

プログラミング演習  
課題2  
10進数と2進数

2回目

## プログラミング演習

# 課題2 10進数と2進数

- 目的
  - 関数の定義の仕方や使い方, 標準関数を利用する仕組みを学ぶ
- 題材
  - (1回目) 整数 10進数の変換
  - (1回目) 学籍番号のチェックディジット生成法
  - (2回目) 整数 2進数の変換
  - (3回目) 2進数での加算
- 課題は5つ

# 課題概要

1回目

- 課題(2-1)
  - 学籍番号(整数)の10進6桁の各桁の数を取り出す
- 課題(2-2)
  - 学籍番号(整数)の10進6桁の各桁の数からチェックディジット生成

2回目

- 課題(2-3)
  - 0 ~ 255の整数の2進8桁の各桁の数を取り出す
- 課題(2-4)
  - 8桁の2進数から,それが表す整数を求める

- 課題(2-5)
  - 2つの8桁の2進数の和を計算する.

3回目

# 5つの課題のプログラムはそれぞれ 保存しておく

- (例) 課題(2-1)のソースコードを課題(2-3)のソースコードにコピーする単純な一手法
  1. Microsoft Visual C++ 6.0を2つ開く
  2. 一方は、既に作成済みの課題(2-1)のワークスペースを開いておく
  3. もう一方で、課題(2-3)用の新規プロジェクトを作成する。
  4. 課題(2-1)のCのソースコード全体を選択し、[コピー]
  5. 課題(2-3)のCの空のソースファイルに[貼り付け]

## 課題(2-1)

学籍番号(整数)の入力に対し, 1の位 ~ 100,000の位の数(10進数1桁目 ~ 6桁目の数)を求め, 出力せよ.

1. キーボードから学籍番号を入力し, int型変数に格納
2. 1の位 ~ 100,000の位について, 各桁の数を求め, それぞれ, int型変数に格納
3. 各桁の数を出力

## 課題(2-3)

0 ~ 255の整数の入力に対し、  
その2進数表現を求め出力せよ。

1. キーボードから0 ~ 255の整数を入力し、  
int型変数に格納
2. 2進数8桁の各桁の数を求め、それぞれ、  
int型変数に格納
3. 各桁の数を出力

# 課題(2-3)

- プログラムの構造は,  
課題(2-1)と同じ

課題(2-1) 10進 6桁

課題(2-3) 2進 8桁

$$n_i = (n / 2^i) \% 2$$

$$n_0 = (n / 2^0) \% 2 = n \% 2$$

$$n_1 = (n / 2^1) \% 2$$

• • • • •

## 課題(2-4)

- 0 / 1による8桁の2進数入力に対して,それが表す整数値を出力せよ(10進数で).このとき8桁の2進数からそれが表す整数を求める関数を作成・利用すること.
  1. キーボードから8桁の2進数を入力し,8個のint型変数に格納
  2. 8個の変数で表されている8桁の2進数を整数に変換する関数を呼び出し,返ってきた値をint型変数に格納する.
  3. 整数を出力する(10進数で)

# 課題(2-4)

- プログラムの構造
  - 関数bin\_to\_intとmain関数

課題(2-4)のソースコード

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

関数bin\_to\_intの定義

```
int main(void)
{
    . . .
}
```

配布資料を見て  
作成のこと。

# 出席

- 課題(2-4)のプログラムを実行し, TAが出題した2進数を入力し, 正しい10進表現が表示されれば出席とします.

# 宣言と定義の違い (p.6)

- **宣言**: 対象の性質をコンパイラに指示
  - 「こういうものを使うのでよろしく」
- **定義**: 宣言とメモリへの割り付けを指示
  - 「こういうものをメモリに配置せよ」
- **例**:
  - 変数宣言
  - グローバル変数定義
  - 関数定義
  - 関数プロトタイプ宣言

# 変数の種類

- グローバル変数: 関数の外で定義した変数
  - 定義・宣言以後はどこからでもアクセス可能
- ローカル変数: ブロック内で宣言した変数
  - そのブロック内でのみアクセス可能
- 関数のパラメータ
  - 呼び出し元が初期値を設定したローカル変数と考える
- グローバル変数とローカル変数に同じ変数名は使わないこと.
  - グローバル変数 `g_xxxx`等の命名規則で

