

# ソフトウェア概論 A/B

-- Compile 環境/関数 --

数学科 栗野 俊一 / 渡辺 俊一

2016/04/22 ソフトウェア概

# 伝言

---

## 私語は慎むように !!

### □ 色々なお知らせについて

- 栗野の Web Page に注意する事

<http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino>

### □ 講義開始前に済ませておく事

- PC の電源を入れる
- ネットワークに接続しておく事
- 今日の資料に目を通しておく事

### □ 講義前の注意

- 講義前は、栗野は準備で忙しいので TA を捕まえてください

### □ やる気のある方へ

- 今日の資料は、すでに上っています
  - ▷ どんどん、先に進んでかまいません

# 前回(2016/04/15)の復習

---

## □ 前回(2016/04/15)の内容

- 講義の進め方 (相変わらず、栗野は小言が多い)
  - ▶ 他の人の学習の邪魔はしてはならない
  - ▶ 講義中は喋るな!!
- 無線 LAN の設定
- 仮想環境上の ubuntu の実行
- SCCM のインストール

## □ 講義内容

- Web で公開されているので、復習する

## □ 演習課題

- 毎回提出する事
- 期限が遅れても、とにかく、「全部」提出する

# お知らせ

---

## □ 出席パスワード : 20160422

○ 出席は **CST Portal** で取りますが、成績には(残念ながら?)無関係です

▶ 単位を取るならば、課題を出しましょう

## □ 本日の予定

○ **Skype** のグループチャット

○ **Compile** の仕方を覚える

○ プログラムを書いてみよう

▶ Hello, World

▶ 「関数呼出し」を並べてみよう

▶ 自分で新しい「関数を作って」みよう

## □ 本日の目標

○ 講議の進行方針を把握する

○ 演習

▶ C 開発環境 (ubuntu) の利用方法

▶ プログラムの作成と実行

▶ 課題の提出

# 前回 (2016/04/15) の課題

---

## □ 前回 (2016/04/15) の課題

### ○ 次の C Program ファイルを作成し提出しなさい

▶ 今回は提出先は二つある ( CST Portal : 去年と同じ / e-mail )

### ○ CST Portal

▶ ファイル名 : 20160415-01-YYYY.c (YYYY は学生番号)

▶ 内容 : 「Hello, 自分の名前」を出力する C 言語のプログラム

▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

### ○ NU-AppsG のメール機能を利用して課題を提出する

▶ 宛先: kurino.shunichi@nihon-u.ac.jp

▶ 表題:「ソフトウェア概論:20160415-01-YYYY」

▶ 内容: 自分の学籍番号と名前

▶ 添付: 20160415-01-YYYY.c (YYYY は学生番号)

# 本日の課題 (2016/04/22)

---

## □ 今回 (2016/04/22) の課題

### ○ 次の C Program ファイルを作成し提出しなさい

- ▶ ファイル名 : 20160422-01-YYYY.c (YYYY は学生番号)
- ▶ 内容 : 「Hello, 自分の名前」を100回以上出力する C 言語のプログラム
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

# Skype のグループチャット

---

## □ Skype のグループチャット

- 「数学科 2016 年度ソフトウェア概論」を講議用に用いる / 全員参加 (ブックマーク)

## □ 参加希望の人

- kurino-2015-math-cst-nihon-u に「数学科 2016 年度ソフトウェア概論登録希望」のメッセージを送ってください
- 昨年コンピュータ概論を受けていない人は、コンタクトを送ってください
  - ▶ メッセージに「数学科 2016 年度ソフトウェア概論登録希望」を入れる事

# プログラミング

---

## □ プログラムとは

- 計算機への指示(作業手順)を記述したもの
  - ▶ 計算機はプログラムに記述されている内容に従って動く
- プログラムはファイルの中に記述される

## □ プログラムの実行

- 計算機がプログラムの内容に従って動く事

## □ プログラムを実行させる

- プログラムが記述されているファイルを指定して実行させる事

## □ プログラミングとは

- プログラムを作成する事
  - ▶ やりたい事を記述するためにその手順を考える
  - ▶ その手順をファイルに記録する



# プログラム vs 料理

---

- プログラム：作業の手順
  - レシピ：料理の作り方
- 計算機：プログラムを実行する
  - 普通の料理人 (主婦)：レシピに従って料理をする
- プログラマ：プログラムを作成する人
  - 創作料理家：レシピを作る人
- プログラムの実行
  - 料理

