

ソフトウェア概論 A/B

-- 引数付き関数と分割コンパイル --

数学科 栗野 俊一 / 渡辺 俊一

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く
禁じます

伝言

- 出席パスワード : 20210423
- 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>
- 対面講義の受講者へ
 - 座れる席は QR コードが貼られている席だけです
 - ▷ 携帯電話で、位置情報を送ってください
- 廊下側の一列は遅刻者専用です(早く来た人は座らない)
- 講義開始前に済ませておく事
 - PC の電源を入れておく
 - ネットワークに接続しておく
 - 今日の資料に目を通しておく
- 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

対面講義の方へ

ソフトウェア概論 A/B (2021/04/23)

対面講義の方へ

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

対面講義の方へ

□ 出席

- 出席は、入口の所にある出席管理システムに学生証をタッチする

□ 席は、QR コードが書いてある所

- QR コードの扱いは、教室の前にあるポスターに従う

□ 教室内では

- 静かに !!
- PC をネットワークに接続する (CST-network がつかえます)
 - ▶ SSID: CST-network
 - ▶ PW: 講義中の黒板に記載しておきます
- PC に電源をいれ、本日の資料に目を通す

前回(2021/04/16)のまとめ

ソフトウェア概論 A/B (2021/04/23)

前回(2021/04/16)のまとめ

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

前回(2021/04/16)の復習

□ 前回(2021/04/16)の内容

○ プログラミング

- ▷ プログラム
- ▷ プログラミング
- ▷ プログラミング言語 (C 言語)
- ▷ コンパイル

○ 「Hello, World」プログラム

□ 実習内容

- Ubuntu でのコンパイル
- 課題の提出

今回(2021/04/23)の予定と課題

ソフトウェア概論 A/B (2021/04/23)

今回(2021/04/23)の予定と課題

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

今回(2021/04/23)の予定

□ 出席パスワード : 20210423

○ 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

▶ 単位を取りたいならば、課題を提出しましょう

□ 本日の予定

○ 引数付き関数

○ 分割コンパイル

○ make と makefile の役割

□ 本日の目標

○ 引数付き関数

▶ プログラミング : 機能の再利用

▶ C 言語 : 引数付き関数の定義の仕方と呼出し方

○ 分割コンパイル

▶ プログラミング : 関数の再利用

▶ C 言語 : extern 宣言

○ make と makefile

▶ プログラミング : 作業の自動化

▶ 操作 : make の利用法

先週 (2021/04/16) の課題

□ 先週 (2021/04/16) の課題

○ 課題 20210416-01:

- ▶ ファイル名 : 20210416-01-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 「Hello, 自分の名前」を 3 回出力する C 言語のプログラム

○ 課題 20210416-02:

- ▶ ファイル名 : 20210416-02-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 「Hello, 自分の名前」を表示する関数を作成しなさい

○ 課題 20210416-03:

- ▶ ファイル名 : 20210416-03-QQQQ.c (QQQQ 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

□ 提出するファイル形式

- 全てテキストファイル(C 言語プログラムファイル)
- 提出先は CST Portal II

今週 (2021/04/23) の課題

□ 今週 (2021/04/23) の課題

○ 課題 20210423-01:

- ▶ ファイル名 : 20210423-01-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 引数で指定された名前に、三度、「こんにちは」をする関数の作成
- ▶ ファイル名 : 20210423-01-QQQQ.c (QQQQ は学生番号)

○ 課題 20210423-02:

- ▶ ファイル名 : 20210423-02-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 童謡の歌詞を出力するプログラムを作成しなさい
- ▶ 可能な限り引数付きの関数で..
- ▶ 曲は何でもよい

□ 提出するファイル形式

- 全てテキストファイル(C 言語プログラムファイル)
- 提出先は CST Portal II

引数付き関数

ソフトウェア概論 A/B (2021/04/23)

引数付き関数

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

引数付き関数

□ 引数付き関数

- 関数本体の一部を「変数(へんすう)」にしたもの

- ▶ 「変数」の内容を、後(実行時)に決める事により、振舞(ふるまい)を変更できる

- 引数(ひきすう)

- ▶ 仮引数(かりひきすう) : 関数の本体に現れる「変数」の宣言 (関数定義)

- ▶ 実引数(じつひきすう) : 「変数」の内容 (関数呼出し)

□ 引数付き関数の効用 (本質的には「再利用[コピペ]」が目的)

- 一つの関数定義で(引数の値を変更する事により..)複数の関数として利用できる

関数 (再録)

□ 関数

- 「命令列」に「名前」を付けたモノ

- ▶ 名前を指定して「呼び出す」だけで、その命令列が実行できる

□ 関数定義：新しい関数を定義し、プログラム内で利用出来るようにする

- 「命令列」を「{」と「}」で囲って、それに「名前」を付ける

- ▶ この「命令列」を関数の「本体」と呼び、「名前」を「関数名」と呼ぶ

- ▶ 「void」とか「()」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

- ▶ 「()」は今回は説明しない

□ 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

- ▶ 「命令列」に対し、「意味のある名前」を付ければ、意味が一目で解る

- ▶ 同じ名前の関数呼び出しは、同じ命令の実行である事が保証される

- 関数を利用すると、プログラムが短くなる

関数の作り方 (その 1)

□ 関数の作り方(引数がない場合)

- 関数の「名前」を決める

- ▷ cf. subfunc

- どの部分の「命令列」を関数にするかを決める

- 関数にしたい「命令列」を切出し、main の外に出し、「ブロック」にする

- ▷ 「命令列」を「ブロック」にするには '}' と '{' で囲めば良い

- ▷ 名前を付ける (cf. void subfunc())

