Programming II 0x0d

構造体(2011.06.25)

塩澤秀和 http://vilab.org

malloc / free（復習）

- 動的なメモリ割り付け
  - プログラム実行中に（動的に）、メモリ領域を確保する
  - 関数 malloc と free を使用する（#include <stdlib.h>）

- malloc（p.153）
  - 指定されたバイト数の“消えない”メモリを確保する
  - 戻り値はメモリ領域の先頭アドレス（失敗するとNULL）
  - 例： char *ptr;
    ptr = (char *) malloc(文字数 + 1); ← +1は'\0'の分

- free（p.154）
  - malloc で確保したメモリを解放する（戻り値はない）
  - 例： free(ptr);

記憶域の割り付け方式（復習）

- 静的な記憶域の割り付け
  - プログラム実行前に、固定のメモリ領域を確保しておく
  - プログラム実行中ずっと存在し、追加・拡張はできない
  - C言語：グローバル（外部）変数、static（静的）変数
- 動的な記憶域の割り付け
  - プログラム実行中に、必要な量のメモリ領域を確保する
  - 必要に応じて、拡張したり、解放（削除）したりできる
  - C言語：関数 malloc を使う（プログラマが管理する）
- （自動的・一時的な割り付け）
  - 関数に入るときに確保され、出ると自動的に解放される
  - C言語：（static以外の）ローカル変数、仮引数

動的な配列の確保（復習）

- 一般的な malloc
  - ポインタ = (型名 *) malloc(要素数 * sizeof(型名));
  - 例： int *ptr;
    ptr = (int *) malloc(n * sizeof(int));
- malloc 使用上の注意
  - 本当に必要なときに使う（普通の配列のほうが速くて安全）
  - 自前でメモリ使用を管理して、使わなくなったら free する
- NULLポインタとは？
  - 無効なことが保証されているポインタ（ポインタの値が0）
  - 注意： NULLポインタとヌル文字（'\0'）は別の意味

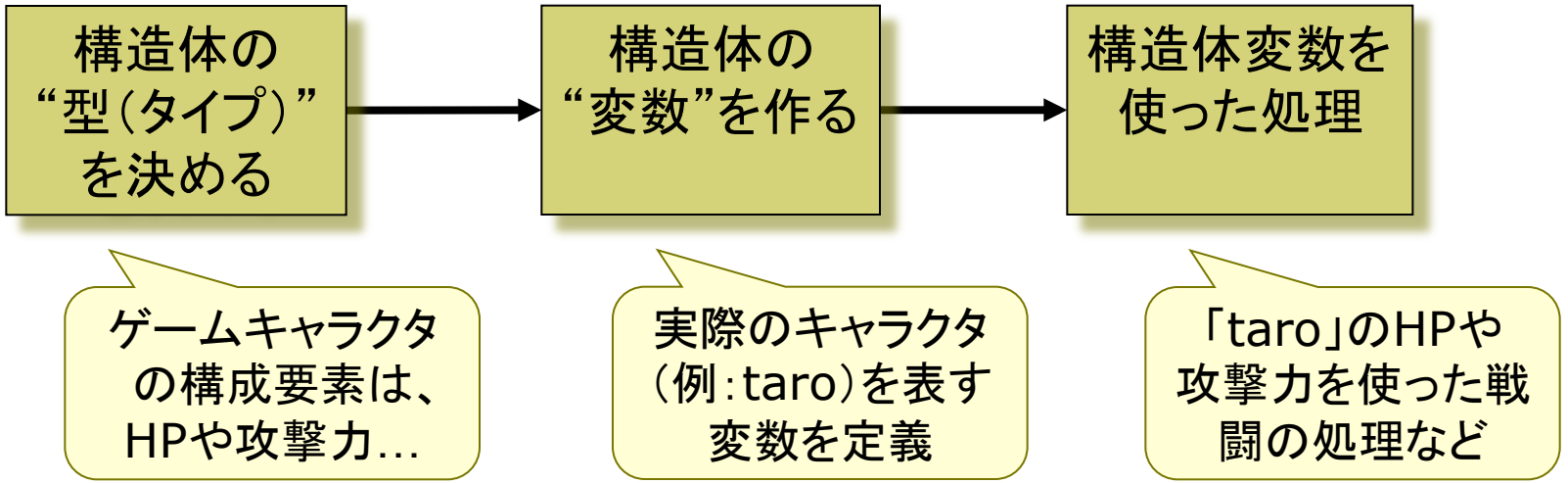
構造体

- 構造体(p.173)
  - データベースのような構造をもったデータ型
  - 関連する変数(属性)を集めて1つの変数の中にまとめて扱う

ゲームキャラクタ型

属性名	データ型
ヒットポイント	整数
攻撃力	整数
防御力	整数

- 構造体を使ったプログラムの流れ



構造体型と構造体変数

- 構造体型の宣言
  - タグ＝構造体の名前
  - メンバ＝構造体の構成要素

```
struct タグ名 {
  データ型  メンバ名;
  データ型  メンバ名;
  …
};
```

```
/* キャラクタ型の宣言例 */
struct character {
    int hp;         /* ヒットポイント */
    int attack;     /* 攻撃力 */
    int defense; /* 防御力 */
};
```

```
/* キャラクタ型の変数として */
/* taro と jiro を定義する例  */
struct character taro;
struct character jiro;
```