# プログラミング言語

---

## □ プログラミング言語とは

- プログラムを記述するための専用の言語 (<反> 自然言語)
  - ▷ C 言語, java, perl, etc..
- 計算機向け
  - ▷ 厳密で簡潔
  - ▷ 融通がきかない

## □ 機械語と高級言語

- 機械語：計算機が直接実行可能な形式
- 高級言語：多少、人間に判りやすい形式
  - ▷ 計算機で実行するには、翻訳 or 通訳が必要

## □ コンパイラ (言語処理系の形式の一つ)

- 高級言語で記述されたプログラムを機械語に翻訳する
  - ▷ ソースプログラム：高級言語で記述されたプログラム
  - ▷ オブジェクトプログラム：機械語で記述されたプログラム
- インタープリターは通訳をする

# C 言語でコンパイル

---

- C 言語 ( c ファイル ) : コンパイル言語
  - C 言語で作られたプログラムは、そのままでは、実行できない
    - ▶ 実行するには「コンパイル」が必要
- リンク
  - オブジェクトプログラム ( obj ファイル ) だけでは動かない
    - ▶ 補助のプログラム ( ライブラリ : lib ファイル ) も必要
- 実行ファイル ( exe ファイル )
  - オブジェクトプログラムとライブラリを繋げたもの
    - ▶ リンクによって行われる
- cc コマンド
  - コンパイルと同時にリンクもする
    - ▶ オブジェクトファイルと実行ファイルの両方が作られる

# C 言語で Hello, World

---

## □ Hello, World プログラム (sample-001.c)

- 「Hello, World[改行]」
- 短いながら完全なプログラムで、意味がある
  - ▶プログラム作成の土台

## □しばらくの「プログラミング」学習

- とりあえず「動けば」よい
  - ▶「理解」は後からもう一度やる
  - ▶細かい話は後回し
- 差分プログラミング
  - ▶結果を少しずつ作って行く
  - ▶すでに動く事が解っているプログラムの一部を変更する

# printf 関数

---

## □ printf 関数

- 「printf ( 引数文字列 );」の形で呼出す
  - ▶ 引数文字列が画面に表示されるという「副作用」がある
  - ▶ 文字列はダブルクォーテーション(")で挟まれている
  - ▶ 「\n」は「改行」の意味

## □ 色々な疑問

- 「関数」って.. ?
- 「引数」って.. ?
- 「呼出す」って.. ?
- 「副作用」って.. ?
  - ▶ ここでは、そう「呼ぶ」のだと思う事しよう
- 他にも「#include」とか「main」とか「{」とか「}」って？
  - ▶ ここでは、とりあえず「オマジナイ」と思う事にする

# 順接

---

## □ 順接

- 命令を並べる事

- ▶ 「関数呼び出し」も命令

- 並べた順に「実行」される

- ▶ printf は文字列を出力する関数

- ▶ printf の呼出しを並べると文字列の出力が並ぶ

## □ 単純なプログラミング

- 計算機にさせたい命令列を、その実行順に並べる

- ▶ 文字列を並べたければ、文字列を出力する命令を並べればよい

- ▶ 一回書けば、一度してくれる

- 一度プログラムを書けば、何度でも実行してくれる

# 関数

---

## □ 関数

### ○ 命令列に名前を付けた物

- ▶ 名前を指定して「呼出す」だけで、その命令列が実行できる

## □ 関数定義

### ○ 命令列を「{」と「}」で囲って、それに関数名を付ける

- ▶ この命令列を関数の「本体」と呼ぶ
- ▶ 「void」とか「()」の意味は、今回は説明しない

## □ 関数呼び出し

### ○ 関数名を指定する事により、関数の本体の命令列が実行できる

- ▶ 「()」の意味今回は説明しない

## □ 関数の効用

- 「名前が付く」のでプログラムが理解り易くなる
- 関数を利用するとプログラムが短くできる

# 今回のまとめ ( 1 : プログラミングとは )

---

## □ プログラミングとは : プログラムを作る事

- プログラムとは : 計算機への指示(作業手順)を記述したもの

- ▶ この講義では C 言語で記述された Text ファイル (\*.c) を作成する

## □ コンパイルとは

- C 言語で記述されたプログラムを実行が出来る形に翻訳する事

- ▶ 実行ファイル (\*.exe) が出来る

- チェックポイント : コンパイルの手順は憶えたか ?

