# 情報処理実習
## 第9回: 文字処理

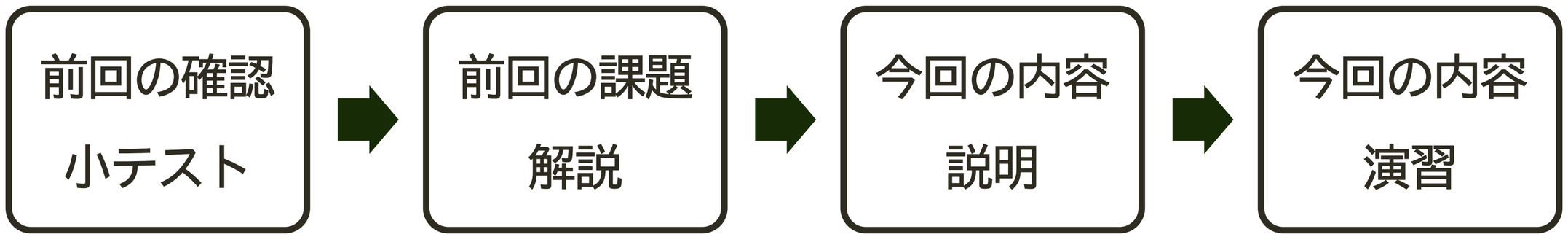
担当: 佐藤

2018年11月19日(月)

# アナウンス

- 授業前にやること
  - 計算機にログオン
  - CoursePowerにログイン
- 注意事項
  - 教室内飲食禁止
  - 原則として，授業中ではなく休み時間にトイレにいきましょう

# 授業の進め方



# 基本事項の確認小テスト

**制限時間: 5分**

① CoursePowerにログインして，すぐに解答が始められるよう準備する

② 開始の合図があるまで，解答を始めずに待機する

**注意**

「閉じる」ボタンは決して押さないこと!

# 文字の扱い方

2018年11月19日(月)

# C言語における文字の扱い

- **0以上の整数(非負整数)**を文字として扱う
- 文字に対応する0以上の整数を**文字コード**という
- 文字を扱うデータ型は**char(チャー，キャラ)**
- **%c**変換指定子で文字の入出力が可能

**例**

```
char c;

scanf("%c", &c);
printf("%cの文字コードは%d¥n", c, c);
```

**実行結果**

```
A⏎
Aの文字コードは65
```

⏎はエンターキー(リターンキー)を押すことを表す

# ASCIIコード

- **0から127まで**の文字コードを**ASCII(アスキー)コード**という
- アルファベットの大文字と小文字，数字や記号だけでなく，**目に見えない文字もある**．例えば，これまでの実習で使ってきた目に見えない文字として，次のものがある
  - ¥n(10): 改行文字
  - ␣(32): 空白文字(半角スペース)
- 割り当ての変更が認められている文字コードがある
  - 日本のコンピュータや日本語フォントでは，\(バックスラッシュ)(92)は，¥(円記号)で表示される場合が多い

# ASCIIコード表

| | 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|
|---|---|---|---|---|---|---|---|---|---|---|
| 0| | | | | | | | | | |
| 10| | | | | | | | | | |
| 20| | | | | | | | | | |
| 30| | | | ! | " | # | $ | % | & | ' |
| 40| ( | ) | * | + | , | - | . | / | 0 | 1 |
| 50| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; |
| 60| < | = | > | ? | @ | A | B | C | D | E |
| 70| F | G | H | I | J | K | L | M | N | O |
| 80| P | Q | R | S | T | U | V | W | X | Y |
| 90| Z | [ | ¥ | ] | ^ | _ | ` | a | b | c |
| 100| d | e | f | g | h | i | j | k | l | m |
| 110| n | o | p | q | r | s | t | u | v | w |
| 120| x | y | z | { | \| | } | ~ | | | |

- 数字やアルファベットの順番と文字コードの順番は同じ
  - 0(48)の次は1(49)
  - A(65)の次はB(66)
  - a(97)の次はb(98)

# 文字コードによる文字の扱い

- 文字コードで文字を表示することもできる

**例**

```
int i;

for (i = 65; i < 91; i++) {
   printf("%c", i);
}
printf("%c", 10); /* printf("¥n"); */
```

**実行結果**

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

# char型

- charは，1文字を記憶するデータ型
- ある文字を文字コードではなく文字として認識させるためには，その文字を'(一重引用符，シングルクォーテーション)で囲む
  - 文字を'で囲んだものを文字定数という

**例**

```
char c;

c = '0';
printf("%cの文字コードは%d¥n", c, c);
```

文字定数

**実行結果**

```
0の文字コードは48
```

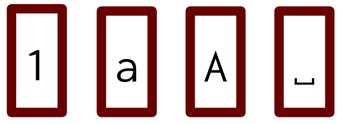
# char⇔intのキャスト
## 【発展】

2018年11月19日(月)

