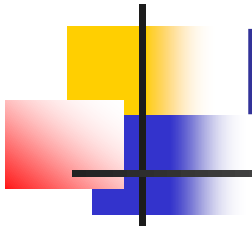


Présentation du Cours

- **Objectif** : Montrer que l'Informatique est un outil puissant pour résoudre des problèmes de Physique, de Mathématique, Scientifiques en général
- **Moyens** : Apprentissage des bases de la programmation impérative au travers du « langage C », langage couramment utilisé dans la communauté scientifique
- **Organisation de l'enseignement** :
  - 10h30 de cours en 7 séances de 1h30,
  - 12h de Tps en 8 séances de 1h30.

Présentation du Cours

Matériel pédagogique :

- Les documents de cours : le cours (« à trous »), les Tps, une fiche de « savoir faire » par chapitre, un mémento sont accessibles en ligne via la plateforme j@lon sur l'ENT.
- Une clé USB pour enregistrer ses exercices effectués en TP

Environnement de programmation utilisé en TP :

Dev-C++ : IDE (Integrated Development Environment)

Evaluation : 2 contrôles continus en TP (50%) , un contrôle final sur la totalité de l'enseignement (50%)

Présentation du Cours

A la maison, pour le choix de l'IDE :

Pour les utilisateurs de Mac, vous pouvez porter votre attention sur :

Code::Blocks -> http://www.codeblocks.org/downloads/26#mac

avec tutorial ici : http://loulou.developpez.com/tutoriels/cpp/codeblocks/

Possibilité d'importer un projet de dev-c++ utilisé en TP !

ou bien

xcode -> https://itunes.apple.com/fr/app/xcode/id497799835?mt=12&ign-mpt=uo%3D2

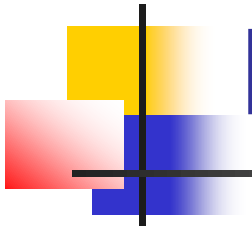
avec un tutorial ici : http://www.siteduzero.com/informatique/tutoriels/apprenez-a-programmer-en-c/xcode-mac-os-seulement

Pour les utilisateurs de Windows 8, une nouvelle version de dev-c++ est nécessaire :

http://www.windows8downloads.com/win8-dev-c--wdoxnrth/

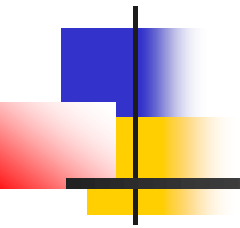
Pour les utilisateurs de Linux Ubuntu, un paquet du logiciel Code::Blocks est disponible via la logithèque Ubuntu.

Pour les autres systèmes Linux, aller directement sur la page de l'IDE Code::Blocks : http://www.codeblocks.org/downloads/26



Présentation du Cours

- **Me contacter** :
  - Courrier électronique : guingne@i3s.unice.fr
  - Petit Valrose 3ème étage



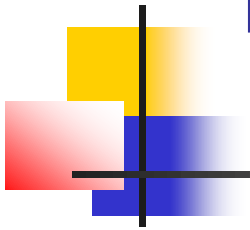
Initiation à la programmation impérative. Langage C.

L1 PC - SF

Franck GUINGNE

D'après le cours de Francis Avnaim

Plan du cours



1. Introduction

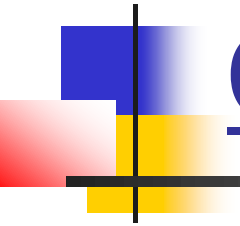
2. Programmation impérative. Introduction au langage C

3. Les bases du langage C

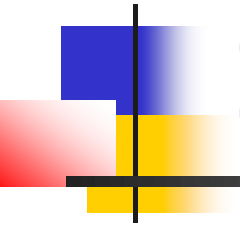
4. Les fonctions du langage C

5. Découpage d'un programme en fonctions.
   Méthodologie de programmation

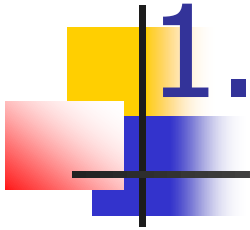
6. Étude de deux applications. Fichiers de données



