourriture {mais,carotte,foin,herbe};

enum type_nourriture trouve_nourriture(enum type_animal animal)
{
    if(animal==poule)
        return mais;
    if(animal==lapin)
        return carotte;
    if(animal==cheval)
        return foin;
    if(animal==chevre)
        return herbe;
}

int main()
{
    enum type_animal mon_animal=lapin;
    enum type_nourriture nourriture=trouve_nourriture(mon_animal);

    printf("Nourriture a acheter: %d\n",nourriture);

    return 0;
}
```

016

Enum

Exemple, encore mieux

```
enum type_animal {poule,lapin,cheval,chevre};
enum type_nourriture {mais,carotte,foin,herbe};

enum type_nourriture trouve_nourriture(enum type_animal animal)
{
    switch(animal)
    {
        case poule:
            return mais;
        case lapin:
            return carotte;
        case cheval:
            return foin;
        case chevre:
            return herbe;
        default:
            puts("Erreur nourriture");
            abort();
    }
}

int main()
{
    enum type_animal mon_animal=lapin;
    enum type_nourriture nourriture=trouve_nourriture(mon_animal);

    printf("Nourriture a acheter: %d\n",nourriture);

    return 0;
}
```

switch/case

Gestion d'erreur

017

Enum

Avec des chaînes de caractères

```
enum type_animal {poule,lapin,cheval,chevre};
enum type_nourriture {mais,carotte,foin,herbe};
enum type_nourriture trouve_nourriture(enum type_animal animal)
{
    switch(animal)
    {
        case poule:
            return mais;
        case lapin:
            return carotte;
        case cheval:
            return foin;
        case chevre:
            return herbe;
        default:
            puts("Erreur nourriture");
            abort();
    }
}

int main()
{
    enum type_animal mon_animal=lapin;
    enum type_nourriture nourriture=trouve_nourriture(mon_animal);

    printf("Nourriture a acheter: %d\n",nourriture);

    return 0;
}
```

switch/case impossible

Fonctions complexe (strcmp, strcpy,...) Pointeurs sous jacents Lourd => A éviter

NE PAS FAIRE

019

Enum

Avec des entiers

```
//On dit que
//poule=0, lapin=1, cheval=2, chevre=3
//mais=0, carotte=1, herbe=2, foin=3
int trouve_nourriture(int mon_animal)
{
    switch(animal)
    {
        case 0:
            return 0;
        case 1:
            return 1;
        case 2:
            return 3;
        case 3:
            return 2;
    }
}

int main()
{
    int mon_animal=2;
    int nourriture=trouve_nourriture(mon_animal);

    printf("Nourriture a acheter: %d\n",nourriture);

    return 0;
}
```

Peu compréhensible (nécessite commentaires) Erreurs faciles! => A éviter !

NE PAS FAIRE

018

Enum

Fonction de conversion enum -> chaîne de caractères

```
enum type_animal {poule,lapin,cheval,chevre};

const char* nomme_animal(enum type_animal animal)
{
    switch(animal)
    {
        case poule:
            return "poule";
        case lapin:
            return "lapin";
        case cheval:
            return "cheval";
        case chevre:
            return "chevre";
        default:
            return "inconnu";
    }
}

int main()
{
    printf("Je suis une %s\n",nomme_animal(poule));
    return 0;
}
```

Simplicité enum + Expressivité chaîne

A FAIRE!

020

Enum

Les enum peuvent indexer des tableaux

```
enum nom_pays {France,
               Allemagne,
               Angleterre,
               Espagne,
               Chine};

int main()
{
    int population[6];

    population[France]=66;
    population[Allemagne]=82;
    population[Angleterre]=53;
    population[Espagne]=47;
    population[Chine]=1351;

    return 0;
}
```

Simple
Expressif



021

Cas des structs

Utilisez dès que possible
les structs pour structurer votre code



```
struct employe
{
    char nom[20];
    int age;
    int salaire;
    int anciennete;
};

struct entreprise
{
    char nom[20];
    long int chiffre_affaire;
    int benefice;
}
```

segmente en entités

représente un
objet/personnage réel

struct = 1 niveau
d'abstraction

023

Qualité du code

Cas des structs

022

Struct

```
struct ma_structure
{
    int variable_1;
    int variable_2;
    float variable_3;
    int variable_4;
};

int main()
{
    struct ma_structure a;
    a.variable_1=5;
    a.variable_3=12.45;

    struct ma_structure b;
    b.variable_4=8;

    printf("%f\n", a.variable_3);

    return 0;
}
```

rappel du mot clé struct

024

Struct

```
enum couleur_carrosserie {rouge,vert,bleu,jaune,violet,noir,blanc};

struct roue_voiture
{
    int prix;
    float pression;
    char nom_constructeur[16];
};

struct carrosserie_voiture
{
    int prix;
    enum couleur_carrosserie couleur;
};

struct vitre_voiture
{
    int prix;
};

struct voiture
{
    struct roue_voiture roue[4];
    struct carrosserie_voiture carrosserie;
    struct vitre_voiture vitre[6];
};
```

enumeration

025

Struct (adresse élément)

```
roue_affecte_michelin(struct roue_voiture* roue)
{
    strcpy(roue->nom_constructeur,"Michelin");
    roue->pression=2.1;
    prix=165;
}

int main()
{
    struct voiture ma_bmw;

    //affectation de la roue 2:
    roue_affecte_michelin( &(ma_bmw.roue[2]) );

    return 0;
}
```

adresse de la roue

027

Struct multiples

```
enum couleur_carrosserie {rouge,vert,bleu,jaune,violet,noir,blanc};

struct roue_voiture
{
    int prix;
    float pression;
    char nom_constructeur[16];
};

struct carrosserie_voiture
{
    int prix;
    enum couleur_carrosserie couleur;
};

struct vitre_voiture
{
    int prix;
};

struct voiture
{
    struct roue_voiture roue[4];
    struct carrosserie_voiture carrosserie;
    struct vitre_voiture vitre[6];
};

int main()
{
    struct voiture ma_bmw;

    //prix de la carrosserie
    ma_bmw.carrosserie=8500;

    //prix de la roue 2
    ma_bmw.roue[2].prix=95;

    //prix de la vitre 0
    ma_bmw.vitre[0].prix=150;

    //couleur de la carrosserie
    ma_bmw.carrosserie.couleur=noir;

    return 0;
}
```

026

Struct anonyme

```
typedef struct
{
    float prix;
    float quantite;
}stock_pomme;

int main()
{
    stock_pomme mon_stock;
    mon_stock.prix=1.14;
    mon_stock.quantite=3.14;

    return 0;
}
```

permet d'éviter la répétition du mot struct

028

Qualité du code

Cas des tableaux

029

Syntaxe pointeur/tableau

Tableau

```
int main()
{
    int tableau[TAILLE];

    //acces a la case indice 5
    int a=tableau[5];
    //affectation sur la premiere case
    tableau[0]=7;
}
```

Pointeur

```
int main()
{
    int* pointeur=NULL;
    int a=4;

    //pointe sur l'adresse de a
    pointeur=&a;

    //valeur du pointeur
    int b=*pointeur;

    //affectation de la valeur du pointeur
    *pointeur=8;
}
```

031

Cas des tableaux

Ne jamais utiliser l'arithmétique pointeur pour indexer les tableaux

```
int T[5];
T[2]=8;
```

Lisible

```
int T[5];
*(T+2)=8;
```

Peu lisible

```
int T[5][8];
T[2][3]=8;
```

Lisible

```
int T[5][8];
*(*(T+2)+3)=8;
```

Illisible

**(p+k) = Exercice scolaire != Développement logiciel*

030

Syntaxe pointeur/tableau

Tableau

T[k]

Pointeur

*p

En C, tableau et pointeurs sont confondus

Uniquement en C!

A NE PAS FAIRE:

```
int main()
{
    int tableau[TAILLE];
    int *p=tableau;

    int a=*(p+5);
    *(tableau+0)=7;
}
```

tableau avec
syntaxe pointeur

```
int main()
{
    int* pointeur=NULL;
    int a=4;
    pointeur=&a;
    int b=pointeur[0];
    pointeur[0]=8;
}
```

pointeur avec
syntaxe tableau

032

Syntaxe pointeur/tableau

Bonne pratique:

Interdisez vous: `*(pointeur+k)` ✗

Autodocumentation: Tableau : `T[k]`
Pointeur : `*p` ✓

Banissez: pointeurs multiples (`int **p`) ✗

structurez en abstraction ✓

033

Syntaxe pointeur/tableau

Ex. Listing de notes

Mathieu: {12,13,14,11,09,12}
Francois: {5,7,11,11,10,8}
Florence: {15,15,16,12,11,18}

```
int main()
{
    int T[3][6]={{12,13,14,11,9,12},
                {5,7,11,11,10,8},
                {15,15,16,12,11,18}};

    int **p=T;

    //acces a la 4eme note de mathieu:
    int note=*(p+3);
    //écriture de la 3eme note de Francois avec un 8
    *(*p+1)+2=8;
}
```

Ecriture pointeur
A ne pas faire

```
int main()
{
    int T[3][6]={{12,13,14,11,9,12},
                {5,7,11,11,10,8},
                {15,15,16,12,11,18}};

    //acces a la 4eme note de mathieu:
    int note=T[0][3];
    //écriture de la 3eme note de Francois avec un 8
    T[1][2]=8;
}
```

Ecriture tableau
OK, avec documentation

035

Syntaxe pointeur/tableau

Exemples de portabilité (différents langages):

Vecteur en C++

<pre>int main() { vector<int> mon_vecteur; mon_vecteur.resize(50); mon_vecteur[10]=5; }</pre>	<pre>int main() { vector<int> mon_vecteur; mon_vecteur.resize(50); *(mon_vecteur+10)=5; }</pre>
---	---

Tableau en Python

<pre>tableau = [4,5,6,7]; tableau[2]=8;</pre>	<pre>tableau = [4,5,6,7]; *(tableau+2)=8;</pre>
---	---

Tableau en Java

<pre>public class Programme { public static void main(String[] args) { int tableau[] = new int [10]; tableau[2]=8; } }</pre>	<pre>public class Programme { public static void main(String[] args) { int tableau[] = new int [10]; *(tableau+2)=8; } }</pre>
--	--

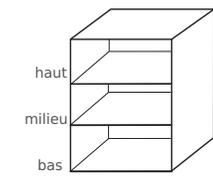
034

Syntaxe pointeur/tableau



Ex. Etagère de CD/DVD/Livres

rock blues classique techno
action horreur sci-fi thriller
fantaisie biographie roman cuisine



=> Enregistrement du nom de l'auteur/chanteur

Formalisme pointeur ✗
(proche du code)