## □ C 言語とは

- 数あるプログラム記述言語の一つ

- ▶ 詳しくはこの講義を最後までしっかり聞こう

## □ プログラミングを学ぶには

- 習うより慣れる : とにかく、手を動かさせ

- ▶ 読書百遍、意、自ら通ず



# 今回のまとめ ( 2 : C 言語 )

---

## □ 「Hello, World」プログラムとは

- 単純だが、完全なプログラム：他のプログラムを作成する土台となる

## □ 関数とは：命令の集まりに名前をつけたもの

- 名前を指定して、その命令列(機能)を呼び出す事ができる

- ▶ cf. printf 関数：メッセージを画面に出力する

- 自分で作成する事もできる

- ▶ cf. main 関数：自分が作成するプログラムの開始地点

## □ 順接とは

- 命令を順に並べる事。これにより、その命令をその並べた順で実行する事ができる

- ▶ 操作の「手順」を与えるという、最も基本的なプログラムの記述方法

## □ 命令とは

- C 言語の中で「何か(計算)」をする記述表現

- ▶ 今回は「関数呼び出し」しかやっていない

- ▶ 「関数作成」と「関数呼び出し」は、プログラムの基本構造

# ubuntu と windows のファイル共有

---

- ubuntu と windows のファイル共有
  - windows の C:\usr\c と ubuntu の ~/c が共有されている
    - ▶ 一方を変更すると、他方も変更される
- 作業の分担
  - ファイルの作成は、windows で C:\usr\c 以下に行う
  - コンパイル実行は、ubuntu の ~/c 以下で行う

# 共有の確認と修正

---

## □[共有の確認]

- 共有が上手くいっているかどうかを次のようにして確認する
- ubuntu** で、次のコマンドを実行して、以下の表示ができればよい
  - ▶コマンド : `mount | grep soft/c`
  - ▶出力 : `.host:/c on /home/soft/c type vmhgfs (rw,ttl=1)`

## □[共有の設定] 共有できていない場合は、次の二つのステップを取る

- ステップ 1 : コマンドを実行して **ubuntu** の準備ができているかどうかを確認
  - ▶コマンド : `mount | grep /mnt/hgfs`
  - ▶出力 : `.host:/ on /mnt/hgfs type vmhgfs (rw,ttl=1)`
  - ▶表示されなければ、暫く待ってからもう一度 (それで駄目なら先生を呼ぶ)
- ステップ 2 : 次のコマンドを実行する
  - ▶コマンド : `sudo /etc/rc.local`
  - ▶もし、これで、もう一度[共有の確認]をして駄目なら先生を呼ぶ

# 環境の更新

---

## □ Graphics (OpenGL) のインストール

- 次のコマンドを実行して、Graphics Programming ができるようにする

- ▶ `sudo apt-get install freeglut3 freeglut3-dev`

- ポイント

- ▶ パスワードを聞かれたら「soft」と入力

- ▶ 「続行しますか？ [Y/n]」と聞かれたら [Enter] を押す

# ファイルのダウンロードと配置

---

## □ 教材のダウンロードと配置

- 本日(2016/04/22)のページから、次の三つのファイルをダウンロードする

- ▶ 20160422.zip, lib.zip, include.zip

- ▶ 保存先は、C:\usr\c にする

- ダウンロードしたら、それぞれ展開する

## □ 設定ファイル(/etc/rc.local)の更新

- ubuntu の方で、次のコマンドを実行して、設定ファイルを更新する

- ▶ \$ sudo ~/c/20160422/update/update.sh

## □ Graphics (OpenGL) の確認

- ubuntu の方で、次のコマンドを実行する

- ▶ \$ cd ~/c/20160422/ogltest

- ▶ \$ make test

- 画面に白い四角が表示されれば OK

- ▶ マウスで適当な所をクリックすると、クルクル回る

- ▶ ウィンドウの左上の「(×)」ボタンをクリックすれば閉じる

# Ubuntu を最新の状態に

---

## □ Ubuntu を最新の状態にするには

- ネットワークに接続した状態で、次の二つのコマンドを実行すると、更新される

- ▶ `sudo apt-get update`

- ▶ `sudo apt-get upgrade`

- 初回は、時間が掛るので、暇な時にする(講議中はさける)

- ▶ 講議のある日の前日の夜にすると良い