Cours 1



1. Introduction



1.1 L'Informatique aujourd'hui

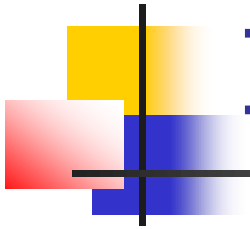
- Domaines d'applications
- Internet
- Les machines

Domaines d'applications de l'Informatique

- Nombreux et variés (l'Informatique a envahi notre quotidien; exemples : paiement par carte de crédit, téléphonie mobile, prévisions météo, GPS, etc.)

- Quelques grands domaines d'application :

  - Sciences
  - Banque, finance, gestion
  - Industrie
  - Médecine
  - Communication



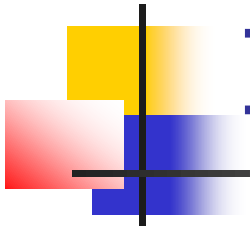
Internet : (bref) historique

- Naissance en 1969 : projet militaire Américain pour une transmission des informations décentralisée (Arpanet, protocole IP)
- Réseau NSF (National Science Fondation) des universitaires Américains
- Multiplication des réseaux et « fusion » en un seul réseau : Internet
- Création du Web par les chercheurs du Cern (Centre Européen de la Recherche Nucléaire) en 1989

Evolution d'Internet

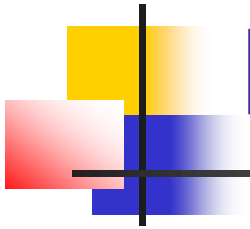
DATE	NUMBER OF USERS	% WORLD POPULATION	INFORMATION SOURCE
December, 1995	16 millions	0.4 %	IDC
March, 2000	304 millions	5.0 %	Nua Ltd.
March, 2005	888 millions	13.9 %	Internet World Stats
June, 2010	1,966 millions	28.7 %	Internet World Stats
Jun, 2011	2,110 millions	30.4 %	Internet World Stats
June, 2012	2,405 millions	34.3 %	Internet World Stats
Dec, 2013	2,802 millions	39.0 %	Internet World Stats
March, 2014 (estimate)	**2,937 millions**	**40.7 %**	Internet World Stats

Internet aujourd'hui



Internet aujourd'hui

- World Wide Web : 1,1 milliard de sites web en 2016 (source netcraft)
- eMail : 215 milliards envoyés par jour en 2016 (hors spam),
  4,4 milliards de comptes mails en 2015
- Réseaux sociaux (Facebook, Twitter, Instagram etc.)
- Téléphonie, vidéo conférences, chat (Skype, Messenger, etc.)
- Forums de discussion (IRC)
- Commerce électronique
- Transmission de fichiers (protocole ftp)



Les machines

- Les « machines » se classifient en fonction de leur puissance

- La puissance d'une machine se mesure en nombre d'opérations par seconde ou Flops (floating point operation per seconde). Les multiples sont :

  - Le Mégaflops : 1 million d'op. par seconde (10^6)
  - Le Gigaflops : 1 milliard d'op. par seconde (10^9)
  - Le Téraflops : mille milliards d'op. par seconde (10^12)
  - Le Pétaflops : un million de milliards d'op. par seconde (10^15)

Les machines

- Les super calculateurs :

Liste du top 5 des super calculateurs sur http://www.top500.org

2008 : **Roadrunner** de IBM : 1 PFLOPS

2010 : **Jaguar Cray XT5-HE :** 1.7 PFLOPS

2012 : **Sequoia – BlueGene/Q** de IBM : 16 PFLOPS

2014 : **Tianhe-2 (MilkyWay-2) - TH-IVB-FEP Cluster** de NUDT: 33 PFLOPS

2016 : **Sunway TaihuLight :** 93 PFLOPS

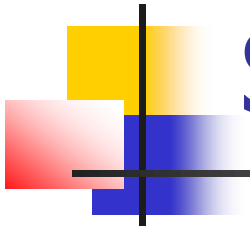
- Les macro ordinateurs ou mainframe :