# ローカル・グローバル変数の例

```
/* リスト10 */
#include <stdio.h>

int g_base;   グローバル変数

int base_to_power(int n)
{
    int p;

    p = 1;
    while (n > 0) {
        p = p * g_base;
        n = n - 1;
    }
    return p;
}
```

```
int main(void)
{
    int n;

    g_base = 2;
    n = 16;
    printf("%d to power %d = %d¥n",
           g_base, n, base_to_power(n));
    return 0;
}
```

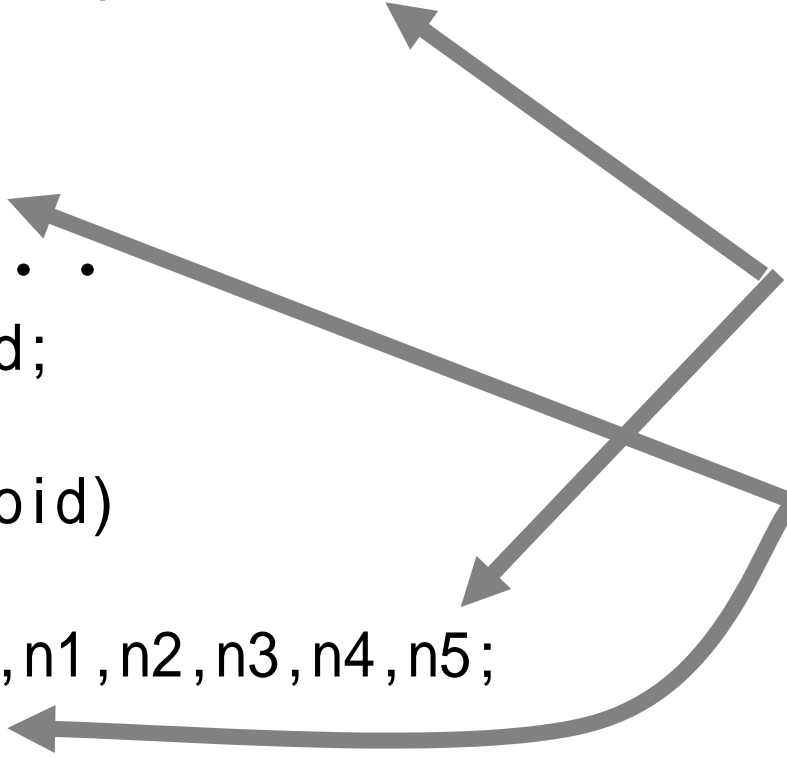
# ローカル変数間の関係

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

