応用プログラミング

自由席です

応用プログラミング

第1回　C言語の基本文法・言語処理系

メディア情報コース
平居　悠（ひらい　ゆたか）

自己紹介

平居　悠　（ひらい　ゆたか）

メディア情報コース　講師

居室：H-2研究室

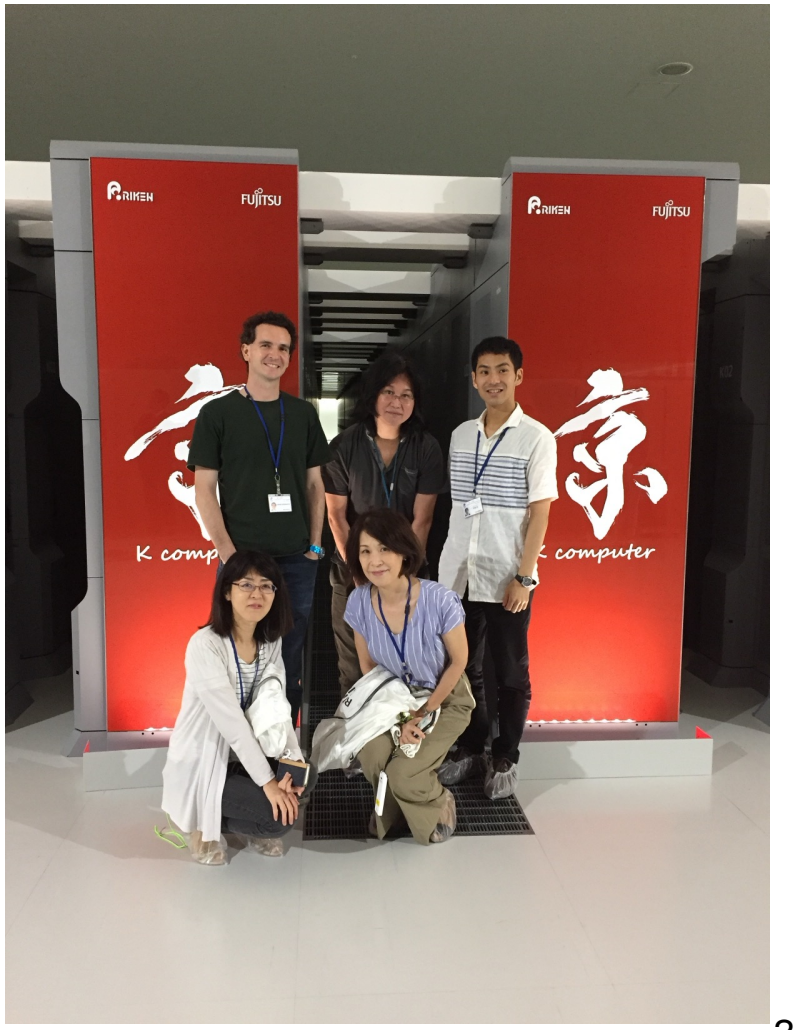
E-mail: yutaka.hirai@koeki-u.ac.jp

出身：埼玉県

専門：天文学、数値シミュレーション

学位：博士（理学）東京大学 2018年

職歴：理化学研究所、東北大学、
アメリカ・ノートルダム大学



データサイエンス・AI教育プログラム

データサイエンスやAIに関する基礎的な知識と技術、及びその知識や技術を他の科目や学修で応用する能力

MDASH Literacy 数理・データサイエンス・AI 教育プログラム認定制度 リテラシーレベル

MDASH Advanced Literacy 数理・データサイエンス・AI 教育プログラム認定制度 応用基礎レベル

「データサイエンス・AI教育プログラム」の流れ （※8単位以上修得）

1年次秋学期	2年次春学期	2年次秋学期
データリテラシー	基礎プログラミングⅠ	基礎プログラミングⅡ

選択必修科目：「AIと社会」「セキュリティ論」「日経講座：デジタル社会論」
選択科目：「数学a」「数学b」「統計学a」「統計学b」「データサイエンス入門b」「数値情報処理b」「画像情報処理」「データ分析手法b」「経営工学a」「経営工学b」「データベース論」「データベース演習」「データ構造とアルゴリズム」「応用プログラミング」「機械学習入門a」「機械学習入門b」