  - Permettent la connexion simultanée de plusieurs centaines d'utilisateurs. Puissance > 100 Mégaflops
  - Présent dans les très grosses entreprises nationales et internationales
  - DEC, HP, Sun, IBM (ES/9000), Unisys

Les machines

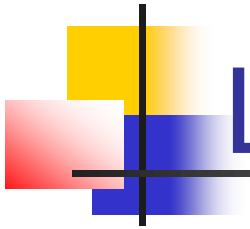
- Les mini ordinateurs :
  - Permettent la connexion simultanée de plusieurs dizaines d'utilisateurs. Puissance > 10 Mégaflops
  - Présents dans les grosses PME
  - DEC, HP, Sun, IBM (AS/400)
- Les ordinateurs personnels :
  - Stations de travail (Sun, Hp)
  - Micro-ordinateurs (PC, Mac) : de bureau (desktop), portables (notebook), de poche (laptop), internet (netbook)

1.2 Hardware (matériel), Software (logiciel)



L'informatique repose sur :

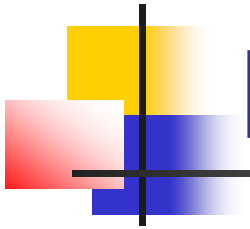
- Le hardware : physique du silicium et technologies associées (transistors miniaturisés, microprocesseurs, circuits intégrés, ordinateurs)
- Le software : programmation des ordinateurs
- Dans ce cours, nous traitons uniquement de la partie programmation



Le logiciel (Software)

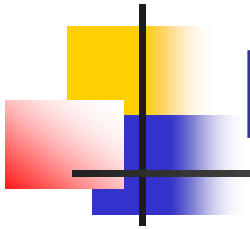
- Une machine est capable de traiter de l'information (symbolique, numérique)
- Pour qu'elle ait une utilité, il faut lui « dire » quels traitements effectuer, c'est-à-dire la programmer
- Logiciel : ensemble des programmes destinés à effectuer un traitement sur un ordinateur

1.3 Les langages de programmation



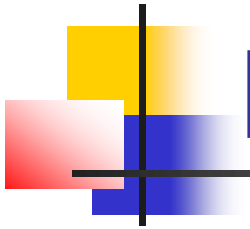
- Historique
- Environnement de programmation
- Champs d'applications des langages

Historique des langages de programmation



- Au commencement (~ 1946) le langage machine : des 1 et des 0 !!
- Un progrès : le langage d'assemblage (~1950)
- Premiers langages de haut niveau
  - Fortran (applications scientifiques -1956-)
  - Cobol (applications de gestion -1960-)
  - Lisp (intelligence artificielle -1960-)

Historique des langages de programmation



- Quelques référents :
  - Pascal (Wirth 1968)
  - ***C (Ritchie 1970)***
  - Smalltalk (Jey 1970)
  - Ada (Ichbiah 1975)
  - Prolog (Colmerauer, Roussel 1975)
  - C++ (Stroustrup 1983)
- Langages les plus récents (tous objets) :
  - Java (Sun) : 1991
  - Python (Guido Van Rossum) : 1991
  - C# (Microsoft) : 2001 … pour contrer Java et C++ !!

