C++ 第2回 レポート

倉庫番 の実装
(sokoban6.cc)

15k0112 北澤 祥太

提出日: 2016-11-18

目次

第 1 章　序論

　倉庫番とは1982年12月に有限会社シンキングラビットから発売されたコンピュータパズルゲームである。基本的に以下のルール [1] に則った 2D ゲームである。

・フィールドは荷物、壁、ゴール地点、そしてプレイヤーが操作する人間からなる。荷物とゴール地点の数は等しい。
・ゲームの目的は人間を操作し、すべての荷物をゴール地点に運ぶことである。
・人間は上下左右に移動することができる。進行方向に荷物がある場合、その荷物を押すことができる。ただし、荷物の進行方向に他の荷物や壁が存在しない場合に限る。荷物を引っ張ることはできない。

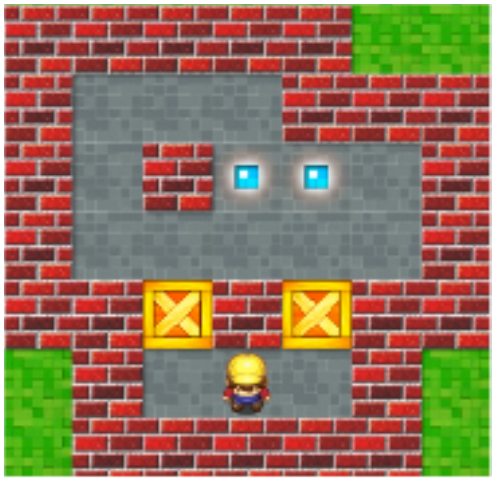

図1 倉庫番の参考画像

　倉庫番を、ncurses ライブラリを用いて TUI で実装するのが今回の目標である。

図2 TUI 版倉庫番の参考画像

以下、2 章では倉庫番を実装するための設計思想について説明し、3 章では実装した際におけるソースコードを提示する。最後の 4 章で本レポートをまとめる。

第 2 章　設計

　倉庫番は、荷物、壁、ゴール地点、プレイヤーが操作する人間からなる。そのうちプレイヤーの操作によって移動するものは、荷物と人間である。そのため自分は、今回の目標となる倉庫番の実装において、設計思想として ”「荷物、壁、ゴール地点、人間」の情報を格納した「ステージオブジェクト」の実装” を意識した。このようなオブジェクトをポインタで管理することで、各ステージは呼び出された際にヒープメモリに保存され、どの関数からでもステージを参照することができるようになる。また、ステージが呼び出されたときに new を行い、ステージを使用しなくなったら delete を行うことができるので、プログラム実行時においてメモリの節約にも繋がる。
　次に、倉庫番の実装においてステージのデータは必要不可欠である。ステージは 1 章の 図2 のようになる。したがって今回の実装では、stage.dat ファイルにステージの最初の状態のマップをテキストベースで別ファイルに保存しておき、ifstream で呼び出し ncurses で表示させる形とした。また、ステージの選択機能に対応するために、stage0.dat から stage9.dat までのファイルを作成しておく。
　また、倉庫番にはゲームクリアが存在する。なのでゲームクリアを示すもの必要である。今回の実装ではアスキーアートをテキストベースで別ファイルに保存しておき、ifstream で呼び出し ncurses で表示させる形とした。
　したがってソースコードのファイル構成は以下となる。

・./sokoban6.cc
　・メイン関数
・ ./Stage.h
　・Stage クラスを定義
・./warehouse/stage[0-9].dat
　・ステージのマップをテキストベースで保存
・./win.dat
　・ゲームクリア時に表示するものをテキストベースで保存

第 3 章　実装

　今回、以下の環境で実装を行った。また、ソースコードは、本レポートとともにアーカイブされているファイルのうち「sokoban6.cc」,「Stage.h」 を参考されたい。

・OS: CentOS 7.2
・カーネル: 3.10.0-327.36.3.el7.x86_64
・コンパイラ: g++ (GCC) 4.8.5 20150623 (Red Hat 4.8.5-4)
・文字コード: ASCII (LF)
・エディタ: VIM - Vi IMproved 7.4