授業概要

C言語は、現在主流な多くのプログラミング言語の基礎となっており、現在でも広く使われている。C言語を学ぶことで、プログラミング言語やコンピュータの動作を理解することができる。本講義では、C言語を通じてコンピュータの根幹部分を理解し、**自分に必要な仕事をこなすプログラムを自由に設計できるようになる**ことを目指す。

到達目標

- C言語を通じてコンピュータの根幹部分を理解する。
- 目的に応じたプログラムを自由に設計できる。

授業計画

第１回　C言語の基本文法・言語処理系
第２回　数学的計算、配列、文字列
第３回　型変換、入出力処理
第４回　デバッグ、関数定義、変数の分離
第５回　プロトタイプ宣言、分割コンパイル
第６回　配列の受け渡し、ポインタ
第７回　まとめ、期末試験

成績評価

平常課題（６割）

期末試験（４割）

授業ページ

学生専用サーバ→平居→応用プログラミング

学内：http://roy.e.koeki-u.ac.jp/
学外：https://www.koeki-prj.org/roy/

質問等

- s4 「応用プログラミング」 https://s4.koeki-prj.org/s4-world-2026au.cgi?grp+157
- メール (yutaka.hirai@koeki-u.ac.jp)
- オフィスアワー（H2研究室木曜2限）

参考書


NEKO series
猫でもわかる
C言語
プログラミング
第3版
粂井康孝 著
Yasutaka Kumei
SoftBank
Creative


新・明解
柴田望洋
BohYoh Shibata
C言語
入門編
4色刷
Textbook MEIKAI
by Dr.BohYoh Shibata
SB Creative


新・標準
プログラマーズ
ライブラリ
C言語
ポインタ
完全制覇
前橋和弥
Kazuya Maebashi
Cはなんでこんな言語に「なっちゃった」のか。そもそもこの悪名高いポインタとは何か。
初心者が必ずひっかかる、配列とポインタのまぎらわしい文法とは。
Cはメモリを実際にどんなふうに使うのか。Cの宣言は英語で読め。
ポインタの真の使い方は。Cの文法を深く知ることで見えてくること納得できること。
技術評論社

授業計画と参考書の対応

第１回　C言語の基本文法・言語処理系	(明) 1, 3, 4章、(猫)1, 2, 5章
第２回　数学的計算、配列、文字列	(明) 2, 5, 9章、(猫)4章
第３回　型変換、入出力処理	(明) 7, 13章、(猫)3, 10章
第４回　デバッグ、関数定義、変数の分離	(明) 6章、(猫) 6章
第５回　プロトタイプ宣言、分割コンパイル	(猫) 11章
第６回　配列の受け渡し、ポインタ	(明) 10, 11章、(猫) 7, 8章
第７回　まとめ、期末試験	

(明): 『新明解C言語』
(猫) : 『猫でもわかるC言語プログラミング』

授業計画

第１回　C言語の基本文法・言語処理系
第２回　数学的計算、配列、文字列
第３回　型変換、入出力処理
第４回　デバッグ、関数定義、変数の分離
第５回　プロトタイプ宣言、分割コンパイル
第６回　配列の受け渡し、ポインタ
第７回　まとめ、期末試験