Utilisation des langages de programmation

- Sur le site http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Aug 2016	Aug 2015	Change	Programming Language	Ratings	Change
1	1		Java	19.010%	-0.26%
2	2		C	11.303%	-3.43%
3	3		C++	5.800%	-1.94%
4	4		C#	4.907%	+0.07%
5	5		Python	4.404%	+0.34%
6	7	^	PHP	3.173%	+0.44%
7	9	^	JavaScript	2.705%	+0.54%
8	8		Visual Basic .NET	2.518%	-0.19%
9	10	^	Perl	2.511%	+0.39%
10	12	^	Assembly language	2.364%	+0.60%
11	14	^	Delphi/Object Pascal	2.278%	+0.87%
12	13	^	Ruby	2.278%	+0.86%
13	11	v	Visual Basic	2.046%	+0.26%
14	17	^	Swift	1.983%	+0.80%
15	6	vv	Objective-C	1.884%	-1.31%
16	37	^^	Groovy	1.637%	+1.27%
17	20	^	R	1.605%	+0.60%
18	15	v	MATLAB	1.538%	+0.31%
19	19		PL/SQL	1.349%	+0.21%
20	95	^^	Go	1.270%	+1.19%

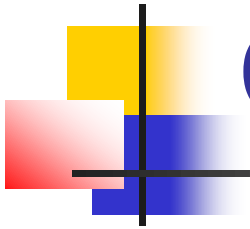
Les environnements de programmations modernes

Ils intègrent :

- Un éditeur intelligent
- Un compilateur
- Un débogueur symbolique
- Un générateur d'interface graphique

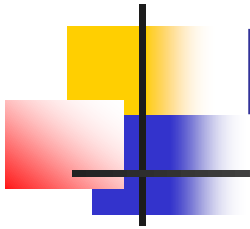
Éditeurs intelligents pour l'écriture de programmes

- Un langage de programmation a des règles syntaxiques strictes
- Un éditeur « intelligent » :
  - Signale des erreurs potentielles
  - Structure le texte automatiquement pour vous
- Exemple : si l'on ouvre une accolade « { », l'éditeur édite automatiquement l'accolade fermante correspondante « } »



Compilateurs

- Un ordinateur ne « comprend » que le langage machine (1 et 0)
- Le compilateur est un « gros » programme qui traduit le programme écrit dans un langage évolué (par exemple C) en un programme machine
- Si le compilateur est correct, l'ordinateur exécute bien les tâches décrites dans le langage de haut niveau
- Problème : l'œuf ou la poule ? (ou dans quel langage écrire un compilateur ?)