```
int main()
{
    char etagere[3][4][3][10][50];
    char *****p;

    //acces: chanteur du 4eme CD de rock sur l'etage du bas
    const char* chanteur=***(p+1)+3;

    //modification: auteur du 8eme Livre de cuisine sur l'etage du haut
    strcpy( *(p+1)(p+2)+2)+7, "Maite et Micheline");
}
```

court
lisible?
debug-able?
correct?

Formalisme bibliotheque ✓✓
(proche de l'objet reel)

```
#define NOMBRE_TYPE_OBJET 3
#define NOMBRE_CATEGORIE 4
#define NOMBRE_ETAGE 3
#define NOMBRE_MAX_OBJET 10
#define TAILLE_MAX_NON_AUTEUR 50

enum type_objet{DVD,CD,Livre};
enum type_etage{bas,milieu,haut};
enum type_CD {rock,blues,classique,techno};
enum type_DVD {action,horreur,scifi,triller};
enum type_CD_Livre {livre,cuisine};

int main()
{
    char etagere[NOMBRE_TYPE_OBJET]
                [NOMBRE_CATEGORIE]
                [NOMBRE_ETAGE]
                [NOMBRE_MAX_OBJET]
                [TAILLE_MAX_NON_AUTEUR];

    //acces: chanteur du 4eme CD de rock sur l'etage du bas
    const char* chanteur=etagere[CD][rock][bas][3];

    //modification: auteur du 8eme Livre de cuisine sur l'etage du haut
    strcpy(etagere[Livre][cuisine][haut][7], "Maite et Micheline");
}
```

commentaire
quasi-inutile

036

Qualité du code

Remarques sur l'optimisation

037

Quel est le code le plus rapide? le plus lisible?

```
void init(int v[],int N)
{
  unsigned int k=0;
  for (k=0;k<N; ++k)
    v[k]=k;
}

void fonction(int v1[],int v2[],int v3[],int N);

int main()
{
  int N=16;
  int v1[N]; int v2[N]; int v3[N];

  init(v1,N); init(v2,N);  init(v3,N);

  fonction(v1,v2,v3,N);
  return 0;
}
```

039

+ Court != + Lisible

Exemple

Code A:

```
int main()
{
  int v1[]={1,2,3,-1},v2[]={4,5,6,-1};
  int *p1=v1,*p2=v2;
  while(*(p1++)!=-1) *(p1-1)+=*(p2++);
}
```

commentaires
indispensables
bug?

Code B:

```
int calcul_longueur(int vecteur[])
{
  int taille_entier=sizeof(int);
  int longueur=sizeof(vecteur)/taille_entier;
  return longueur;
}

int main()
{
  int v1[]={1,2,3};
  int v2[]={4,5,6};

  int longueur_vecteur=calcul_longueur(v1);

  int resultat[longueur_vecteur];
  int k=0;
  for(k=0;k<longueur_vecteur;k++)
  {
    resultat[k] = v1[k] + v2[k];
  }
}
```

vraiment besoin
de commentaires?

038

Quel est le code le plus rapide? le plus lisible?

1

```
void fonction(int v1[],int v2[],int v3[],int N)
{
  unsigned int k=0;
  for(k=0;k<N; ++k)
  {
    v3[k]=v1[k]+v2[k];
  }
}
```

2

```
void fonction(int v1[],int v2[],int v3[],int N)
{
  int *p_v1=v1,*p_v2=v2,*p_v3=v3;
  int *p_v1_end=p_v1+N;
  int *p_v2_end=p_v2+N;
  register unsigned int k=1;
  while(k<N)
  {
    *p_v3++ = *p_v1++ + *p_v2++;
    (*p_v1, *p_v2, *p_v3)++;
    ++k;
  }
}
```

3