今回の問い

- C言語はどこで活用されているのか？
- C言語の基本文法はどのようになっているのか？
- C言語で条件分岐や繰り返し処理はどのように行うのか？

今回の目標

C言語の簡単なコードを書いて実行できるようになる。

今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. C言語の基本
4. 条件分岐
5. 繰り返し処理

今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. C言語の基本
4. 条件分岐
5. 繰り返し処理

なぜC言語を学ぶのか？

- 多くのソフトウェアやOSはC言語で書かれている
- 処理が速く、メモリ消費量が少ない
- C言語を知っていると他のプログラミング言語の習得が容易になる

C言語の利用例

OS

アプリケーション

C言語の利用例

家電製品（組み込み機器）

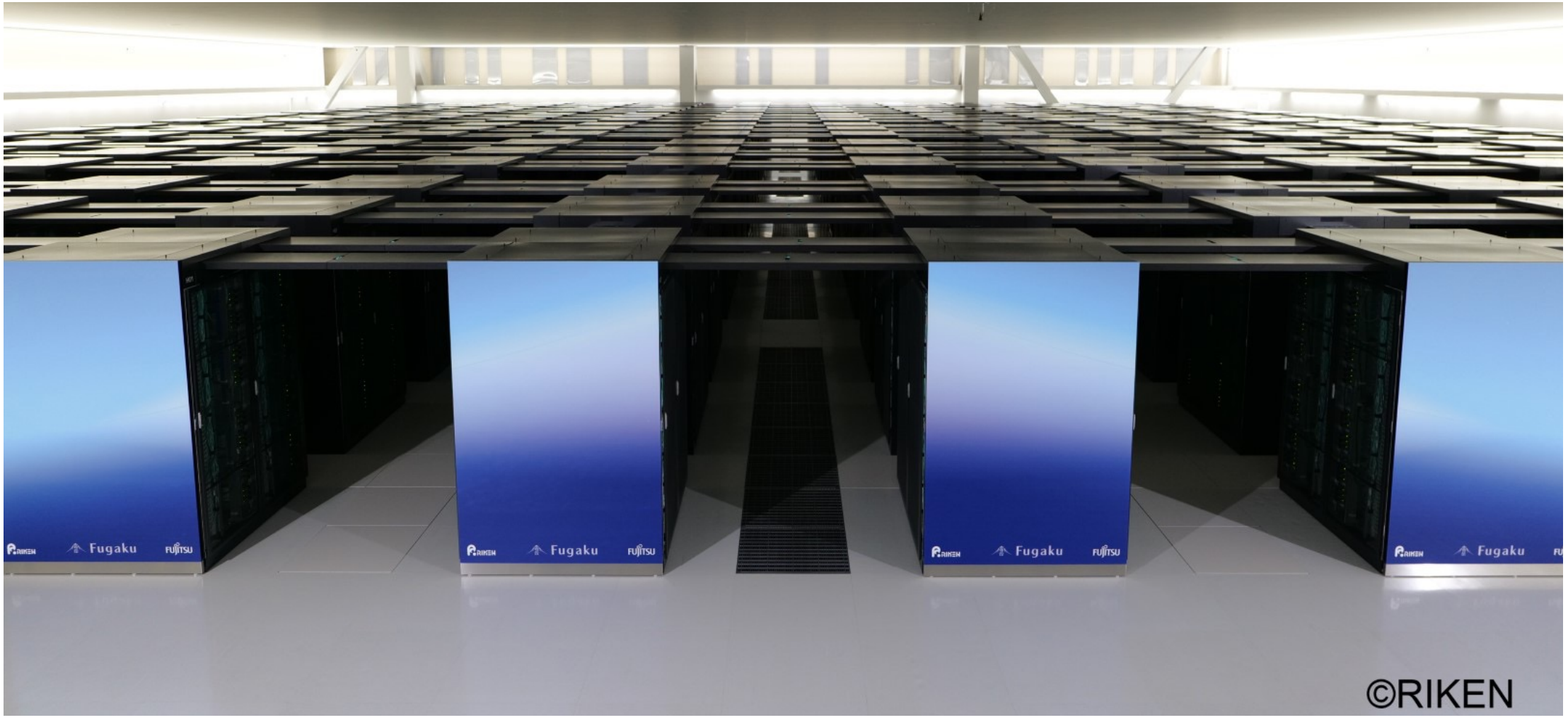
C言語の利用例

プログラミング言語

Rubyソースコード：
https://github.com/ruby/ruby

C言語の利用例

数値シミュレーション

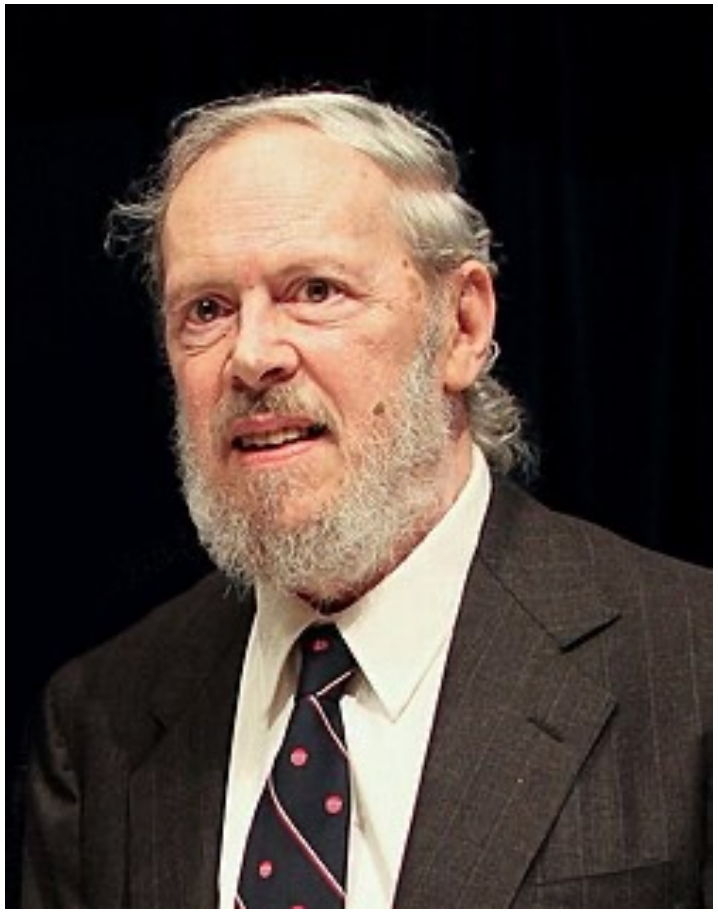


今日の問い

- C言語はどこで利用されているのか？
  - OS、アプリ、家電製品など広く利用されている。
- C言語の基本文法はどのようになっているのか？
- C言語で条件分岐や繰り返し処理はどのように行うのか？