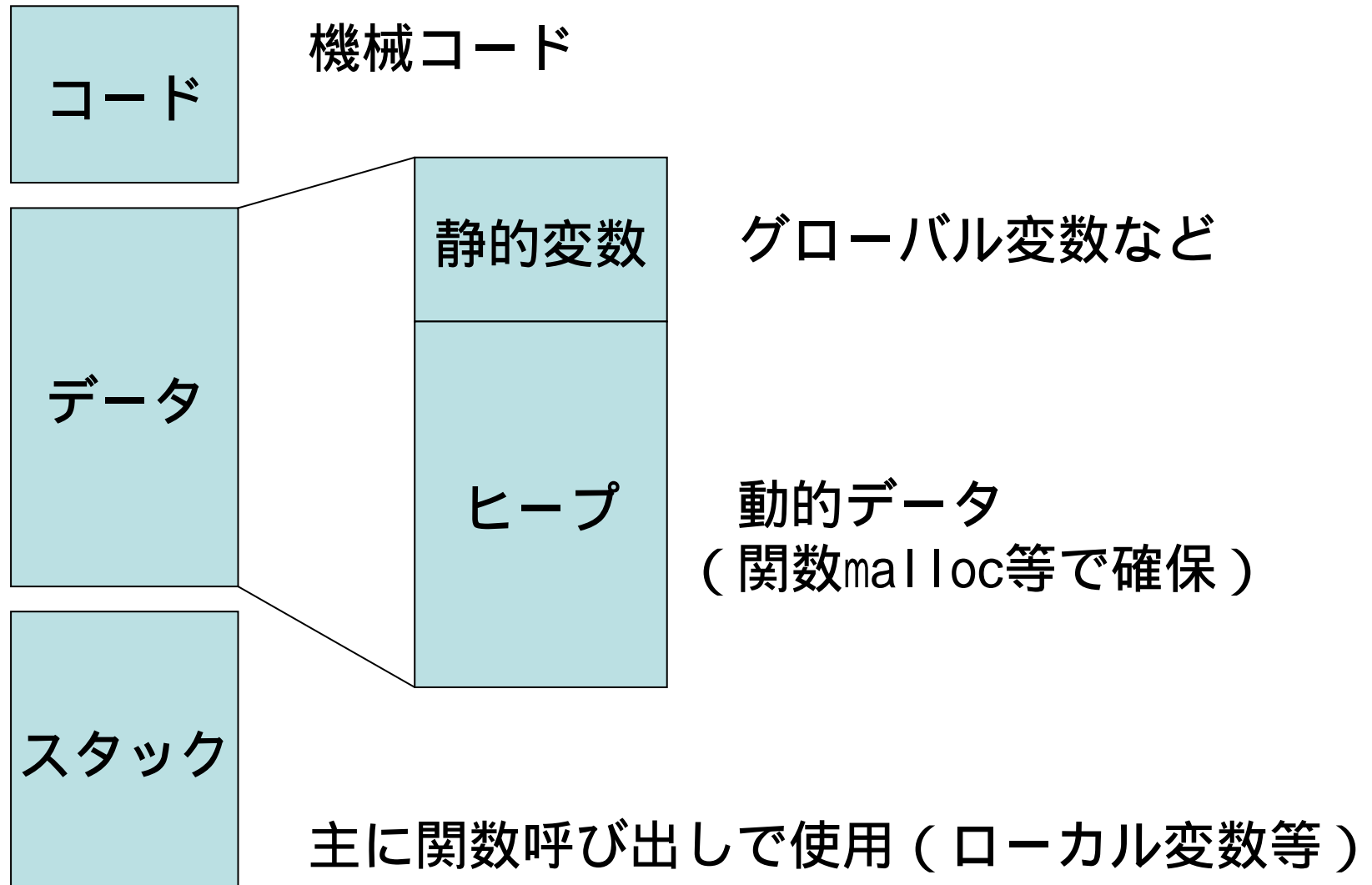
```
char check_digit(int n5, int n4, int n3, int n2, int n1, int n0)
{
    int m;
    char cd;
    . . . . .
    return cd;
}
int main(void)
{
    int n, n0, n1, n2, n3, n4, n5;
    char cd;
    . . . . .
    return 0;
}
```

無関係（異なる名前でもよい）

無関係（異なる名前でもよい）

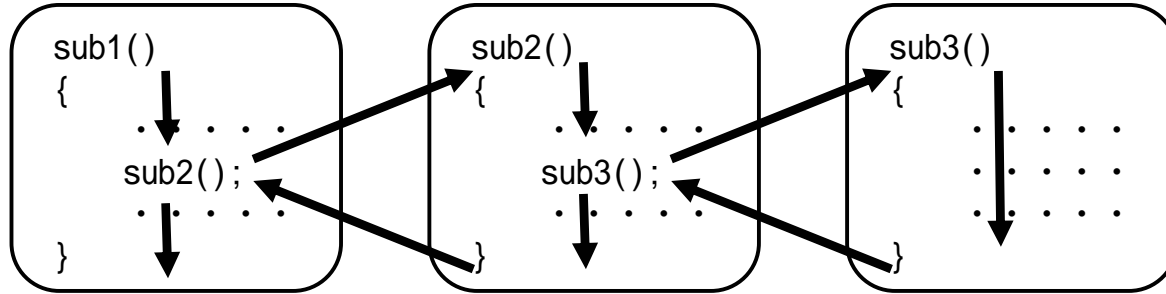


# C言語のメモリ領域



# スタックフレーム

スタックフレームに  
ローカル変数



関数を抜けると  
ローカル変数は  
なくなる

実行順序