　オブジェクトの実装においてまず考えることは、オブジェクトにどのようなメンバ変数をもたせるかである。今回の実装において、自分は「ステージオブジェクト」に以下のメンバ変数をもたせた。以下の説明が指す ”行動” とは、プレイヤーが動かす人間が左右上下いずれかに移動することである。

・vector<vector<char> > warehouse;
　・壁, ゴール地点のマップを2次元配列で表現する。(定数)
・int height;
　・ステージの縦幅である。(定数)
・int width;
　・ステージの横幅である。(定数)
・vector<vector<int> > baggage;
　・荷物のある座標を2次元配列で格納する。(1行動毎に更新)
・int own[2];
　・自分の座標を1次元配列で格納する。(1行動毎に更新)
・vector<int> way;
　・前回の行動でどの方向に動いたかの情報を格納する。(1行動毎に追加)
・vector<int> flag;
　・前回の行動で荷物を押したかどうかを格納する。(1行動毎に追加)
・vector<vector<int> > prebaggage;
　・前回の行動の前に荷物があった座標を2次元配列で格納する。(荷物を押す毎に追加)

　次に、このオブジェクトを呼び出すときにどのような引数が必要かを考える。今回の実装においては以下を引数に取るようにした。

・“stage[0-9].dat”
　・string 型で、呼び出したいステージファイルの名前を引数に取る。

これにより、初期コンストラクタにステージ名を渡すことで、そのステージの情報をメンバ変数に格納することが出来る。

次に、コンストラクタについてである。今回の実装では以下のようなパブリックコンストラクタを作成した。

・void moving(int n)
　・数字に対応した方向に進む。
・void print()
　・現在のステージ状態を ncurses で表示する。
・bool beat()
　・クリア判定を行う。
・void goback()
　・一手戻る。

あとは sokoban6.cc 内の main 関数にて以下の手続を行えばよい。

1. 初期コンストラクタを new で呼び出し、インスタンス領域の確保、インスタンスのポインタを作成する。
2. キー入力を受け付ける。
3. switch 文で以下のいずれかを実行する
　・矢印キー、または h, j, k, l キーが入力されたならば、入力キーに対応した数字を引数にとった moving コンストラクタを呼び出す。
　・b キーが入力されたならば goback コンストラクタを呼び出す。
　・r キーが入力されたならば、現在のインスタンスを delete し、再度初期コンストラクタを new で呼び出す。
4. printコンストラクタを呼び出し、ncurses でステージを表示する。
5. beatコンストラクタを呼び出し、クリア判定を行う。
　・もしクリアならば win.dat ファイルの内容を ncurses で表示する。
　・クリアでなければ 2. に戻る。

これにより、倉庫番を実装することができた。しかしこの倉庫番プログラムは「sokoban6.cc」に書かれたファイル名のステージしか参照できない。なので次に、「./warehouse/」以下にあるファイルをステージとして選択できる機能を実装する。

この機能は main 関数内に実装する。なぜなら、ステージ選択はインスタンスを作成する前の手続きであるからだ。今回の実装でそれを実現するために、以下の関数を作成した。

・string initstage()

また、 initstage 関数での手続は以下である。

1. dirent.h を用いて、「./warehouse/」以下のファイル名を取ってくる
2. ncurses で取ってきたファイル名と数字の添字を表示する。
3. キー入力を受け付ける。
4. 入力されたキーが添字に存在するならば、それに対応するファイルの相対パスを string 型で返す。存在しなければ空文字を返す。

以上の手続きによってファイル選択も実装することができた。

第 4 章 まとめ

今回の課題では TUI での簡易的なゲームの作り方を学んだ。具体的には、TUI での基本的な倉庫番の実装、一手戻る機能の実装、最初に戻る機能の実装を行った。

引用

[1] Wikipediaより: https://ja.wikipedia.org/wiki/倉庫番

参考

プログラミング演習1 (C/C++) 2016 講義資料 第5回 - 第9回