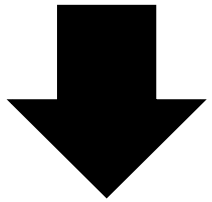
C言語の歴史

1972年　デニス・リッチー（ベル研究所）によりB言語の改良版として誕生



C言語の歴史

1973年UnixがC言語で書き直される



Unix系統のOSが広く普及

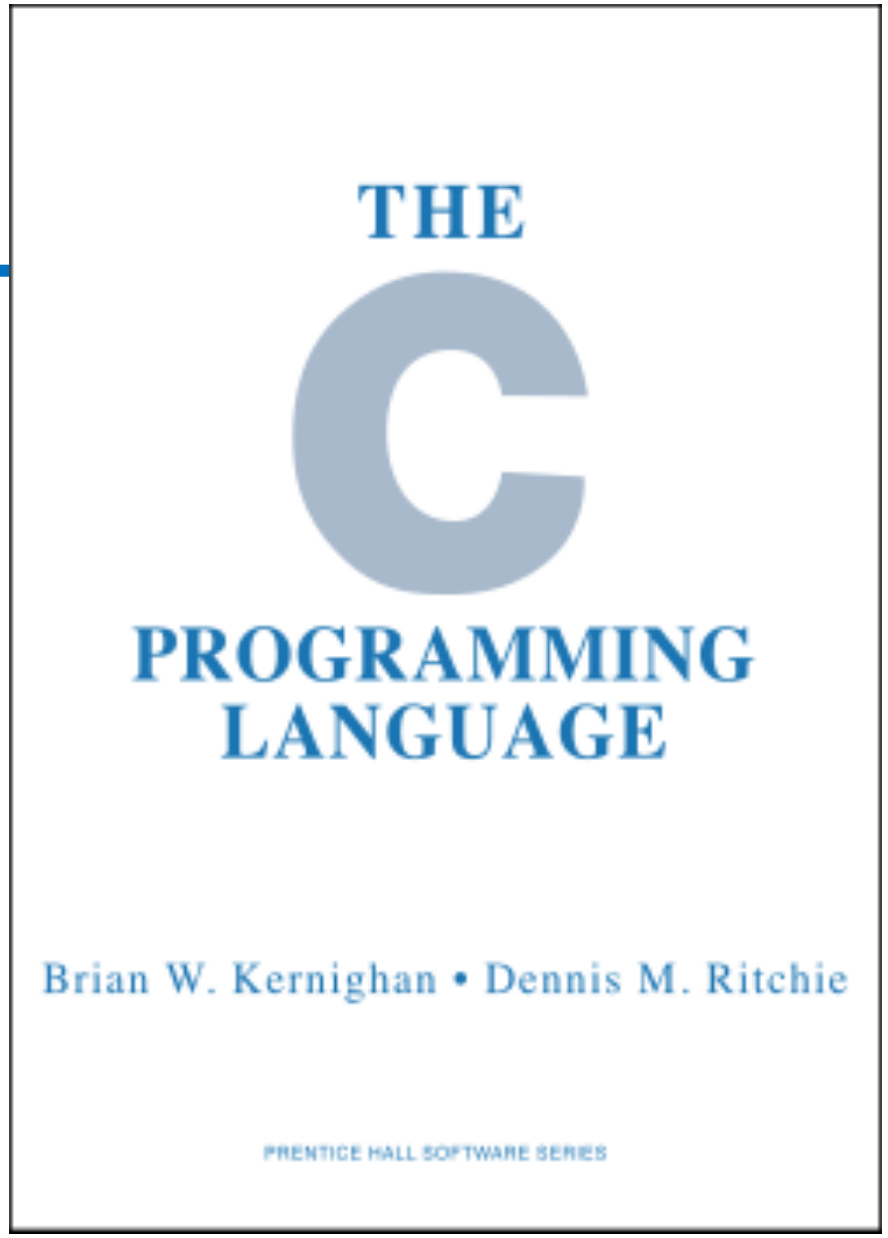
MacOS, Linux, BSDなど

C言語の歴史

1978年　開発者による教科書 “The C Programing Language”の出版

日本語版：https://amzn.asia/d/35q8Lrh

1989年　ANSI (米国規格協会）による標準化



C言語の歴史

現在でも人気プログラミング言語ランキング2位（TIOBE index）

https://www.tiobe.com/tiobe-index/

Sep 2026	Sep 2025	Change	Programming Language
1	1		Python
2	3	↑	C
3	2	↓	C++
4	4		Java
5	5		C#
6	6		JavaScript
7	7		Visual Basic
8	11	↑	SQL
9	13	⇈	R
10	18	⇈	Rust
11	12	↑	Fortran
12	**8**	⇊	**Go**
13	9	⇊	Delphi/Object Pascal

今回学ぶこと

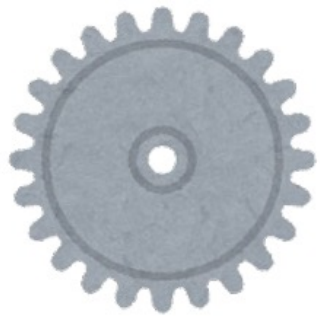
1. C言語の利用例と歴史
2. **言語処理系**
3. C言語の基本
4. 条件分岐
5. 繰り返し処理

コンパイル

プログラミング言語をコンピュータが読める機械語に翻訳

プログラミング言語

print(“Hello, world!”)



コンパイラ

機械語

1001000 1100101

言語処理系

あるプログラミング言語で書かれたソースプログラムを実際に実行させるまでの処理を行うシステムのこと。

言語処理系

本講義でC言語の学習に用いる言語処理系は**GCC (GNU Compiler Collection)**といい、Free Software Foundationという世界規模の団体が数十年前から開発を続けて無料で世界中に配布している。

インタプリタ言語とコンパイラ言語

インタプリタ言語

Ruby
A PROGRAMMER'S BEST FRIEND

```
print “Hello, world!”
```

コードを1行ずつ翻訳

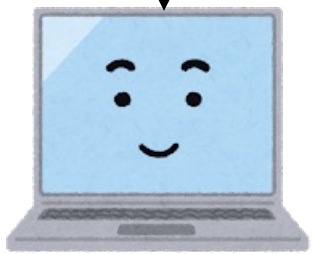
コンパイラ言語

C

```
#include <stdio.h>

int main()
{
   printf(“Hello, world!\n”);
   return 0;
}
```

コードを全体を翻訳



今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. **C言語の基本**
4. 条件分岐
5. 繰り返し処理