```
void fonction(int v1[],int v2[],int v3[],int N)
{
  v1[1]=v3;
  v2[1]=v1[0]+v2[2];v3[1]=v1[0]+v2[2];
  v1[2]=v3;
  v3[2]=v1[1]+v2[3];v3[2]=v1[1]+v2[3];
  v1[3]=v3;
  v3[3]=v1[2]+v2[4];v3[3]=v1[2]+v2[4];
  v1[4]=v3;
  v3[4]=v1[3]+v2[5];v3[4]=v1[3]+v2[5];
  v1[5]=v3;
  v3[5]=v1[4]+v2[6];v3[5]=v1[4]+v2[6];
  v1[6]=v3;
  v3[6]=v1[5]+v2[7];v3[6]=v1[5]+v2[7];
  v1[7]=v3;
  v3[7]=v1[6]+v2[8];v3[7]=v1[6]+v2[8];
  v1[8]=v3;
  v3[8]=v1[7]+v2[9];v3[8]=v1[7]+v2[9];
  v1[9]=v3;
  v3[9]=v1[8]+v2[10];v3[9]=v1[8]+v2[10];
  v1[10]=v3;
  v3[10]=v1[9]+v2[11];v3[10]=v1[9]+v2[11];
  v1[11]=v3;
  v3[11]=v1[10]+v2[12];v3[11]=v1[10]+v2[12];
  v1[12]=v3;
  v3[12]=v1[11]+v2[13];v3[12]=v1[11]+v2[13];
  v1[13]=v3;
  v3[13]=v1[12]+v2[14];v3[13]=v1[12]+v2[14];
  v1[14]=v3;
  v3[14]=v1[13]+v2[15];v3[14]=v1[13]+v2[15];
  v1[15]=v3;
  v3[15]=v1[14]+v2[16];v3[15]=v1[14]+v2[16];
  v1[16]=v3;
  v3[16]=v1[15]+v2[17];v3[16]=v1[15]+v2[17];
  v1[17]=v3;
  v3[17]=v1[16]+v2[18];v3[17]=v1[16]+v2[18];
  v1[18]=v3;
  v3[18]=v1[17]+v2[19];v3[18]=v1[17]+v2[19];
  v1[19]=v3;
  v3[19]=v1[18]+v2[20];v3[19]=v1[18]+v2[20];
  v1[20]=v3;
  v3[20]=v1[19]+v2[21];v3[20]=v1[19]+v2[21];
  v1[21]=v3;
  v3[21]=v1[20]+v2[22];v3[21]=v1[20]+v2[22];
  v1[22]=v3;
  v3[22]=v1[21]+v2[23];v3[22]=v1[21]+v2[23];
  v1[23]=v3;
  v3[23]=v1[22]+v2[24];v3[23]=v1[22]+v2[24];
  v1[24]=v3;
  v3[24]=v1[23]+v2[25];v3[24]=v1[23]+v2[25];
  v1[25]=v3;
  v3[25]=v1[24]+v2[26];v3[25]=v1[24]+v2[26];
  v1[26]=v3;
  v3[26]=v1[25]+v2[27];v3[26]=v1[25]+v2[27];
  v1[27]=v3;
  v3[27]=v1[26]+v2[28];v3[27]=v1[26]+v2[28];
  v1[28]=v3;
  v3[28]=v1[27]+v2[29];v3[28]=v1[27]+v2[29];
  v1[29]=v3;
  v3[29]=v1[28]+v2[30];v3[29]=v1[28]+v2[30];
  v1[30]=v3;
  v3[30]=v1[29]+v2[31];v3[30]=v1[29]+v2[31];
  v1[31]=v3;
  v3[31]=v1[30]+v2[32];v3[31]=v1[30]+v2[32];
  v1[32]=v3;
  v3[32]=v1[31]+v2[33];v3[32]=v1[31]+v2[33];
  v1[33]=v3;
  v3[33]=v1[32]+v2[34];v3[33]=v1[32]+v2[34];
  v1[34]=v3;
  v3[34]=v1[33]+v2[35];v3[34]=v1[33]+v2[35];
  v1[35]=v3;
  v3[35]=v1[34]+v2[36];v3[35]=v1[34]+v2[36];
  v1[36]=v3;
  v3[36]=v1[35]+v2[37];v3[36]=v1[35]+v2[37];
  v1[37]=v3;
  v3[37]=v1[36]+v2[38];v3[37]=v1[36]+v2[38];
  v1[38]=v3;
  v3[38]=v1[37]+v2[39];v3[38]=v1[37]+v2[39];
  v1[39]=v3;
  v3[39]=v1[38]+v2[40];v3[39]=v1[38]+v2[40];
  v1[40]=v3;
  v3[40]=v1[39]+v2[41];v3[40]=v1[39]+v2[41];
  v1[41]=v3;
  v3[41]=v1[40]+v2[42];v3[41]=v1[40]+v2[42];
  v1[42]=v3;
  v3[42]=v1[41]+v2[43];v3[42]=v1[41]+v2[43];
  v1[43]=v3;
  v3[43]=v1[42]+v2[44];v3[43]=v1[42]+v2[44];
  v1[44]=v3;
  v3[44]=v1[43]+v2[45];v3[44]=v1[43]+v2[45];
  v1[45]=v3;
  v3[45]=v1[44]+v2[46];v3[45]=v1[44]+v2[46];
  v1[46]=v3;
  v3[46]=v1[45]+v2[47];v3[46]=v1[45]+v2[47];
  v1[47]=v3;
  v3[47]=v1[46]+v2[48];v3[47]=v1[46]+v2[48];
  v1[48]=v3;
  v3[48]=v1[47]+v2[49];v3[48]=v1[47]+v2[49];
  v1[49]=v3;
  v3[49]=v1[48]+v2[50];v3[49]=v1[48]+v2[50];
  v1[50]=v3;
  v3[50]=v1[49]+v2[51];v3[50]=v1[49]+v2[51];
  v1[51]=v3;
  v3[51]=v1[50]+v2[52];v3[51]=v1[50]+v2[52];
  v1[52]=v3;
  v3[52]=v1[51]+v2[53];v3[52]=v1[51]+v2[53];
  v1[53]=v3;
  v3[53]=v1[52]+v2[54];v3[53]=v1[52]+v2[54];
  v1[54]=v3;
  v3[54]=v1[53]+v2[55];v3[54]=v1[53]+v2[55];
  v1[55]=v3;
  v3[55]=v1[54]+v2[56];v3[55]=v1[54]+v2[56];
  v1[56]=v3;
  v3[56]=v1[55]+v2[57];v3[56]=v1[55]+v2[57];
  v1[57]=v3;
  v3[57]=v1[56]+v2[58];v3[57]=v1[56]+v2[58];
  v1[58]=v3;
  v3[58]=v1[57]+v2[59];v3[58]=v1[57]+v2[59];
  v1[59]=v3;
  v3[59]=v1[58]+v2[60];v3[59]=v1[58]+v2[60];
  v1[60]=v3;
  v3[60]=v1[59]+v2[61];v3[60]=v1[59]+v2[61];
  v1[61]=v3;
  v3[61]=v1[60]+v2[62];v3[61]=v1[60]+v2[62];
  v1[62]=v3;
  v3[62]=v1[61]+v2[63];v3[62]=v1[61]+v2[63];
  v1[63]=v3;
  v3[63]=v1[62]+v2[64];v3[63]=v1[62]+v2[64];
  v1[64]=v3;
  v3[64]=v1[63]+v2[65];v3[64]=v1[63]+v2[65];
  v1[65]=v3;
  v3[65]=v1[64]+v2[66];v3[65]=v1[64]+v2[66];
  v1[66]=v3;
  v3[66]=v1[65]+v2[67];v3[66]=v1[65]+v2[67];
  v1[67]=v3;
  v3[67]=v1[66]+v2[68];v3[67]=v1[66]+v2[68];
  v1[68]=v3;
  v3[68]=v1[67]+v2[69];v3[68]=v1[67]+v2[69];
  v1[69]=v3;
  v3[69]=v1[68]+v2[70];v3[69]=v1[68]+v2[70];
  v1[70]=v3;
  v3[70]=v1[69]+v2[71];v3[70]=v1[69]+v2[71];
  v1[71]=v3;
  v3[71]=v1[70]+v2[72];v3[71]=v1[70]+v2[72];
  v1[72]=v3;
  v3[72]=v1[71]+v2[73];v3[72]=v1[71]+v2[73];
  v1[73]=v3;
  v3[73]=v1[72]+v2[74];v3[73]=v1[72]+v2[74];
  v1[74]=v3;
  v3[74]=v1[73]+v2[75];v3[74]=v1[73]+v2[75];
  v1[75]=v3;
  v3[75]=v1[74]+v2[76];v3[75]=v1[74]+v2[76];
  v1[76]=v3;
  v3[76]=v1[75]+v2[77];v3[76]=v1[75]+v2[77];
  v1[77]=v3;
  v3[77]=v1[76]+v2[78];v3[77]=v1[76]+v2[78];
  v1[78]=v3;
  v3[78]=v1[77]+v2[79];v3[78]=v1[77]+v2[79];
  v1[79]=v3;
  v3[79]=v1[78]+v2[80];v3[79]=v1[78]+v2[80];
  v1[80]=v3;
  v3[80]=v1[79]+v2[81];v3[80]=v1[79]+v2[81];
  v1[81]=v3;
  v3[81]=v1[80]+v2[82];v3[81]=v1[80]+v2[82];
  v1[82]=v3;
  v3[82]=v1[81]+v2[83];v3[82]=v1[81]+v2[83];
  v1[83]=v3;
  v3[83]=v1[82]+v2[84];v3[83]=v1[82]+v2[84];
  v1[84]=v3;
  v3[84]=v1[83]+v2[85];v3[84]=v1[83]+v2[85];
  v1[85]=v3;
  v3[85]=v1[84]+v2[86];v3[85]=v1[84]+v2[86];
  v1[86]=v3;
  v3[86]=v1[85]+v2[87];v3[86]=v1[85]+v2[87];
  v1[87]=v3;
  v3[87]=v1[86]+v2[88];v3[87]=v1[86]+v2[88];
  v1[88]=v3;
  v3[88]=v1[87]+v2[89];v3[88]=v1[87]+v2[89];
  v1[89]=v3;
  v3[89]=v1[88]+v2[90];v3[89]=v1[88]+v2[90];
  v1[90]=v3;
  v3[90]=v1[89]+v2[91];v3[90]=v1[89]+v2[91];
  v1[91]=v3;
  v3[91]=v1[90]+v2[92];v3[91]=v1[90]+v2[92];
  v1[92]=v3;
  v3[92]=v1[91]+v2[93];v3[92]=v1[91]+v2[93];
  v1[93]=v3;
  v3[93]=v1[92]+v2[94];v3[93]=v1[92]+v2[94];
  v1[94]=v3;
  v3[94]=v1[93]+v2[95];v3[94]=v1[93]+v2[95];
  v1[95]=v3;
  v3[95]=v1[94]+v2[96];v3[95]=v1[94]+v2[96];
  v1[96]=v3;
  v3[96]=v1[95]+v2[97];v3[96]=v1[95]+v2[97];
  v1[97]=v3;
  v3[97]=v1[96]+v2[98];v3[97]=v1[96]+v2[98];
  v1[98]=v3;
  v3[98]=v1[97]+v2[99];v3[98]=v1[97]+v2[99];
  v1[99]=v3;
  v3[99]=v1[98]+v2[100];v3[99]=v1[98]+v2[100];
  v1[100]=v3;
  v3[100]=v1[99]+v2[101];v3[100]=v1[99]+v2[101];
  v1[101]=v3;
  v3[101]=v1[100]+v2[102];v3[101]=v1[100]+v2[102];
  v1[102]=v3;
  v3[102]=v1[101]+v2[103];v3[102]=v1[101]+v2[103];
  v1[103]=v3;
  v3[103]=v1[102]+v2[104];v3[103]=v1[102]+v2[104];
  v1[104]=v3;
  v3[104]=v1[103]+v2[105];v3[104]=v1[103]+v2[105];
  v1[105]=v3;
  v3[105]=v1[104]+v2[106];v3[105]=v1[104]+v2[106];
  v1[106]=v3;
  v3[106]=v1[105]+v2[107];v3[106]=v1[105]+v2[107];
  v1[107]=v3;
  v3[107]=v1[106]+v2[108];v3[107]=v1[106]+v2[108];
  v1[108]=v3;
  v3[108]=v1[107]+v2[109];v3[108]=v1[107]+v2[109];
  v1[109]=v3;
  v3[109]=v1[108]+v2[110];v3[109]=v1[108]+v2[110];
  v1[110]=v3;
  v3[110]=v1[109]+v2[111];v3[110]=v1[109]+v2[111];
  v1[111]=v3;
  v3[111]=v1[110]+v2[112];v3[111]=v1[110]+v2[112];
  v1[112]=v3;
  v3[112]=v1[111]+v2[113];v3[112]=v1[111]+v2[113];
  v1[113]=v3;