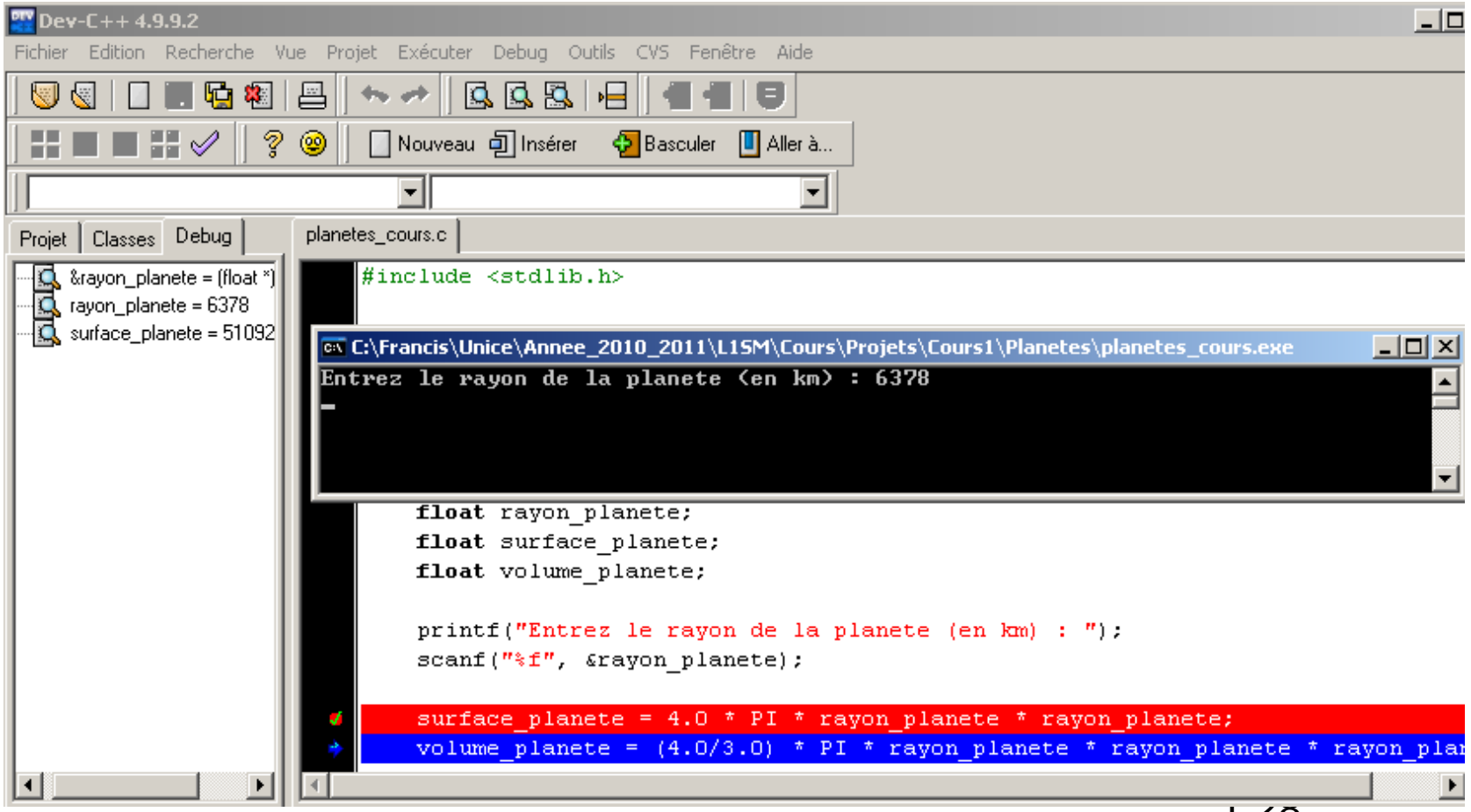
Débogueurs

- Un débogueur est un programme qui permet de suivre pas à pas l'exécution d'un programme, ce qui permet de chercher plus facilement les erreurs éventuelles (les bugs !)

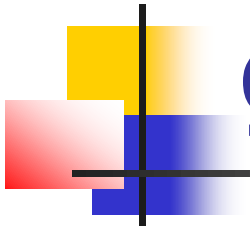
- Il permet essentiellement :

  - D'avancer pas à pas dans l'exécution du programme
  - De créer des points d'arrêt
  - De lire (et de modifier) le contenu des variables

Débogueurs

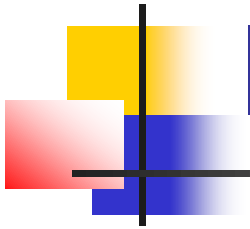

Dev-C++ 4.9.9.2
Fichier Edition Recherche Vue Projet Exécuter Debug Outils CVS Fenêtre Aide
Nouveau Insérer Basculer Aller à...
Projet Classes Debug
&rayon_planete = (float *)
rayon_planete = 6378
surface_planete = 51092
planetes_cours.c
#include <stdlib.h>
C:\Francis\Unice\Annee_2010_2011\L1SM\Cours\Projets\Cours1\Planetes\planetes_cours.exe
Entrez le rayon de la planete (en km) : 6378
float rayon_planete;
float surface_planete;
float volume_planete;
printf("Entrez le rayon de la planete (en km) : ");
scanf("%f", &rayon_planete);
surface_planete = 4.0 * PI * rayon_planete * rayon_planete;
volume_planete = (4.0/3.0) * PI * rayon_planete * rayon_planete * rayon_plan

Générateurs d'interfaces graphiques



- La plupart des applications professionnelles ont une interface graphique évoluée comprenant boutons, cases à cocher, menu déroulant, etc.
- Certains langages disposent d'instructions permettant la création de ces interfaces
- Certaines plateformes de développement proposent même des outils graphiques et de gestion de code qui permettent au programmeur non expérimenté de développer facilement ce type d'interfaces (citons delphi, visual basic, visual C++, jbuilder, etc.)