- 構造体変数の定義
  - 「struct タグ名」がデータ型名になる
  - 「int」や「double」と同じように、構造体型の変数を作れる

構造体メンバの参照

- メンバアクセス演算子「.」
  - メンバを変数として参照
  - 「構造体変数名.メンバ名」

    taro.hp = 100;

    printf("%d\n", taro.hp);

    scanf("%d", &taro.hp);

- 構造体変数の初期化
  - 配列の初期化のように { } の中にメンバの初期値を順番に並べる
  - struct character taro

    = { 100, 45, 56 };

```
#include <stdio.h>

/* 点の座標を表す構造体の宣言 */
struct point {
  double x, y;   /* メンバの宣言 */
};

int main(void)
{
  /* 構造体変数の定義 */
  struct point pt;

  /* 構造体メンバのアクセス */
  pt.x = 2.0;
  pt.y = 3.0;
  printf("x = %f\n", pt.x);
  printf("y = %f\n", pt.y);
  return 0;
}
```

構造体の使用例

```
#include <stdio.h>

/* 点の座標を表す構造体の宣言 */
struct point {
  double x, y; /* メンバの宣言 */
};

int main(void)
{
  struct point p1, p2;
  struct point pm; /* 中点 */

  printf("点P1の座標\n");
  printf("x = ");
  scanf("%lf", &p1.x);
  printf("y = ");
  scanf("%lf", &p1.y);

  printf("点P2の座標\n");
  printf("x = ");
  scanf("%lf", &p2.x);
  printf("y = ");
  scanf("%lf", &p2.y);

  /* 中点の計算(メンバのアクセス) */
  pm.x = (p1.x + p2.x) / 2;
  pm.y = (p1.y + p2.y) / 2;

  printf("中点の座標\n");
  printf("x = %f\n", pm.x);
  printf("y = %f\n", pm.y);
  return 0;
}
```

構造体の配列

```
#include <stdio.h>
#define N 3

/* 構造体の宣言 */
struct person {
  char name[20]; /* 氏名 */
  double height; /* 身長 */
  double weight; /* 体重 */
};

/* 構造体変数の配列       */
/*   グローバル変数による */
/*   データ配列の定義例   */
struct person data[N];

int main(void)
{
  int i;

  for (i = 0; i < N; i++) {
    printf("%d番\n", i + 1);
    printf("名前: ");
    scanf("%s", data[i].name);
    printf("身長: ");
    scanf("%lf", &data[i].height);
    printf("体重: ");
    scanf("%lf", &data[i].weight);
  }
  for (i = 0; i < N; i++) {
    printf("%d番 %sさん\n",
           i + 1, data[i].name);
    printf("身長 %5.1f cm\n",
           data[i].height);
    printf("体重 %5.1f kg\n",
           data[i].weight);
  }
  return 0;
}
```

構造体のポインタ

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

/* 点の座標を表す構造体の宣言 */
struct point { double x, y; };

/* 構造体へのポインタを引数とする関数 */
double dist(struct point *p1,
            struct point *p2)
{
  double d, dx, dy;

  /* 間接メンバアクセス演算子「->」 */
  /* p2->x は(*p2).x と同じ意味 */
  dx = p2->x - p1->x;
  dy = p2->y - p1->y;
  d = sqrt(dx*dx + dy*dy);
  return d;
}
```

```
int main(void)
{
  struct point p1, p2;
  double d;

  printf("点P1の座標\n");
  printf("x y: ");
  scanf("%lf %lf",
        &p1.x, &p1.y);

  printf("点P2の座標\n");
  printf("x y: ");
  scanf("%lf %lf",
        &p2.x, &p2.y);

  d = dist(&p1, &p2);
  printf("距離 %lf\n", d);
  return 0;
}
```

演習問題（レポート7/14）

13a. 3次元座標 (x, y, z) からなる構造体 point3d を定義して、その変数を2つ作成しなさい。キーボードから2点の座標を読み込み、中点の座標を表示するプログラムを作成しなさい。

13b. 氏名（文字列）、年齢（整数）、身長（実数）からなる構造体を定義し、その構造体の配列にキーボードから10人分のデータを読み込むプログラムを作成しなさい。

13c. 13bを改造し、何人分のデータがあるか最初にキーボードから読み込んで、配列をmallocで確保するようにしなさい。

13d. 13cの続きとして、読み込んだデータをすべてファイルに保存するようにしなさい。（どういう形式で保存したか？）

連絡事項

- 統一期末試験（第14回）
  - 日時　6月30日（木）　7･8限（15:00～17:00）
  - 場所　8号館　325教室（いつもの教室）
  - 範囲　ポインタ，malloc，（ファイル処理，配列）
  - 今回の対策問題と後半の演習課題をすべて見直すこと

- 第15回（レポート）
  - 授業は行わず，自習&レポート提出（節電対策）
  - 構造体の演習課題 13a～13d
  - 7/14（木）24:00 までに，解答を電子メールで提出
  - 提出先：shiozawa @ eng.tamagawa.ac.jp