  v3[113]=v1[112]+v2[114];v3[113]=v1[112]+v2[114];
  v1[114]=v3;
  v3[114]=v1[113]+v2[115];v3[114]=v1[113]+v2[115];
  v1[115]=v3;
  v3[115]=v1[114]+v2[116];v3[115]=v1[114]+v2[116];
  v1[116]=v3;
  v3[116]=v1[115]+v2[117];v3[116]=v1[115]+v2[117];
  v1[117]=v3;
  v3[117]=v1[116]+v2[118];v3[117]=v1[116]+v2[118];
  v1[118]=v3;
  v3[118]=v1[117]+v2[119];v3[118]=v1[117]+v2[119];
  v1[119]=v3;
  v3[119]=v1[118]+v2[120];v3[119]=v1[118]+v2[120];
  v1[120]=v3;
  v3[120]=v1[119]+v2[121];v3[120]=v1[119]+v2[121];
  v1[121]=v3;
  v3[121]=v1[120]+v2[122];v3[121]=v1[120]+v2[122];
  v1[122]=v3;
  v3[122]=v1[121]+v2[123];v3[122]=v1[121]+v2[123];
  v1[123]=v3;
  v3[123]=v1[122]+v2[124];v3[123]=v1[122]+v2[124];
  v1[124]=v3;
  v3[124]=v1[123]+v2[125];v3[124]=v1[123]+v2[125];
  v1[125]=v3;
  v3[125]=v1[124]+v2[126];v3[125]=v1[124]+v2[126];
  v1[126]=v3;
  v3[126]=v1[125]+v2[127];v3[126]=v1[125]+v2[127];
  v1[127]=v3;
  v3[127]=v1[126]+v2[128];v3[127]=v1[126]+v2[128];
  v1[128]=v3;
  v3[128]=v1[127]+v2[129];v3[128]=v1[127]+v2[129];
  v1[129]=v3;
  v3[129]=v1[128]+v2[130];v3[129]=v1[128]+v2[130];
  v1[130]=v3;
  v3[130]=v1[129]+v2[131];v3[130]=v1[129]+v2[131];
  v1[131]=v3;
  v3[131]=v1[130]+v2[132];v3[131]=v1[130]+v2[132];
  v1[132]=v3;
  v3[132]=v1[131]+v2[133];v3[132]=v1[131]+v2[133];
  v1[133]=v3;
  v3[133]=v1[132]+v2[134];v3[133]=v1[132]+v2[134];
  v1[134]=v3;
  v3[134]=v1[133]+v2[135];v3[134]=v1[133]+v2[135];
  v1[135]=v3;
  v3[135]=v1[134]+v2[136];v3[135]=v1[134]+v2[136];
  v1[136]=v3;
  v3[136]=v1[135]+v2[137];v3[136]=v1[135]+v2[137];
  v1[137]=v3;
  v3[137]=v1[136]+v2[138];v3[137]=v1[136]+v2[138];
  v1[138]=v3;
  v3[138]=v1[137]+v2[139];v3[138]=v1[137]+v2[139];
  v1[139]=v3;
  v3[139]=v1[138]+v2[140];v3[139]=v1[138]+v2[140];
  v1[140]=v3;
  v3[140]=v1[139]+v2[141];v3[140]=v1[139]+v2[141];
  v1[141]=v3;
  v3[141]=v1[140]+v2[142];v3[141]=v1[140]+v2[142];
  v1[142]=v3;
  v3[142]=v1[141]+v2[143];v3[142]=v1[141]+v2[143];
  v1[143]=v3;
  v3[143]=v1[142]+v2[144];v3[143]=v1[142]+v2[144];
  v1[144]=v3;
  v3[144]=v1[143]+v2[145];v3[144]=v1[143]+v2[145];
  v1[145]=v3;
  v3[145]=v1[144]+v2[146];v3[145]=v1[144]+v2[146];
  v1[146]=v3;
  v3[146]=v1[145]+v2[147];v3[146]=v1[145]+v2[147];
  v1[147]=v3;
  v3[147]=v1[146]+v2[148];v3[147]=v1[146]+v2[148];
  v1[148]=v3;
  v3[148]=v1[147]+v2[149];v3[148]=v1[147]+v2[149];
  v1[149]=v3;
  v3[149]=v1[148]+v2[150];v3[149]=v1[148]+v2[150];
  v1[150]=v3;
  v3[150]=v1[149]+v2[151];v3[150]=v1[149]+v2[151];
  v1[151]=v3;
  v3[151]=v1[150]+v2[152];v3[151]=v1[150]+v2[152];
  v1[152]=v3;
  v3[152]=v1[151]+v2[153];v3[152]=v1[151]+v2[153];
  v1[153]=v3;
  v3[153]=v1[152]+v2[154];v3[153]=v1[152]+v2[154];
  v1[154]=v3;
  v3[154]=v1[153]+v2[155];v3[154]=v1[153]+v2[155];
  v1[155]=v3;
  v3[155]=v1[154]+v2[156];v3[155]=v1[154]+v2[156];
  v1[156]=v3;
  v3[156]=v1[155]+v2[157];v3[156]=v1[155]+v2[157];
  v1[157]=v3;
  v3[157]=v1[156]+v2[158];v3[157]=v1[156]+v2[158];
  v1[158]=v3;
  v3[158]=v1[157]+v2[159];v3[158]=v1[157]+v2[159];
  v1[159]=v3;
  v3[159]=v1[158]+v2[160];v3[159]=v1[158]+v2[160];
  v1[160]=v3;
  v3[160]=v1[159]+v2[161];v3[160]=v1[159]+v2[161];
  v1[161]=v3;
  v3[161]=v1[160]+v2[162];v3[161]=v1[160]+v2[162];
  v1[162]=v3;
  v3[162]=v1[161]+v2[163];v3[162]=v1[161]+v2[163];
  v1[163]=v3;
  v3[163]=v1[162]+v2[164];v3[163]=v1[162]+v2[164];
  v1[164]=v3;
  v3[164]=v1[163]+v2[165];v3[164]=v1[163]+v2[165];
  v1[165]=v3;
  v3[165]=v1[164]+v2[166];v3[165]=v1[164]+v2[166];
  v1[166]=v3;
  v3[166]=v1[165]+v2[167];v3[166]=v1[165]+v2[167];
  v1[167]=v3;
  v3[167]=v1[166]+v2[168];v3[167]=v1[166]+v2[168];
  v1[168]=v3;
  v3[168]=v1[167]+v2[169];v3[168]=v1[167]+v2[169];
  v1[169]=v3;
  v3[169]=v1[168]+v2[170];v3[169]=v1[168]+v2[170];
  v1[170]=v3;
  v3[170]=v1[169]+v2[171];v3[170]=v1[169]+v2[171];
  v1[171]=v3;
  v3[171]=v1[170]+v2[172];v3[171]=v1[170]+v2[172];
  v1[172]=v3;
  v3[172]=v1[171]+v2[173];v3[172]=v1[171]+v2[173];
  v1[173]=v3;
  v3[173]=v1[172]+v2[174];v3[173]=v1[172]+v2[174];
  v1[174]=v3;
  v3[174]=v1[173]+v2[175];v3[174]=v1[173]+v2[175];
  v1[175]=v3;
  v3[175]=v1[174]+v2[176];v3[175]=v1[174]+v2[176];
  v1[176]=v3;
  v3[176]=v1[175]+v2[177];v3[176]=v1[175]+v2[177];
  v1[177]=v3;
  v3[177]=v1[176]+v2[178];v3[177]=v1[176]+v2[178];
  v1[178]=v3;
  v3[178]=v1[177]+v2[179];v3[178]=v1[177]+v2[179];
  v1[179]=v3;
  v3[179]=v1[178]+v2[180];v3[179]=v1[178]+v2[180];
  v1[180]=v3;
  v3[180]=v1[179]+v2[181];v3[180]=v1[179]+v2[181];
  v1[181]=v3;
  v3[181]=v1[180]+v2[182];v3[181]=v1[180]+v2[182];
  v1[182]=v3;
  v3[182]=v1[181]+v2[183];v3[182]=v1[181]+v2[183];
  v1[183]=v3;
  v3[183]=v1[182]+v2[184];v3[183]=v1[182]+v2[184];
  v1[184]=v3;
  v3[184]=v1[183]+v2[185];v3[184]=v1[183]+v2[185];
  v1[185]=v3;
  v3[185]=v1[184]+v2[186];v3[185]=v1[184]+v2[186];
  v1[186]=v3;
  v3[186]=v1[185]+v2[187];v3[186]=v1[185]+v2[187];
  v1[187]=v3;
  v3[187]=v1[186]+v2[188];v3[187]=v1[186]+v2[188];
  v1[188]=v3;
  v3[188]=v1[187]+v2[189];v3[188]=v1[187]+v2[189];
  v1[189]=v3;
  v3[189]=v1[188]+v2[190];v3[189]=v1[188]+v2[190];
  v1[190]=v3;
  v3[190]=v1[189]+v2[191];v3[190]=v1[189]+v2[191];
  v1[191]=v3;
  v3[191]=v1[190]+v2[192];v3[191]=v1[190]+v2[192];
  v1[192]=v3;
  v3[192]=v1[191]+v2[193];v3[192]=v1[191]+v2[193];
  v1[193]=v3;
  v3[193]=v1[192]+v2[194];v3[193]=v1[192]+v2[194];
  v1[194]=v3;
  v3[194]=v1[193]+v2[195];v3[194]=v1[193]+v2[195];
  v1[195]=v3;
  v3[195]=v1[194]+v2[196];v3[195]=v1[194]+v2[196];
  v1[196]=v3;
  v3[196]=v1[195]+v2[197];v3[196]=v1[195]+v2[197];
  v1[197]=v3;
  v3[197]=v1[196]+v2[198];v3[197]=v1[196]+v2[198];
  v1[198]=v3;
  v3[198]=v1[197]+v2[199];v3[198]=v1[197]+v2[199];
  v1[199]=v3;
  v3[199]=v1[198]+v2[200];v3[199]=v1[198]+v2[200];
  v1[200]=v3;
  v3[200]=v1[199]+v2[201];v3[200]=v1[199]+v2[201];
  v1[201]=v3;
  v3[201]=v1[200]+v2[202];v3[201]=v1[200]+v2[202];
  v1[202]=v3;
  v3[202]=v1[201]+v2[203];v3[202]=v1[201]+v2[203];
  v1[203]=v3;
  v3[203]=v1[202]+v2[204];v3[203]=v1[202]+v2[204];
  v1[204]=v3;
  v3[204]=v1[203]+v2[205];v3[204]=v1[203]+v2[205];
  v1[205]=v3;
  v3[205]=v1[204]+v2[206];v3[205]=v1[204]+v2[206];
  v1[206]=v3;
  v3[206]=v1[205]+v2[207];v3[206]=v1[205]+v2[207];
  v1[207]=v3;
  v3[207]=v1[206]+v2[208];v3[207]=v1[206]+v2[208];
  v1[208]=v3;
  v3[208]=v1[207]+v2[209];v3[208]=v1[207]+v2[209];
  v1[209]=v3;
  v3[209]=v1[208]+v2[210];v3[209]=v1[208]+v2[210];
  v1[210]=v3;
  v3[210]=v1[209]+v2[211];v3[210]=v1[209]+v2[211];
  v1[211]=v3;
  v3[211]=v1[210]+v2[212];v3[211]=v1[210]+v2[212];
  v1[212]=v3;
  v3[212]=v1[211]+v2[21
```

Quel est le code le plus rapide? le plus lisible?

Pour 75 000 000 executions:

1	245ms	2	245ms	3	245ms	4	245ms