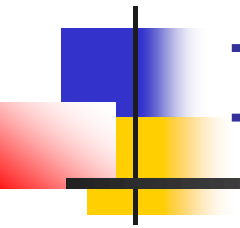
Champs d'applications des langages



- Un langage est plus ou moins bien adapté à un domaine donné

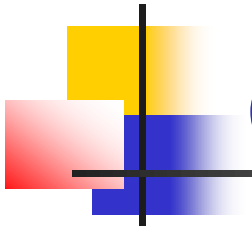
- Certains langages sont relativement spécialisés
  (Perl : traitement des chaînes de caractères, JavaScript : écriture de pages web interactives, etc.)

- D'autres plus ou moins généralistes (C, Pascal, C++, Objective C, Java, Python, etc.)



2. Programmation impérative. Introduction au langage C

2.1 Les différents paradigmes de programmation



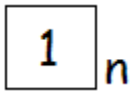
- Il existe plusieurs techniques de programmation :
  - Impérative
  - Fonctionnelle
  - Logique
  - Orientée objets
  - Parallèle
- Le langage que nous allons étudier, le langage C, appartient à la catégorie des langages de programmation impérative

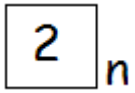
La programmation impérative

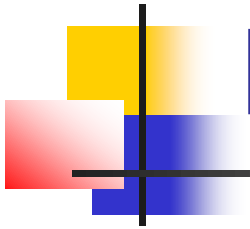
- Dans ce type de programmation, on manipule explicitement la mémoire grâce à des instructions de haut niveau

- Exemple :

```
1.   int n = 1; // on initialise une variable nommée n à 1
2.    n = n + 1; // on incrémente n de 1
```

- État de la mémoire :
  - Après 1 : 

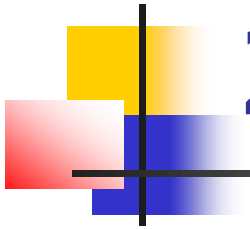
  - Après 2 : 



La programmation impérative

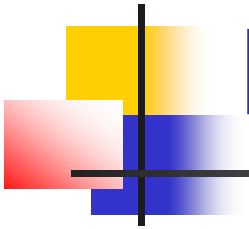
- Langages emblématiques : Pascal, C, Basic, Fortran

- Remarque : dans tous les cas, au niveau de la machine, l'exécution d'un programme revient toujours à des manipulations sur la mémoire



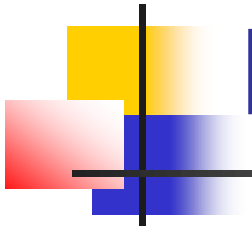
2.2 Le langage C, introduction

- Historique
- Compilateurs, environnements de programmation



Historique du langage C

- Créé en 1972 par Denis Ritchie (2011†) (Bell Laboratories)
- Au départ, pour développer le système d'exploitation Unix
- Aujourd'hui utilisé par une large communauté de programmeurs



Historique du langage C

- Première définition rigoureuse en 1978 «The C programming Language » par Kerniguan & Ritchie

- Aujourd'hui le langage est normalisé, on parle de « C ANSI »

Compilateurs, environnements de programmation

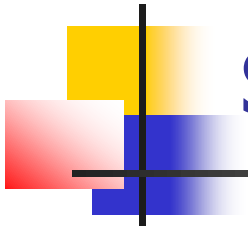
- Il existe des compilateurs et/ou environnements de programmation pour tous les systèmes d'exploitation.