# 半角文字と全角文字
## 【発展】

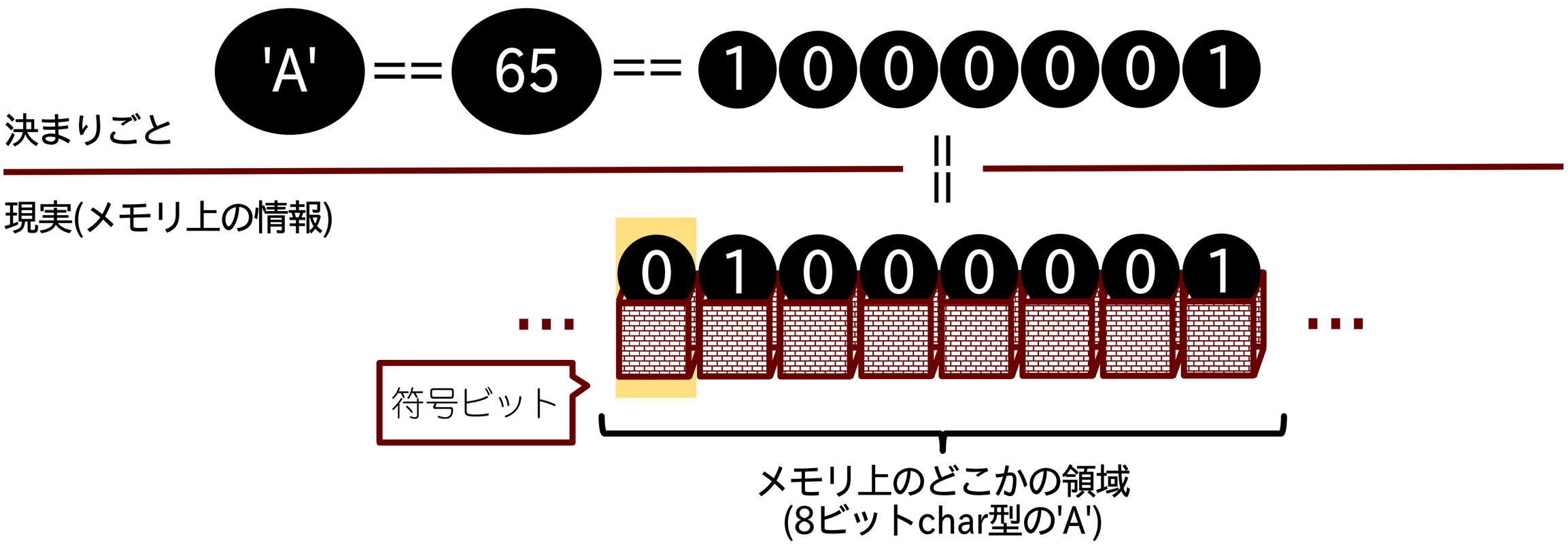
- 半角文字を「1バイト文字」，全角文字を「多バイト文字」ともいう
  - バイト(byte)は「複数ビット」を意味する情報量の単位
  - 1バイトは8ビットであることが一般的
    - ビット(bit)は，情報量の最小単位
    - 1ビットは2つの状態(0/1)を表現できる
- 半角文字は，1バイトで表現できる範囲の文字コードに対応する
  - 1バイトが8ビットである場合，$2^8$==256なので，0から255までの文字コードに半角文字1つを対応付けられる．ただし，0から127までの文字コードはASCIIコードであり，$2^7$==128であることから，8ビット中の下位7ビットで表現される半角文字は必ずASCIIコードに対応する文字である
- 日本語で使用される文字(平仮名，片仮名，漢字，etc.)はたくさんあるため，1バイトで表現できる範囲の文字コードにすべての文字を対応させることはできない．これらの文字は，通常2バイト以上で表現され，その場合，全角文字に該当する