<pre>void fonction(int v1[],int v2[],int v3[],int N) { unsigned int k=0; for(k=1;k<N-1;++k) { v1[k]=v2; v3[k]=v1[k-1]-v2[k+1]; v3[k]/=10; } }</pre>	<pre>void fonction(int v1[],int v2[],int v3[],int N) { int *p_v1=v1,*p_v2=v2,*p_v3=v3+1; int *p_v2v2=v2; int *p_v3v3=v3; register unsigned int k=1; while(k<N-1) { *p_v1_2v1++ += 3; *p_v3 += *p_v1++-*p_v2v2++; *p_v3+=*p_v3; ++k; } }</pre>	<pre>void fonction(int v1[],int v2[],int v3[],int N) { v1[1]=v2; v2[1]=v1[0]-v2[2];v2[2]=v3[1]-v2; v3[1]=v2; v3[2]=v1[1]-v2[3];v3[3]=v2-v2; v3[4]=v2; v3[5]=v1[2]-v2[4];v3[6]=v2; v3[7]=v2; v3[8]=v1[3]-v2[5];v3[9]=v2; v3[10]=v2; v3[11]=v1[4]-v2[6];v3[12]=v2; v3[13]=v2; v3[14]=v1[5]-v2[7];v3[15]=v2; v3[16]=v2; v3[17]=v1[6]-v2[8];v3[18]=v2; v3[19]=v2; v3[20]=v1[7]-v2[9];v3[21]=v2; v3[22]=v2; v3[23]=v1[8]-v2[10];v3[24]=v2; v3[25]=v2; v3[26]=v1[9]-v2[11];v3[27]=v2; v3[28]=v2; v3[29]=v1[10]-v2[12];v3[30]=v2; v3[31]=v2; v3[32]=v1[11]-v2[13];v3[33]=v2; v3[34]=v2; v3[35]=v1[12]-v2[14];v3[36]=v2; v3[37]=v2; v3[38]=v1[13]-v2[15];v3[39]=v2; v3[40]=v2; v3[41]=v1[14]-v2[16];v3[42]=v2; v3[43]=v2; v3[44]=v1[15]-v2[17];v3[45]=v2; v3[46]=v2; v3[47]=v1[16]-v2[18];v3[48]=v2; v3[49]=v2; v3[50]=v1[17]-v2[19];v3[51]=v2; v3[52]=v2; v3[53]=v1[18]-v2[20];v3[54]=v2; v3[55]=v2; v3[56]=v1[19]-v2[21];v3[57]=v2; v3[58]=v2; v3[59]=v1[20]-v2[22];v3[60]=v2; v3[61]=v2; v3[62]=v1[21]-v2[23];v3[63]=v2; v3[64]=v2; v3[65]=v1[22]-v2[24];v3[66]=v2; v3[67]=v2; v3[68]=v1[23]-v2[25];v3[69]=v2; v3[70]=v2; v3[71]=v1[24]-v2[26];v3[72]=v2; v3[73]=v2; v3[74]=v1[25]-v2[27];v3[75]=v2; v3[76]=v2; v3[77]=v1[26]-v2[28];v3[78]=v2; v3[79]=v2; v3[80]=v1[27]-v2[29];v3[81]=v2; v3[82]=v2; v3[83]=v1[28]-v2[30];v3[84]=v2; v3[85]=v2; v3[86]=v1[29]-v2[31];v3[87]=v2; v3[88]=v2; v3[89]=v1[30]-v2[32];v3[90]=v2; v3[91]=v2; v3[92]=v1[31]-v2[33];v3[93]=v2; v3[94]=v2; v3[95]=v1[32]-v2[34];v3[96]=v2; v3[97]=v2; v3[98]=v1[33]-v2[35];v3[99]=v2; v3[100]=v2; v3[101]=v1[34]-v2[36];v3[102]=v2; v3[103]=v2; v3[104]=v1[35]-v2[37];v3[105]=v2; v3[106]=v2; v3[107]=v1[36]-v2[38];v3[108]=v2; v3[109]=v2; v3[110]=v1[37]-v2[39];v3[111]=v2; v3[112]=v2; v3[113]=v1[38]-v2[40];v3[114]=v2; v3[115]=v2; v3[116]=v1[39]-v2[41];v3[117]=v2; v3[118]=v2; v3[119]=v1[40]-v2[42];v3[120]=v2; v3[121]=v2; v3[122]=v1[41]-v2[43];v3[123]=v2; v3[124]=v2; v3[125]=v1[42]-v2[44];v3[126]=v2; v3[127]=v2; v3[128]=v1[43]-v2[45];v3[129]=v2; v3[130]=v2; v3[131]=v1[44]-v2[46];v3[132]=v2; v3[133]=v2; v3[134]=v1[45]-v2[47];v3[135]=v2; v3[136]=v2; v3[137]=v1[46]-v2[48];v3[138]=v2; v3[139]=v2; v3[140]=v1[47]-v2[49];v3[141]=v2; v3[142]=v2; v3[143]=v1[48]-v2[50];v3[144]=v2; v3[145]=v2; v3[146]=v1[49]-v2[51];v3[147]=v2; v3[148]=v2; v3[149]=v1[50]-v2[52];v3[151]=v2; v3[152]=v2; v3[153]=v1[51]-v2[53];v3[154]=v2; v3[155]=v2; v3[156]=v1[52]-v2[54];v3[157]=v2; v3[158]=v2; v3[159]=v1[53]-v2[55];v3[159]=v2; v3[160]=v2; v3[161]=v1[54]-v2[56];v3[161]=v2; v3[162]=v2; v3[163]=v1[55]-v2[57];v3[163]=v2; v3[164]=v2; v3[165]=v1[56]-v2[58];v3[165]=v2; v3[166]=v2; v3[167]=v1[57]-v2[59];v3[167]=v2; v3[168]=v2; v3[169]=v1[58]-v2[60];v3[169]=v2; v3[170]=v2; v3[171]=v1[59]-v2[61];v3[171]=v2; v3[172]=v2; v3[173]=v1[60]-v2[62];v3[173]=v2; v3[174]=v2; v3[175]=v1[61]-v2[63];v3[175]=v2; v3[176]=v2; v3[177]=v1[62]-v2[64];v3[177]=v2; v3[178]=v2; v3[179]=v1[63]-v2[65];v3[179]=v2; v3[180]=v2; v3[181]=v1[64]-v2[66];v3[181]=v2; v3[182]=v2; v3[183]=v1[65]-v2[67];v3[183]=v2; v3[184]=v2; v3[185]=v1[66]-v2[68];v3[185]=v2; v3[186]=v2; v3[187]=v1[67]-v2[69];v3[187]=v2; v3[188]=v2; v3[189]=v1[68]-v2[70];v3[189]=v2; v3[190]=v2; v3[191]=v1[69]-v2[71];v3[191]=v2; v3[192]=v2; v3[193]=v1[70]-v2[72];v3[193]=v2; v3[194]=v2; v3[195]=v1[71]-v2[73];v3[195]=v2; v3[196]=v2; v3[197]=v1[72]-v2[74];v3[197]=v2; v3[198]=v2; v3[199]=v1[73]-v2[75];v3[199]=v2; v3[200]=v2; v3[201]=v1[74]-v2[76];v3[201]=v2; v3[202]=v2; v3[203]=v1[75]-v2[77];v3[203]=v2; v3[204]=v2; v3[205]=v1[76]-v2[78];v3[205]=v2; v3[206]=v2; v3[207]=v1[77]-v2[79];v3[207]=v2; v3[208]=v2; v3[209]=v1[78]-v2[80];v3[209]=v2; v3[210]=v2; v3[211]=v1[79]-v2[81];v3[211]=v2; v3[212]=v2; v3[213]=v1[80]-v2[82];v3[213]=v2; v3[214]=v2; v3[215]=v1[81]-v2[83];v3[215]=v2; v3[216]=v2; v3[217]=v1[82]-v2[84];v3[217]=v2; v3[218]=v2; v3[219]=v1[83]-v2[85];v3[219]=v2; v3[220]=v2; v3[221]=v1[84]-v2[86];v3[221]=v2; v3[222]=v2; v3[223]=v1[85]-v2[87];v3[223]=v2; v3[224]=v2; v3[225]=v1[86]-v2[88];v3[225]=v2; v3[226]=v2; v3[227]=v1[87]-v2[89];v3[227]=v2; v3[228]=v2; v3[229]=v1[88]-v2[90];v3[229]=v2; v3[230]=v2; v3[231]=v1[89]-v2[91];v3[231]=v2; v3[232]=v2; v3[233]=v1[90]-v2[92];v3[233]=v2; v3[234]=v2; v3[235]=v1[91]-v2[93];v3[235]=v2; v3[236]=v2; v3[237]=v1[92]-v2[94];v3[237]=v2; v3[238]=v2; v3[239]=v1[93]-v2[95];v3[239]=v2; v3[240]=v2; v3[241]=v1[94]-v2[96];v3[241]=v2; v3[242]=v2; v3[243]=v1[95]-v2[97];v3[243]=v2; v3[244]=v2; v3[245]=v1[96]-v2[98];v3[245]=v2; v3[246]=v2; v3[247]=v1[97]-v2[99];v3[247]=v2; v3[248]=v2; v3[249]=v1[98]-v2[100];v3[249]=v2; v3[250]=v2; v3[251]=v1[99]-v2[101];v3[251]=v2; v3[252]=v2; v3[253]=v1[100]-v2[102];v3[253]=v2; v3[254]=v2; v3[255]=v1[101]-v2[103];v3[255]=v2; v3[256]=v2; v3[257]=v1[102]-v2[104];v3[257]=v2; v3[258]=v2; v3[259]=v1[103]-v2[105];v3[259]=v2; v3[260]=v2; v3[261]=v1[104]-v2[106];v3[261]=v2; v3[262]=v2; v3[263]=v1[105]-v2[107];v3[263]=v2; v3[264]=v2; v3[265]=v1[106]-v2[108];v3[265]=v2; v3[266]=v2; v3[267]=v1[107]-v2[109];v3[267]=v2; v3[268]=v2; v3[269]=v1[108]-v2[110];v3[269]=v2; v3[270]=v2; v3[271]=v1[109]-v2[111];v3[271]=v2; v3[272]=v2; v3[273]=v1[110]-v2[112];v3[273]=v2; v3[274]=v2; v3[275]=v1[111]-v2[113];v3[275]=v2; v3[276]=v2; v3[277]=v1[112]-v2[114];v3[277]=v2; v3[278]=v2; v3[279]=v1[113]-v2[115];v3[279]=v2; v3[280]=v2; v3[281]=v1[114]-v2[116];v3[281]=v2; v3[282]=v2; v3[283]=v1[115]-v2[117];v3[283]=v2; v3[284]=v2; v3[285]=v1[116]-v2[118];v3[285]=v2; v3[286]=v2; v3[287]=v1[117]-v2[119];v3[287]=v2; v3[288]=v2; v3[289]=v1[118]-v2[120];v3[289]=v2; v3[290]=v2; v3[291]=v1[119]-v2[121];v3[291]=v2; v3[292]=v2; v3[293]=v1[120]-v2[122];v3[293]=v2; v3[294]=v2; v3[295]=v1[121]-v2[123];v3[295]=v2; v3[296]=v2; v3[297]=v1[122]-v2[124];v3[297]=v2; v3[298]=v2; v3[299]=v1[123]-v2[125];v3[299]=v2; v3[300]=v2; v3[301]=v1[124]-v2[126];v3[301]=v2; v3[302]=v2; v3[303]=v1[125]-v2[127];v3[303]=v2; v3[304]=v2; v3[305]=v1[126]-v2[128];v3[305]=v2; v3[306]=v2; v3[307]=v1[127]-v2[129];v3[307]=v2; v3[308]=v2; v3[309]=v1[128]-v2[130];v3[309]=v2; v3[310]=v2; v3[311]=v1[129]-v2[131];v3[311]=v2; v3[312]=v2; v3[313]=v1[130]-v2[132];v3[313]=v2; v3[314]=v2; v3[315]=v1[131]-v2[133];v3[315]=v2; v3[316]=v2; v3[317]=v1[132]-v2[134];v3[317]=v2; v3[318]=v2; v3[319]=v1[133]-v2[135];v3[319]=v2; v3[320]=v2; v3[321]=v1[134]-v2[136];v3[321]=v2; v3[322]=v2; v3[323]=v1[135]-v2[137];v3[323]=v