最初のプログラム

hello.cというファイルを作成し、右のプログラムを書いてみよう

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

最初のプログラム

ポイント

- 大文字と小文字を区別する。
- 半角英数字で記述する（スペースも半角）。

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

コンパイル

```
gcc hello.c -o hello
```

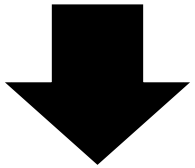
gccコマンド：コンパイルのため、言語処理系GCCを呼び出す

“-o”オプション：実行形式ファイル名を指定。
指定しないと“a.out”という実行形式ファイルが生成される。

実行

実行形式ファイルを実行

```
./hello
```



```
Hello, world!
```

と表示される。

ヘッダーファイル

関数や型などを定義するファイル

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

“stdio.h”というヘッダーファイルを読み込んでいる

main関数

```
#include <stdio.h>

int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

C言語のプログラムには必ずmain関数があり、プログラムはここから開始される（エントリポイント）。

main関数

```
#include <stdio.h>

int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

「{」から「}」で囲まれた部分をブロックという。

文(statement)

1つの処理単位

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

文末には必ず**セミコロン (;)**をつける

printf関数

```
#include <stdio.h>

int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

画面（標準出力）に文字を出力する。

printf関数

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

「**\n**」：改行
「****」：エスケープシーケンス（拡張表記）

return文

関数をそこで中断（終了）する。

```
#include <stdio.h>


int main()
{
    printf(“Hello, world!\n”);
    return 0;
}
```

システムに0
を返す。

コメントの書き方

「/*」から「*/」で囲まれた部分はコメントとみなされる。
「//」でも同様。

```
#include <stdio.h>
/*    これはコメント     */
int main()
{    //これもコメント
    printf(“Hello, world!\n”);
    return 0;
}
```

C言語の基本

- C言語はmain関数から始まる。
- 文の最後に「;」(セミコロン)をつける。
- 文字列を表示するにはprintf関数を使う。

今回の問い

- C言語はどこで利用されているのか？
  - OS、アプリ、家電製品など広く利用されている。
- **C言語の基本文法はどのようになっているのか？**
  - **C言語はmain関数から始まる。**
- C言語で条件分岐や繰り返し処理はどのように行うのか？

今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. C言語の基本
4. **条件分岐**
5. 繰り返し処理

制御文

プログラムの実行順序を変える文

if文

ある条件を満たした場合のみ文を実行

```
if (条件) {
    文1;
    文2;
}
```

if文

menkyo.cというファイルを作成し、右のプログラムを書いて実行してみよう。

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

   if (age < 18) {
      printf(“普通免許を取得できません。\n”);
   }

   return 0;
}
```

変数の宣言

int型(整数)の変数を宣言

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

   if (age < 18) {
      printf(“普通免許を取得できません。\n”);
   }

   return 0;
}
```

scanf関数

キーボードから入力したデータを読み込む

scanf(“指定演算子”, &変数);

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

    if (age < 18) {
       printf(“普通免許を取得できません。\n”);
    }

    return 0;
}
```

scanf関数

“%d”：10進整数入力

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

   if (age < 18) {
      printf(“普通免許を取得できません。\n”);
   }

   return 0;
}
```

scanf関数

「**&age**」：変数「age」に入力した整数を格納

「&」はアンパサンドと読む

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

   if (age < 18) {
      printf(“普通免許を取得できません。\n”);
   }
   return 0;
}
```

if else文

プログラムを右のように書き換え、実行してみよう。

```
#include <stdio.h>

int main()
{
   int age;

   printf(“年齢を入力してください。\n”);
   scanf(“%d”, &age);

   if (age < 18) {
      printf(“普通免許を取得できません。\n”);
   } else if (age >= 18 && age < 21) {
      printf(“普通免許は取得できます。\n”);
   } else {
      printf(“大型免許も取得できます。\n”);
   }

   return 0;
}
```

if else文

条件を満たさない場合を実行するにはelseを用いる。

```
#include <stdio.h>

int main()
{
    int age;

    printf(“年齢を入力してください。\n”);
    scanf(“%d”, &age);

    if (age < 18) {
        printf(“普通免許を取得できません。\n”);
    } else if (age >= 18 && age < 21) {
        printf(“普通免許は取得できます。\n”);
    } else {
        printf(“大型免許も取得できます。\n”);
    }

    return 0;
}
```

if else文

2つ以上の条件がある場合は右のように“else if (条件)”を用いる

```
if(条件X) {
    文1;
} else if (条件Y) {
    文2;
} else {
    文3;
}
```