ASCIIコード表に含まれる文字はすべて半角文字



# char型のサイズ
## 【発展】

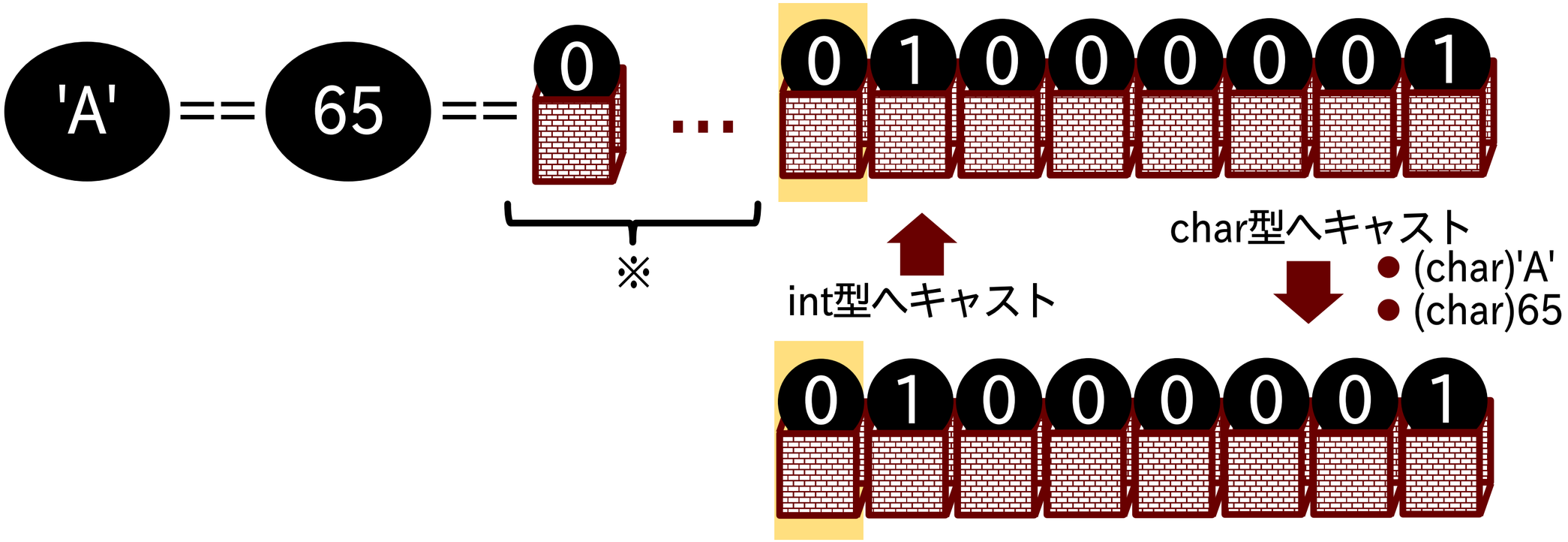
- C言語で単に「文字」という場合，ASCIIコードに対応する文字を意味する．よって，**char型は0から127までの整数を扱える整数型**
  - $2^7$==128なので少なくとも7ビットが必要．また，整数型なので正負の値を表現できる型であることが必要．そのため，char型のサイズ(情報量)は最上位ビットを**符号ビット**(0: +，1: -)とする**少なくとも8ビット**である
- C言語では，**1バイトはchar型のサイズ**



# char⇔intのキャストの詳細
## 【発展】

- char⇔intのキャストでchar型のサイズ分の下位ビットは不変
  - char型の下位7ビットで表現可能な範囲の整数値である**ASCIIコードは，char⇔intのキャストにより値が変わらない**

整数定数はint型として扱われるため，文字定数はint型
（変数だけでなく定数にもデータ型がある）




※int(大)➔char(小)のキャストでは，下位ビットのみが残り上位ビットは捨てられる
※char(小)➔int(大)のキャストでは，符号ビットと同じビットで上位ビットが埋められる(符号拡張)

# 文字の入出力

2018年11月19日(月)

# getchar()

- キーボードから1文字読み込み，読み込んだ文字に対応する文字コードを返す関数
- stdio.hをインクルードすると使用可能
- **戻り値の型はcharではなくint**
  - getchar()で文字を読み込む場合，読み込んだ文字を格納する変数はchar型ではなくint型として宣言しておくこと!

**文法**

```
getchar()
```

()の中には何も書かない

**例**

```
int c;

printf("1文字入力してください¥n");
c = getchar();
printf("入力された文字¥n%c¥n", c);
printf("入力された文字の文字コード¥n%d¥n", c);
```

**実行結果**

```
1文字入力してください
a⏎
入力された文字
a
入力された文字の文字コード
97
```

# putchar()

- 実引数が文字コードを表すものとしてそれに対応する文字を表示
- stdio.hをインクルードすると使用可能

**文法**

putchar(表示したい文字の**文字コード**)

- 文字aを表示したい場合の実引数
  - 'a'または97

**例**

```
int c;

printf("1文字入力してください¥n");
c = getchar();
printf("入力された文字は");
putchar(c);
printf("です¥n");
```

**実行結果**

```
1文字入力してください
a⏎
入力された文字はaです
```

# EOF

- 入力終了を表す文字コード^Zが入力された場合のgetchar()の戻り値は，EOF(End Of File)という整数値
  - コマンドプロンプト上で^Zを入力するにはCtrl+Zを押す
- stdio.hをインクルードすると使用可能

**例**

```
int c;

while((c = getchar()) != EOF) {
    putchar(c);
}
```

読み込んだ1文字をすぐに表示することを繰り返す

「c = getghar(), c != EOF」と書くこともできる

**実行結果**

```
Aoyama Gakuin University⏎
Aoyama Gakuin University
^Z⏎
続行するには何かキーを押してください . . .
```

⏎が入力されるまで文字を読み込み続ける

読み込んだ文字が一気に表示される

- この例のように，getchar()を用いる場合，1文字読み込んだだけでは反応がなく，改行文字などの区切りの文字に出会った際に，それまで読み込んだ文字をまとめて処理する効率的な入力処理が行われる．この処理をバッファリングという

# 課題

2018年11月19日(月)

# レポートの作成

① レポートの冒頭に以下を適切なレイアウトで書く
- 「情報処理実習第9回課題レポート」というタイトル
- 学生番号
- 氏名

② 課題ごとに以下を載せる
- 作成したプログラムのソースコード
- 作成したプログラムの実行結果を示すコマンドプロンプトのスクリーンショット

③ 完成したレポートをCoursePower上で提出する

**提出〆切: 18:30**