  Par exemple :

  - Sous Windows : Dev-C++ (éditeur + compilateur + débogueur)
  - Sous Linux : gcc (éditeur emacs + débogueur ddd)

- Notons que tous les environnements de programmations pour C++ et Objective C sont aussi des environnements de programmation pour C. Nous utiliserons d'ailleurs Dev-C++ qui comme son nom l'indique est un environnement de programmation pour C++

2.3 Etude d'un programme simple en C

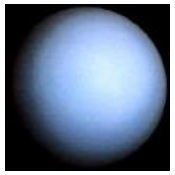
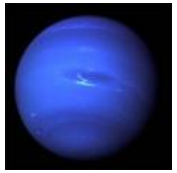


- Objet du programme : calcul de la surface et du volume des planètes du système Solaire
- Rappels des formules
- Listing du programme
- Analyse détaillée du code

Le système Solaire

Planète	**Image**	**Rayon** (km)
Mercure		2439
Vénus		6050
Terre		6378
Mars		3387

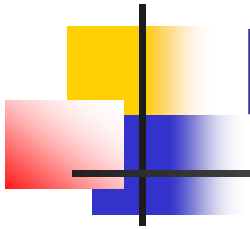
Le système Solaire

Jupiter		71400
Saturne		60000
Uranus		26150
Neptune		24300

Surface et volume d'une sphère en fonction du rayon

$$S = 4\pi R^2$$

$$V = \frac{4}{3}\pi R^3$$



Le programme « planetes »

- Un programme se développe dans un fichier
- On développera nos programmes dans un fichier d'extension .c
- Le programme qui calcule l'aire et le volume des planètes à partir de leur rayon est développé dans le fichier planetes.c
- Règles de bonne programmation :
  - On donnera au fichier un nom qui rappelle la fonction du programme qui y est développé
  - On n'utilise pas les accents (planetes.c au lieu de planètes.c) ni les espaces

Le fichier planetes.c

```
#include <stdio.h>

const float PI = 3.14;

int main() {
  float rayon_planete; // le rayon de la planete
  float surface_planete; // sa surface
  float volume_planete; // son volume

  /* entree de la donnee rayon */
  printf("Entrez le rayon de la planete (en km) : ");
  scanf("%f", &rayon_planete);

  /* calcul :
      de la surface, du volume */
  surface_planete = 4.0 * PI * rayon_planete * rayon_planete;
  volume_planete = (4.0/3.0) * PI * rayon_planete * rayon_planete * rayon_planete;

  // impression des resultats
  printf("%s%f%s\n", "Surface de la planete : ", surface_planete, " km2");
  printf("%s%f%s\n", "Volume de la planete : ", volume_planete, " km3");

  return 0;
}
```

Analyse du programme « planetes »

```
#include <stdio.h>
```

- Demande d'inclusion du fichier **`stdio.h`** (« standard input ouput »). L'inclusion de ce fichier est nécessaire pour pouvoir utiliser les fonctions d'entrée-sortie

Analyse du programme « planetes »

```
const float PI = 3.14;
```

- Déclaration d'une constante symbolique **`PI`** égale à 3.14 de type **`float`** (nombre réel)
- Dans le code la référence à **`PI`** plutôt qu'à la valeur 3.14 permet une modification facile de celle-ci (une ligne à modifier contre autant de lignes où apparaîtrait une valeur « en dur »)
- Règle de bonne programmation : utiliser des constantes symboliques pour les valeurs qui changent rarement (constantes physiques, économiques (ex : taux de tva), etc.)

Analyse du programme « planetes »

```
int main() {
```

- **`main`** est la fonction principale en C. Elle est exécutée quand on lance le programme
- Une fonction passe une liste d'arguments (ici la liste est vide) et renvoie une valeur (ici un entier -**`int`** comme integer-)
- Le code de la fonction est donné entre l'accolade ouvrante "**`{`**" et la fermante correspondante "**`}`**"

Analyse du programme « planetes »

```
float rayon_planete; // le rayon de la planete
float surface_planete; // sa surface
float volume_planete; // son volume
```

- Déclaration de trois variables qui mémorisent le rayon, la surface et le volume de la planète. Un commentaire indique ce que mémorisent les variables