□ 関数の呼出し方(引数がない場合)

- 元々の命令が在った所に「関数呼び出し」を置く

- ▷ cf. subfunc();

関数の作り方 (その 2)

□ 関数の作り方(引数がある場合)

- 二つの(引数のない)関数が、ほぼ同じ形をしている時

 - ▶ 一つの引数付き関数にまとめる事が可能(かもしれない)

- 違う部分を、変数(x,y,z..) に置き換える

 - ▶ 置き換えた結果、同じ形をしていれば、それは一つの関数にまとめられる

- 関数定義の関数名の後ろにある「()」の中に変数名(仮引数名)を書き、その変数の前「char *」を追加

 - ▶ 引数付き関数が定義される

□ 関数の呼出し方(引数がある場合)

- 元々の命令が在った所に「関数呼び出し」を置く

 - ▶ 後ろの「()」の中に、変数の値(実引数)を書く

関数呼び出しの挙動

- 関数呼び出しは、次の様に振舞う
 - 関数呼び出しのある場所から関数本体の先頭に行く
 - 関数の本体を実行する
 - 関数呼び出しのあった場所の次に戻る
- 関数の「引数(ひきすう)」とは
 - 関数の振舞いを変更するための「情報 (パラメータ)」
 - ▶ 同じ関数でも「引数」が異れば異なる「振舞い」をする
- 引数付きの関数の呼び出し
 - 関数の中の「仮引数変数」に、「実引数の値」が入っている

分割コンパイル

ソフトウェア概論 A/B (2021/04/23)

分割コンパイル

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

分割コンパイル

□ C 言語で記述されたプログラムの構造

○ main 関数が必ず必要

▶ 他の関数は main 関数から呼び出される

○ 関数の定義

▶ ソースファイル (*.c) の中に記述して、コンパイルする

▶ 同じファイル内である必要はない

□ 分割コンパイル

○ 関数を別のファイルで定義し、個々にコンパイルする事

▶ 後でリンクにより一つの実行ファイルにまとめる

○ ファイル間で情報交換が必要

▶ extern 宣言が必要

make と makefile

ソフトウェア概論 A/B (2021/04/23)

make と makefile

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

make と makefile

□ make とは

- 様々な操作を自動的に行ってくれるコマンド
 - ▶ 「ファイルを『作成』する」ので「make (作る)」という名前

□ Makefile

- 様々な「ファイルの作成方法」を記述したファイル
 - ▶ make は Makefile を読み込んで、ファイルを作成する
- cf. 料理で考えると、Makefile がレシピで、make が料理人、ファイルが料理

□ Makefile の記述方法

- 基本の形式は次の形
 - <<作りたいファイル>> : <<材料となるファイル>> ...
 - <<TAB コード>><<作りために実行する命令>>

- 例: (hello.exe を作る Makefile)

```
hello.o : hello.c # hello.c を材料に hello.o を作る
    cc -c hello.c # hello.o を作るには「cc -c hello.c」とする
hello.exe : hello.o # hello.o を材料に hello.exe を作る
    cc -o hello.exe hello.o
```

make と分割コンパイル

- 分割コンパイルは複数のファイル进行处理
 - 作業も面倒だし、間違いも起きやすい
 - ▶ コンパイルの手順を記述してコンピュータにやらせちゃおう
- **Makefile**
 - コンパイルの手順などを記述したファイル
- **make**
 - **Makefile** を読んで、コンパイルを自動的に行ってくれる
- **make -f 「ファイル名」**
 - **Makefile** の代わりに「ファイル名」を利用する
- **make 「ターゲット名」**
 - 先頭の「ターゲット」ではなく、指定した「ターゲット」を作る

Makefile の色々

□ Makefile には色々な便利な機能がある

○ 変数 : 文字列に名前を付ける事ができる

▶ 変数(の値の)定義 : 「変数名」=「値」とすると変数に値の内容を割り当てる

▶ 変数(の値の)参照 : 「\$(変数名)」と書くと「変数の

○ 例: (hello.exe を作る Makefile)

```
BASE=hello      # 変数 BASE に hello という値を割り当てる
```

```
${BASE}.o : ${BASE}.c # ${BASE} はどれも hello に置き換わる
```

```
    cc -c ${BASE}.c # 結果的に「cc -c hello.c」と同じ
```

```
${BASE}.exe : ${BASE}.o
```

```
    cc -o ${BASE}.exe ${BASE}.o
```

Top-Down と Bottom-Up

ソフトウェア概論 A/B (2021/04/23)

Top-Down と Bottom-Up

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます

プログラミング：積み木の積み上げ

□ プログラムを作成する行為は..

○ 沢山の部品を組み上げて、製品を完成させる行為

▶ ライブラリは、「部品」に相当

▶ 「大きな部品」があれば、「ちょっと、飾るだけで完成」する

▶ cf. カレーのレトルトパックがあれば、御飯を炊くだけでカレーライス完成

○ 「大きな部品」は高機能

▶ 扱いが難しい (API をよく理解する必要がある)

□ Top-Down と Bottom-Up

○ Top-Down：上から下に向けて進める

▶ 目的を分割して分かり易くする (設計)

○ Bottom-Up：下から上に向けて進める

▶ 機能を組み上げて、目的を実現してゆく (コーディング)

○ Top-Down に考えて、Bottom-Up に作成する

おしまい

ソフトウェア概論 A/B (2021/04/23)

おしまい

講義内容の静止画・動画での撮影、及び SNS 等への転載を固く禁じます