2; v3[324]=v2; v3[325]=v1[136]-v2[138];v3[325]=v2; v3[326]=v2; v3[327]=v1[137]-v2[139];v3[327]=v2; v3[328]=v2; v3[329]=v1[138]-v2[140];v3[329]=v2; v3[330]=v2; v3[331]=v1[139]-v2[141];v3[331]=v2; v3[332]=v2; v3[333]=v1[140]-v2[142];v3[333]=v2; v3[334]=v2; v3[335]=v1[141]-v2[143];v3[335]=v2; v3[336]=v2; v3[337]=v1[142]-v2[144];v3[337]=v2; v3[338]=v2; v3[339]=v1[143]-v2[145];v3[339]=v2; v3[340]=v2; v3[341]=v1[144]-v2[146];v3[341]=v2; v3[342]=v2; v3[343]=v1[145]-v2[147];v3[343]=v2; v3[344]=v2; v3[345]=v1[146]-v2[148];v3[345]=v2; v3[346]=v2; v3[347]=v1[147]-v2[149];v3[347]=v2; v3[348]=v2; v3[349]=v1[148]-v2[150];v3[349]=v2; v3[350]=v2; v3[351]=v1[149]-v2[151];v3[351]=v2; v3[352]=v2; v3[353]=v1[150]-v2[152];v3[353]=v2; v3[354]=v2; v3[355]=v1[151]-v2[153];v3[355]=v2; v3[356]=v2; v3[357]=v1[152]-v2[154];v3[357]=v2; v3[358]=v2; v3[359]=v1[153]-v2[155];v3[359]=v2; v3[360]=v2; v3[361]=v1[154]-v2[156];v3[361]=v2; v3[362]=v2; v3[363]=v1[155]-v2[157];v3[363]=v2; v3[364]=v2; v3[365]=v1[156]-v2[158];v3[365]=v2; v3[366]=v2; v3[367]=v1[157]-v2[159];v3[367]=v2; v3[368]=v2; v3[369]=v1[158]-v2[160];v3[369]=v2; v3[370]=v2; v3[371]=v1[159]-v2[161];v3[371]=v2; v3[372]=v2; v3[373]=v1[160]-v2[162];v3[373]=v2; v3[374]=v2; v3[375]=v1[161]-v2[163];v3[375]=v2; v3[376]=v2; v3[377]=v1[162]-v2[164];v3[377]=v2; v3[378]=v2; v3[379]=v1[163]-v2[165];v3[379]=v2; v3[380]=v2; v3[381]=v1[164]-v2[166];v3[381]=v2; v3[382]=v2; v3[383]=v1[165]-v2[167];v3[383]=v2; v3[384]=v2; v3[385]=v1[166]-v2[168];v3[385]=v2; v3[386]=v2; v3[387]=v1[167]-v2[169];v3[387]=v2; v3[388]=v2; v3[389]=v1[168]-v2[170];v3[389]=v2; v3[390]=v2; v3[391]=v1[169]-v2[171];v3[391]=v2; v3[392]=v2; v3[393]=v1[170]-v2[172];v3[393]=v2; v3[394]=v2; v3[395]=v1[171]-v2[173];v3[395]=v2; v3[396]=v2; v3[397]=v1[172]-v2[174];v3[397]=v2; v3[398]=v2; v3[399]=v1[173]-v2[175];v3[399]=v2; v3[400]=v2; v3[401]=v1[174]-v2[176];v3[401]=v2; v3[402]=v2; v3[403]=v1[175]-v2[177];v3[403]=v2; v3[404]=v2; v3[405]=v1[176]-v2[178];v3[405]=v2; v3[406]=v2; v3[407]=v1[177]-v2[179];v3[407]=v2; v3[408]=v2; v3[409]=v1[178]-v2[180];v3[409]=v2; v3[410]=v2; v3[411]=v1[179]-v2[181];v3[411]=v2; v3[412]=v2; v3[413]=v1[180]-v2[182];v3[413]=v2; v3[414]=v2; v3[415]=v1[181]-v2[183];v3[415]=v2; v3[416]=v2; v3[417]=v1[182]-v2[184];v3[417]=v2; v3[418]=v2; v3[419]=v1[183]-v2[185];v3[419]=v2; v3[420]=v2; v3[421]=v1[184]-v2[186];v3[421]=v2; v3[422]=v2; v3[423]=v1[185]-v2[187];v3[423]=v2; v3[424]=v2; v3[425]=v1[186]-v2[188];v3[425]=v2; v3[426]=v2; v3[427]=v1[187]-v2[189];v3[427]=v2; v3[428]=v2; v3[429]=v1[188]-v2[190];v3[429]=v2; v3[430]=v2; v3[431]=v1[189]-v2[191];v3[431]=v2; v3[432]=v2; v3[433]=v1[190]-v2[192];v3[433]=v2; v3[434]=v2; v3[435]=v1[191]-v2[193];v3[435]=v2; v3[436]=v2; v3[437]=v1[192]-v2[194];v3[437]=v2; v3[438]=v2; v3[439]=v1[193]-v2[195];v3[439]=v2; v3[440]=v2; v3[441]=v1[194]-v2[196];v3[441]=v2; v3[442]=v2; v3[443]=v1[195]-v2[197];v3[443]=v2; v3[444]=v2; v3[445]=v1[196]-v2[198];v3[445]=v2; v3[446]=v2; v3[447]=v1[197]-v2[199];v3[447]=v2; v3[448]=v2; v3[449]=v1[198]-v2[200];v3[449]=v2; v3[450]=v2; v3[451]=v1[199]-v2[201];v3[451]=v2; v3[452]=v2; v3[453]=v1[200]-v2[202];v3[453]=v2; v3[454]=v2; v3[455]=v1[201]-v2[203];v3[455]=v2; v3[456]=v2; v3[457]=v1[202]-v2[204];v3[457]=v2; v3[458]=v2; v3[459]=v1[203]-v2[205];v3[459]=v2; v3[460]=v2; v3[461]=v1[204]-v2[206];v3[461]=v2; v3[462]=v2; v3[463]=v1[205]-v2[207];v3[463]=v2; v3[464]=v2; v3[465]=v1[206]-v2[208];v3[465]=v2; v3[466]=v2; v3[467]=v1[207]-v2[209];v3[467]=v2; v3[468]=v2; v3[469]=v1[208]-v2[210];v3[469]=v2; v3[470]=v2; v3[471]=v1[209]-v2[211];v3[471]=v2; v3[472]=v2; v3[473]=v1[210]-v2[212];v3[473]=v2; v3[474]=v2; v3[475]=v1[211]-v2[213];v3[475]=v2; v3[476]=v2; v3[477]=v1[212]-v2[214];v3[477]=v2; v3[478]=v2; v3[479]=v1[213]-v2[215];v3[479]=v2; v3[480]=v2; v3[481]=v1[214]-v2[216];v3[481]=v2; v3[482]=v2; v3[483]=v1[215]-v2[217];v3[483]=v2; v3[484]=v2; v3[485]=v1[216]-v2[218];v3[485]=v2; v3[486]=v2; v3[487]=v1[217]-v2[219];v3[487]=v2; v3[488]=v2; v3[489]=v1[218]-v2[220];v3[489]=v2; v3[490]=v2; v3[491]=v1[219]-v2[221];v3[491]=v2; v3[492]=v2; v3[493]=v1[220]-v2[222];v3[493]=v2; v3[494]=v2; v3[495]=v1[221]-v2[223];v3[495]=v2; v3[496]=v2; v3[497]=v1[222]-v2[224];v3[497]=v2; v3[498]=v2; v3[499]=v1[223]-v2[225];v3[499]=v2; v3[500]=v2; v3[501]=v1[224]-v2[226];v3[501]=v2; v3[502]=v2; v3[503]=v1[225]-v2[227];v3[503]=v2; v3[504]=v2; v3[505]=v1[226]-v2[228];v3[505]=v2; v3[506]=v2; v3[507]=v1[227]-v2[229];v3[507]=v2; v3[508]=v2; v3[509]=v1[228]-v2[230];v3[509]=v2; v3[510]=v2; v3[511]=v1[229]-v2[231];v3[511]=v2; v3[512]=v2; v3[513]=v1[230]-v2[232];v3[513]=v2; v3[514]=v2; v3[515]=v1[231]-v2[233];v3[515]=v2; v3[516]=v2; v3[517]=v1[232]-v2[234];v3[517]=v2; v3[518]=v2; v3[519]=v1[233]-v2[235];v3[519]=v2; v3[520]=v2; v3[521]=v1[234]-v2[236];v3[521]=v2; v3[522]=v2; v3[523]=v1[235]-v2[237];v3[523]=v2; v3[524]=v2; v3[525]=v1[236]-v2[238];v3[525]=v2; v3[526]=v2; v3[527]=v1[237]-v2[239];v3[527]=v2; v3[528]=v2; v3[529]=v1[238]-v2[240];v3[529]=v2; v3[530]=v2; v3[531]=v1[239]-v2[241];v3[531]=v2; v3[532]=v2; v3[533]=v1[240]-v2[242];v3[533]=v2; v3[534]=v2; v3[535]=v1[241]-v2[243];v3[535]=v2; v3[536]=v2; v3[537]=v1[242]-v2[244];v3[537]=v2; v3[538]=v2; v3[539]=v1[243]-v2[245];v3[539]=v2; v3[540]=v2; v3[541]=v1[244]-v2[246];v3[541]=v2; v3[542]=v2; v3[543]=v1[245]-v2[247];v3[543]=v2; v3[544]=v2; v3[545]=v1[246]-v2[248];v3[545]=v2; v3[546]=v2; v3[547]=v1[247]-v2[249];v3[547]=v2; v3[548]=v2; v3[549]=v1[248]-v2[250];v3[549]=v2; v3[550]=v2; v3[551]=v1[249]-v2[251];v3[551]=v2; v3[552]=v2; v3[553]=v1[250]-v2[252];v3[553]=v2; v3[554]=v2; v3[555]=v1[251]-v2[253];v3[555]=v2; v3[556]=v2; v3[557]=v1[252]-v2[254];v3[557]=v2; v3[558]=v2; v3[559]=v1[253]-v2[255];v3[559]=v2; v3[560]=v2; v3[561]=v1[254]-v2[256];v3[561]=v2; v3[562]=v2; v3[563]=v1[255]-v2[257];v3[563]=v2; v3[564]=v2; v3[565]=v1[256]-v2[258];v3[565]=v2; v3[566]=v2; v3[567]=v1[257]-v2[259];v3[567]=v2; v3[568]=v2; v3[569]=v1[258]-v2[260];v3[569]=v2; v3[570]=v2; v3[571]=v1[259]-v2[261];v3[571]=v2; v3[572]=v2; v3[573]=v1[260]-v2[262];v3[573]=v2; v3[574]=v2; v3[575]=v1[261]-v2[263];v3[575]=v2; v3[576]=v2; v3[577]=v1[262]-v2[264];v3[577]=v2; v3[578]=v2; v3[579]=v1[263]-v2[265];v3[579]=v2; v3[580]=v2; v3[581]=v1[264]-v2[266];v3[581]=v2; v3[582]=v2; v3[583]=v1[265]-v2[267];v3[583]=v2; v3[584]=v2; v3[585]=v1[266]-v2[268];v3[585]=v2; v3[586]=v2; v3[587]=v1[267]-v2[269];v3[587]=v2; v3[588]=v2; v3[589]=v1[268]-v2[270];v3[589]=v2; v3[590]=v2; v3[591]=v1[269]-v2[271];v3[591]=v2; v3[592]=v2; v3[593</pre>					

Bonnes pratiques

Remarque optimisation 2:

GCC optimise | très bien
| souvent mieux qu'un humain!

ex. somme coefficient d'une matrice:

```
int main()
{
    int matrice[3][3]={{1,2,3},
                      {4,1,-5},
                      {7,7,1}};

    int somme=0;

    int kx=0,ky=0;
    for(kx=0;kx<3;++kx)
        for(ky=0;ky<3;++ky)
            somme += matrice[kx][ky];

    printf("%d\n",somme);
}
```

déjà calculé
O(1)

```
main:
.LFB0:
.cfi_startproc
movl $21,%esi
movl $.LC0,%edi
xorl %eax,%eax
jmp printf
.cfi_endproc
```

\$ gcc -O2

045

Gestion code important

- IDE
- Séparation en-tête/implémentation
- Gestionnaire de version

047

Bonnes pratiques

Remarque optimisation 2:

GCC optimise | très bien
| souvent mieux qu'un humain!

ex. somme coefficient d'une matrice:

```
int main()
{
    int matrice[3][3]={{1,2,3},
                      {4,1,-5},
                      {7,7,1}};

    int somme=0;

    int *p=matrice;
    register int c=0;
    while(c++ < 9)
        somme += *(p++);

    printf("%d\n",somme);
}
```

- lisible
- rapide !!

```
main:
.LFB0:
.cfi_startproc
movl $1,-56(%rsp)
movl $2,-52(%rsp)
movl $.LC0,%edi
movl $3,-48(%rsp)
movl $4,-44(%rsp)
xorl %eax,%eax
movdqa -56(%rsp),%xmm0
movl $1,-40(%rsp)
movl $-5,-36(%rsp)
movl $7,-32(%rsp)
movl $7,-28(%rsp)
paddq -40(%rsp),%xmm0
movdqa %xmm0,%xmm1
movl $1,-24(%rsp)
psrldq $8,%xmm1
paddq %xmm1,%xmm0
movdqa %xmm0,%xmm1
psrldq $4,%xmm1
paddq %xmm1,%xmm0
movd %xmm0,-60(%rsp)
movl -60(%rsp),%esi
addl $1,%esi
jmp printf
.cfi_endproc
```

\$ gcc -O2

046

Utilisation d'un IDE

*Integrated
Development
Environments*

Evite perte de temps
Complétion automatique
Rappel mémoire

```
int main()
{
    struct voiture ma_bmw;

    ma_bmw.roue[2].
    return 0;
}
```

nom_constructeur
pression
prix
roue_voiture

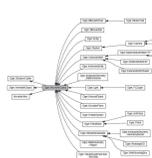
IDE Linux:
QtCreator
Eclipse
Code::Blocks

Editeur texte:
Vi, Emacs, gedit, ...

048

Developpement logiciel

Réaliser le **design** d'un **code volumineux**.



reflection en amont de coder
coder
reflection en aval



1 personne: ~30 000 lignes
équipe: 200 000 lignes
logiciel: 1 000 000 lignes
ex. Windows XP (40 000 000 lignes)

049

Fichiers en-tête

Signature/en tête d'une fonction

Implémentation / corps d'une fonction

051

Gestion code important

IDE

→ **Séparation en-tête/implémentation**

Gestionnaire de version

050

Fichiers en-tête

Solution 1:

```
int somme(int tableau[], unsigned int taille)
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0; k<taille; ++k)
        somme_courante += tableau[k];
    return somme_courante;
}

int main()
{
    int T[]={1,4,5,7};
    int s=somme(T, sizeof(T)/sizeof(int));
    printf("%d\n", s);
    return 0;
}
```

Solution 2:

```
int somme(int tableau[], unsigned int taille);

int main()
{
    int T[]={1,4,5,7};
    int s=somme(T, sizeof(T)/sizeof(int));
    printf("%d\n", s);
    return 0;
}

int somme(int tableau[], unsigned int taille)
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0; k<taille; ++k)
        somme_courante += tableau[k];
    return somme_courante;
}
```

052

Fichiers en-tête

Solution 3:



fichier.h

```
int somme(int tableau[], unsigned int taille)
```



main.c

```
#include <fichier.h>
int main()
{
    int T[]={1,4,5,7};

    int s=somme(T,sizeof(T)/sizeof(int));
    printf("%d\n",s);

    return 0;
}
```



fichier.c

```
int somme(int tableau[], unsigned int taille;
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0;k<taille;++k)
        somme_courante += tableau[k];
    return somme_courante;
}
```

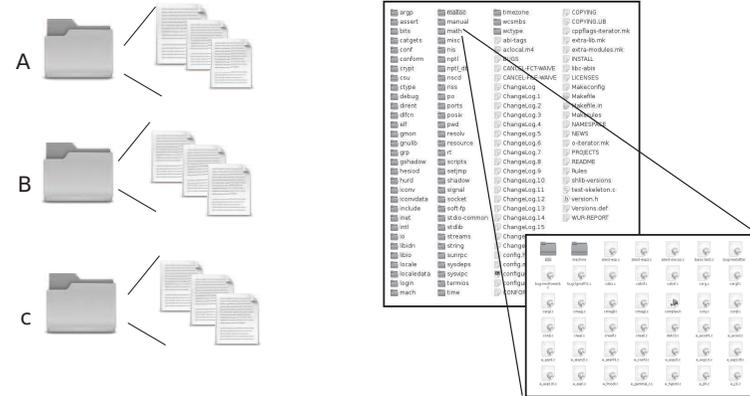
053

Fichiers en-tête

Synthèse:

Pour les très gros projets:

ex. LibC



055

Fichiers en-tête

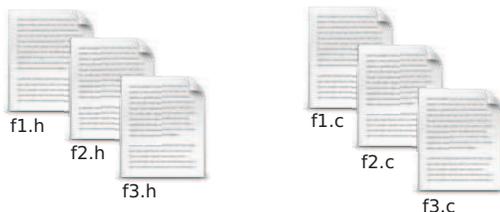
Synthèse:

- + Evite les longs fichiers
- + Séparation en-tête / implémentation



En pratique:

Autant de fichiers que d'abstractions



054

Fichiers en-tête : en pratique



main.c

```
#include "fichier.h"

int main()
{
    int T[]={1,4,5,7};

    int s=somme(T,4);
    printf("%d\n",s);

    return 0;
}
```

Le fichier d'en tête (pas l'implémentation .c!)

#include "XXX" : locale
#include <XXX> : système

Copie-Colle le contenu

La signature est connue à partir de fichier.h

056

Fichiers en-tête : en pratique

```
main.c
#include "fichier.h"
int main()
{
    int T[]={1,4,5,7};
    int s=somme(T,4);
    printf("%d\n",s);
    return 0;
}
```

Implémentation (corps) fonctions

```
fichier.c
#include "fichier.h"
int somme(int tableau[],unsigned int taille)
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0;k<taille;++k)
        somme_courante += tableau[k];
    return somme_courante;
}
```

Copie-Colle le contenu

057

Fichiers en-tête : en pratique

```
main.c
#include "fichier.h"
int main()
{
    int T[]={1,4,5,7};
    int s=somme(T,4);
    printf("%d\n",s);
    return 0;
}

fichier.c
#include "fichier.h"
int somme(int tableau[],unsigned int taille)
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0;k<taille;++k)
        somme_courante += tableau[k];
    return somme_courante;
}

fichier.h
#ifndef FICHIER_H
#define FICHIER_H
//Realise la somme des elements
// d'un tableau d'entier
//Recoit un tableau et une taille n (>0)
//
//Retourne la somme des n elements
// du tableau
int somme(int tableau[],
         unsigned int taille);
#endif
```

Compilation séparée:

```
$ gcc -c main.c -g -Wall -Wextra
$ gcc -c fichier.c -g -Wall -Wextra
$ gcc main.o fichier.o -o mon_executable
```

→ Compilation main.c -> main.o
 → Compilation fichier.c -> fichier.o
 → Edition des liens création executable avec main.o et fichier.o

On ne compile pas les fichier .h (ils sont copiés-collés)

059

Fichiers en-tête : en pratique

```
main.c
#include "fichier.h"
int main()
{
    int T[]={1,4,5,7};
    int s=somme(T,4);
    printf("%d\n",s);
    return 0;
}

fichier.c
#include "fichier.h"
int somme(int tableau[],unsigned int taille)
{
    int somme_courante=0;
    unsigned int k=0;
    for(k=0;k<taille;++k)
        somme_courante += tableau[k];
    return somme_courante;
}
```

En tête/signature

```
fichier.h
#include guards: #ifndef/#define ID_FICHIER
#include "fichier.h"
#include "fichier.h"
> Error multiple definition

//Realise la somme des elements
// d'un tableau d'entier
//Recoit un tableau et une taille n (>0)
//
//Retourne la somme des n elements
// du tableau
int somme(int tableau[],
         unsigned int taille);
signature fonction: entrés/sorties

#endif
Fin include guards
```

Documentation 1ère importance fichier .h (vu de tous!)

058

Gestion code important

- IDE
- Séparation en-tête/implémentation
- **Gestionnaire de version**

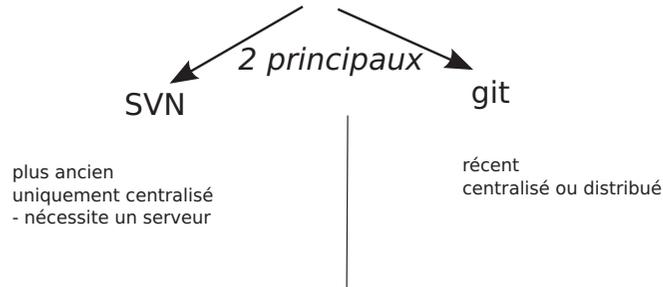
060

Programmer à plusieurs

Parties spécifiques *inter-dépendance*

Passage par mails *petit code uniquement*

Logiciel de controle de version (pro)



061

Programmer à plusieurs

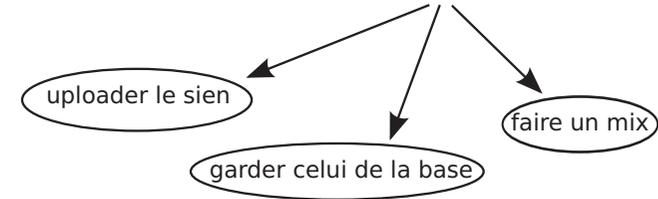
Principe d'un logiciel de controle de version

A chaque *commit*:

Si nouvelle version mise à jour entre temps
entre update et commit

Modifications indépendantes => pas de conflits

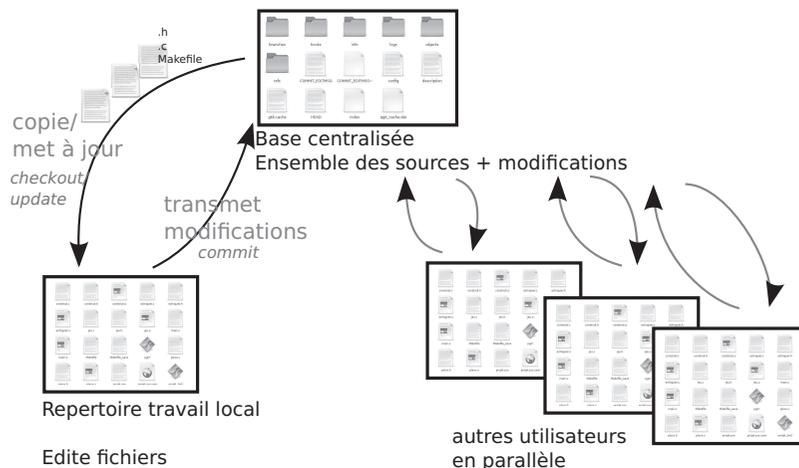
Modifications d'un même contenu => choix



063

Programmer à plusieurs

Principe d'un logiciel de controle de version



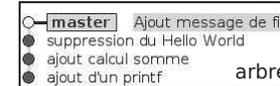
062

Programmer à plusieurs

Principe d'un logiciel de controle de version

La base garde en mémoire l'ensemble des modifications!

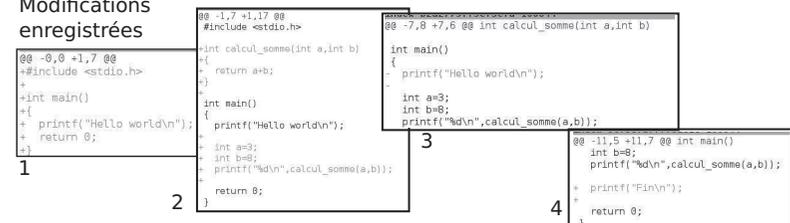
- + On ne perd aucune version
- + On peut revenir en arrière



Fichier local:

```
#include <stdio.h>
int calcul_somme(int a,int b)
{
    return a+b;
}
int main()
{
    int a=3;
    int b=8;
    printf("%d\n",calcul_somme(a,b));
    printf("Fin\n");
    return 0;
}
```

Modifications enregistrées



064

Programmer à plusieurs

Principe d'un logiciel de controle de version.
Mode d'emploi:

```
$ emacs mon_fichier.c #creation d'un/plusieurs fichiers sources
$ git init             #initialisation du repertoire .git
$ git add mon_fichier.c #ajout du suivit du fichier d sign
$ emacs mon_fichier.c #Modification du/des fichiers sources ...
$ git commit -a       #upload des modifications
```

<http://git-scm.com/book/en/Git-Basics-Getting-a-Git-Repository>

github
SOCIAL CODING



065

Logiciel Libre

Un logiciel libre est-il gratuit?
Un freeware est-il un logiciel libre?
Un shareware est-il un logiciel libre?
Un logiciel open-source est-il un logiciel libre?
Peut-on revendre un logiciel libre?
Puis-je intégrer du code libre dans le mien?
Peut-on réutiliser à son nom du code d'un logiciel libre?

067

Licences logiciels

066

Logiciel Libre

Logiciel libre = définition de Richard Stallman
(*mouvement de pensée*)



créateur de la FSF

L'expression « logiciel libre » veut dire que le logiciel respecte la liberté de l'utilisateur et de la communauté. En gros, les utilisateurs ont la liberté d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer le logiciel. Avec ces libertés, les utilisateurs (à la fois individuellement et collectivement) contrôlent le programme et ce qu'il fait pour eux.

FSF FREE SOFTWARE
FOUNDATION
The Free Software Foundation (FSF) is a nonprofit with a worldwide mission to promote computer user freedom and to defend the rights of all free software users.

Un programme est un logiciel libre si vous, en tant qu'utilisateur de ce programme, avez les quatre libertés essentielles :

- 0/ La liberté d'exécuter le programme, pour tous les usages (liberté 0) ;
- 1/ La liberté d'étudier le fonctionnement du programme, et de le modifier pour qu'il effectue vos tâches informatiques comme vous le souhaitez (liberté 1) ;
l'accès au code source est une condition nécessaire ;
- 2/ La liberté de redistribuer des copies, donc d'aider votre voisin (liberté 2) ;
- 3/ La liberté de distribuer aux autres des copies de vos versions modifiées (liberté 3) ;
en faisant cela, vous donnez à toute la communauté une possibilité de profiter de vos changements ;
l'accès au code source est une condition nécessaire.

Note: Il existe des musiques libres, films libres, art libre, ...

068

Logiciel Open Source

Logiciel dont la licence respecte les critères de l'Open Source Initiative



définition: <http://opensource.org/docs/osd>

Pas uniquement code source disponible (mais ambiguïté existe)

Note: *Logiciel Libre => OpenSource*
Mais on peut être OpenSource et non Libre

Licences Open Source
<http://opensource.org/licenses/category>



069

Licences Copyright vs Copyleft

Licences fermées copyright: (all right reserved)

Vous n'avez pas le droit de redistribuer sous aucune forme.
Demande explicite nécessaire.

Attention: recopie de passage de site internet = illégale

ex. Copyright © 1997-2011 Cprogramming.com. All rights reserved.

Attention:

Freeware, Shareware = licence sous copyright

071

Licences Open Source

Il existe plusieurs licences open sources.

Licence=condition d'utilisation et diffusion du logiciel et de son code.

Licences issues de la FSF

Les plus répandues:

<http://www.gnu.org/copyleft/gpl.html>

GPL Utilisation code GPL => totalité du logiciel licence GPL
Ne peut pas être utilisé dans un projet sous copyright

LGPL Utilisation code LGPL => n'implique pas que l'ensemble soit LGPL
BSD Peut être utilisé pour une projet sous copyright
MIT

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (c) <year> <name of author>

This program is Free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see <http://www.gnu.org/licenses/>.
```

Note pour vidéos, arts, ...: licences Creative Common

070

Libre et gratuit

Libre et/ou OpenSource ne signifie pas gratuit!

Nombreux projets commerciaux:

Red Hat
GNAT (compilateur ADA)
MySQL Enterprise
NetBeans
Zimbra

Distribution gratuite ou payante
Service généralement payant
Entreprise fournissant un service

Oracle
Mandriva
Mozilla

...

...

072

Droits d'auteurs

Droit morale

Paternité
Choix de diffusion
...

Inaliénable, Imprescriptible

=> Il est interdit (et impossible)
de changer le nom de
l'auteur original!!

Droit patrimoniale

Privilège d'exploitation
(royalties, ...)

Note:
En France pas de brevets logiciels!