- Règles de bonne programmation :
  - **Donner des noms aux variables qui correspondent à l'information qu'elles mémorisent**
  - **Commenter le programme pour qu'il soit plus compréhensible**

- En C on a droit aux commentaires sur une ligne avec « `//` » et aux commentaires sur plusieurs lignes encadrés par « `/* */` ». Les commentaires sont ignorés par le compilateur. Ils ne s'adressent qu'aux lecteurs du programme

Analyse du programme « planetes »

```
/* entree de la donnee rayon */
printf("Entrez le rayon de la planete (en km) : ");
scanf("%f", &rayon_planete);
```

- L'utilisateur doit entrer le rayon de la planète au clavier. Une phrase à l'écran lui indique la donnée attendue (fonction **`printf`**).
- La donnée est alors tapée sur le clavier est récupérée dans la variable **`rayon_planete`** grâce à la fonction **`scanf`** (le nom de la variable doit être précédé du caractère **`&`**)
- Le premier argument de la fonction **`scanf`**, "**`%f`**", indique que la donnée attendue est un nombre réel

Analyse du programme « planetes »

```
/* calcul :
    de la surface, du volume */
surface_planete = 4.0 * PI * rayon_planete *
                  rayon_planete;
volume_planete = (4.0/3.0) * PI * rayon_planete *
                 rayon_planete * rayon_planete;
```

- Calcul de la surface et du volume et mémorisation dans les variables correspondantes
- `*` est le signe pour la multiplication et `/` est le signe pour la division

Analyse du programme « planetes »

```
// impression des resultats
printf("%s%f%s\n", "Surface de la planete : ",
                        surface_planete, "km2");
printf("%s%f%s\n", "Volume de la planete : ",
                                volume_planete, "km3");
```

- Impression à l'écran du contenu des variables ***surface_planete*** et ***volume_planete*** avec le commentaire correspondant

- Le premier argument de **printf** ("**%s%f%s\n**") indique la nature des données à imprimer (paramètres suivants). **%s** indique l'impression d'une chaîne de caractères, **%f** d'un nombre réel, **\n** indique un saut de ligne après impression

Analyse du programme « planetes »

```
 return 0;
}
```

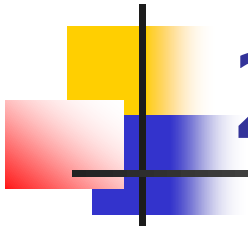
- La fonction **`main`** doit renvoyer un entier. Ici 0 (code correspondant en général à une exécution sans erreur)
- On ferme l'accolade correspondant à la fonction **`main`**
- Règle de bonne programmation : les accolades ouvrantes et fermantes se correspondant doivent être au même niveau d'indentation. Le code d'une fonction est en retrait par rapport aux accolades correspondantes

2.4 Organisation du code en C

- En C, le code se répartit en deux types de fichiers : les .h (en-tête) et les .c (code source)

- Les fichiers .h contiennent uniquement du code déclaratif. Ce code est utilisable dans le fichier .c qui l'inclut (**`#include`**). Par exemple les fonctions d'entrées/sorties **`printf`** et **`scanf`** sont utilisables dans planetes.c car le fichier planetes.c inclut le fichier stdio.h qui contient les définitions de ces fonctions.

- Avant la compilation, le *pré-processeur* expanse dans les fichiers .c le code des fichiers .h inclus



2.5 Compilation, exécution

- Le programme développé dans le fichier planetes.c est écrit en langage C. Pour que ce programme devienne effectif, il faut le transformer en un programme exécutable compréhensible par le processeur

- On compile le programme planetes.c en utilisant l'option « Compiler » du menu déroulant « Exécuter » de Dev-C++

- On exécute le programme exécutable généré en utilisant l'option « Exécute » du menu déroulant «Exécuter» de Dev-C++

- L'exécutable généré, de nom « planetes.exe », se trouve dans le même répertoire que le fichier planete.c

Exécution du programme

- Voici le résultat du programme pour la planète Terre :