条件分岐まとめ

条件によって処理を分けるには**if, else if, else**を使う。

今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. C言語の基本
4. 条件分岐
5. **繰り返し処理**

for文

希望の回数だけ繰り返して実行ができる。

```
for (初期設定式; 継続条件式; 再設定式) {
    文1;
    文2;
}
```

for文

for01.cというファイルを作成し、右のプログラムを書いて実行してみよう。

```
#include <stdio.h>

int main()
{
    int i;

    for (i = 1; i < 10; i++) {
        printf(“%d回目の繰り返しです。\n”, i);
    }

    return 0;
}
```

for文

```
for (i = 1; i < 10; i++) {
    printf(“%d回目の繰り返しです。\n”, i);
}
```

「i++」：iに1を足すという意味（i + 1）と同様

for文

```
for (i = 1; i < 10; i++) {
    printf(“%d回目の繰り返しです。\n”, i);
}
```

「%d」：整数値を出力。出力する整数はカンマ (,) の後。

for文

```
for (i = 1; i < 10; i++) {
    printf(“%d回目の繰り返しです。\n”, i);
}
```

1. 初期設定式は「i = 1」で、最初のみ評価される。

for文

```
for (i = 1; i < 10; i++) {
    printf("%d回目の繰り返しです。\n", i);
}
```

2. 継続条件式 (i < 10)は真なので、printfが実行される。

for文

```
for (i = 1; i < 10; i++) {
    printf(“%d回目の繰り返しです。\n”, i);
}
```

3. 再設定式 (i++)が評価される (まだ1増えない) 。

for文

```
for (i = 1; i < 10; i++) {
    printf(“%d回目の繰り返しです。\n”, i);
}
```

4. 条件式 (i < 10)が評価される (iは1増えて2になる) 。
5. 条件式が真なので、printfが実行される。
6. 同様の評価をiが10になるまで繰り返す。
7. iが10になると条件式は偽なのでprintfは実行されない。

while文

継続条件式が真（0以外）である限り、ブロック内が実行される。

```
while (継続条件式) {
    文1;
    文2;
}
```

while文

while01.cというファイルを作成し、右のプログラムを書いて実行してみよう。

```
#include <stdio.h>

int main()
{
    int i = 10;

    while (i > 0) {
        printf(“%d回目の実行です。\n”, 11- i);
        i--;
    }

    return 0;
}
```

while文

初期値はi = 10

```
int i = 10;

while (i > 0) {
    printf(“%d回目の実行です。\n”, 11- i);
    i--;
}
```

while文

継続条件式 (i > 0) は真なので、ブロック内が実行される。

```
int i = 10;

while (i > 0) {
    printf(“%d回目の実行です。\n”, 11- i);
    i--;
}
```

while文

「i--」：iから1を引く。i -1と同様。

```
int i = 10;

while (i > 0) {
    printf(“%d回目の実行です。\n”, 11- i);
    i--;
}
```

while文

i = 0になったら条件式(i > 0)は偽なので、ブロック内は実行されない。

```
int i = 10;

while (i > 0) {
    printf(“%d回目の実行です。\n”, 11- i);
    i--;
}
```

繰り返し処理まとめ

- 希望の回数だけ繰り返し処理を行うには**for文**を使う。
- 継続条件を先に判断してから繰り返し処理を行うには**while文**を使う。

今回の問い

- C言語はどこで利用されているのか？
  - OS、アプリ、家電製品など広く利用されている。
- C言語の基本文法はどのようになっているのか？
  - C言語はmain関数から始まる。
- **C言語で条件分岐や繰り返し処理はどのように行うのか？**
  - **if文、for文、while文などを使う。**

今回学ぶこと

1. C言語の利用例と歴史
2. 言語処理系
3. C言語の基本
4. 条件分岐
5. 繰り返し処理

今回の目標

C言語の簡単なコードを書いて実行できるようになる。

課題

年齢を入力させ、18歳未満なら「選挙権はありません。」、18歳以上30歳未満なら「選挙権はあります。」、30歳以上なら「被選挙権もあります。」と表示するプログラムsenkyo.cを作成せよ。生成AIでのコード生成は不可。

提出先：yutaka.hirai@koeki-u.ac.jp

件名：応用プログラミング第1回課題 [学籍番号]

締め切り：10月8日 (木)