プログラミング演習I レポート

課題名： 課題 1 階乗を求めるプログラム

学籍番号	123456A
氏名	宇都宮　太郎
提出日	令和　8　年　4　月　1　日

【チェック項目】

☒ テンプレートの章立てを遵守すること
☒ 赤文字による指示文は提出時に削除すること
☒ 全ての図に図番号と図タイトルをつけること
☒ 全ての表（リスト）に表番号と表タイトルをつけること
☒ 全ての数式に数式番号をつけること
☒ 図番号は図の下部，表番号は表の上部に記載し，図表とも左右中央寄せにすること
☒ 数式番号は数式と同じ行の右端に寄せること
☒ 本文の文章は「である調」で統一すること
☒ 本文の文章はフォントの種類とサイズを統一すること
（見出しや図表中のフォントは異なっても良い）
☒ 本文の文章はむやみに改行しないこと
（内容や話題が大きく変わる場合にのみ，改行して段落を分けること）
☒ 段落の最初は全角空白を 1 文字入れること
☒ 段落と段落の間に空行を空けないこと
（見出しや図表の前後は空行を空けてもよい）
☒ ページ番号が正しいか確認すること
（表紙のページ番号は不要で，次のページから 1 ページ目が開始される）
☒ 表紙の全ての項目を記入し，チェックボックスにチェックをすること
☒ レポートファイルは PDF 形式に変換してから提出すること

コメントの追加 [SK1]: 該当する授業の数字を記入する．

コメントの追加 [SK2]: 該当する課題名を記入する．

コメントの追加 [SK3]: 学籍番号，氏名，提出年月日を記入する．

コメントの追加 [SK4]: 提出前にチェック項目を再度確認し，チェックをつけてから提出する．

1　レポート課題の概要

本課題では，再帰呼び出しを用いて n の階乗を実装し，入力された数値に対応する階乗を出力するプログラムを作成する．

2　データ構造とアルゴリズム

2.1　再帰的手続きによる階乗の計算

階乗とは，自然数 n に対し，1 から n までの自然数の総乗をいう．これを $n!$ と書き，式 (1) のように表せる．

$$n! = n \times (n-1) \times (n-2) \times (n-3) \times \cdots \times 1 \tag{1}$$

先の定義では $0!$ は定義されないが，便宜上 $0! = 1$ とされる．これは，$(n-1)! = n!/n$ であるので，$0! = 1!/1 = 1$ と考えられるためである．式 (1) は再帰的な手続きにより表現でき，式 (2) のように表せる．

$$n! = \begin{cases} n \times (n-1)! & (n > 0) \\ 1 & (n = 0) \end{cases} \tag{2}$$

本課題では，式 (2) を用いて再帰呼び出しによる階乗計算を実装する．n に相当する数値はユーザーが入力から与えるものとし，標準出力として計算結果を出力する．さらに，入力および結果の出力は複数回繰り返せるものとし，繰り返し回数はプログラム内の定数によって決定する．

2.2　データ構造

定数として MAX_INPUT を宣言し，この数値を階乗計算の繰り返し回数とする．

（注：課題やプログラムによっては，構造体，グローバル変数，複雑なポインタや配列などを利用することがあり，一見すると何にどう使用するデータか不明瞭なものが多い．その場合は，それらの型および役割について本節で説明することが望ましい．）

コメントの追加 [SK5]: 章タイトルのフォントはゴシックかつ本文より大きいポイントを推奨する．

コメントの追加 [SK6]: 本文のフォントは MS 明朝 10.5 ポイントを推奨する．

コメントの追加 [SK7]: 章構成および章タイトルはレポート課題ごとに異なるため，担当者の指示に従うこと．

コメントの追加 [SK8]: 数式番号を付与した数式は本文中で必ず参照する．

コメントの追加 [SK9]: 数式には数式番号を右端に付与する．

コメントの追加 [SK10]: 文中に含まれる短い数式の場合は，数式番号の付与は不要になる．

コメントの追加 [SK11]: ページの下部中央にページ番号をつける．表紙にページ番号はつけず，表紙の次のページから 1 ページ目が開始されるようにする．

2.3　アルゴリズム

プログラム全体の動作のフローチャートを図 1 に示す．このプログラムでは，まずデータの入力を促し，整数型の変数 data に格納する．data が正の場合は，再帰呼び出しを用いて data の値の階乗を求め，結果を表示する．負の場合はエラーメッセージを出力する．階乗を計算する処理は，MAX_INPUT で定義された回数分繰り返す．

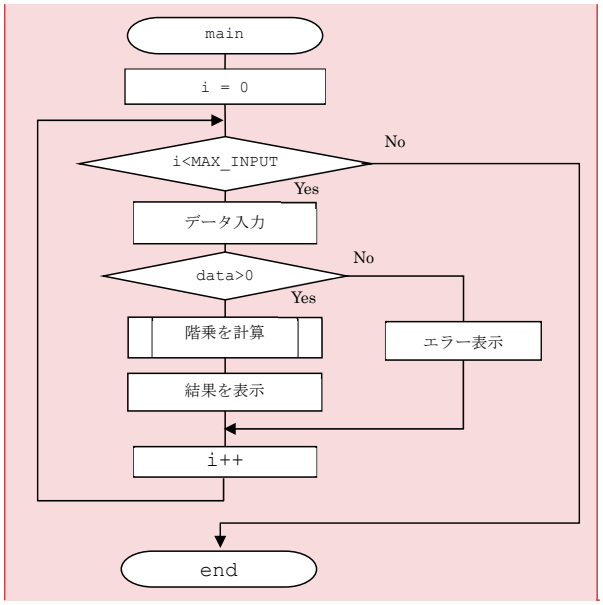

図 1　main 関数のフローチャート

2.4　関数の仕様

以下，関数ごとにその引数，戻り値，動作を説明する．

- int factorial(int n)
  - 引数 n：int 型．階乗を計算する値を表す．
  - 戻り値：int 型．n の階乗を計算した結果を返す．
  - 動作：関数 factorial は n の階乗を求める関数である．式 (2) の考え方に従って，再帰的に階乗の値を計算する．再帰的に階乗を計算するには…（説明を省略）

3　プログラム

作成したプログラムのソースリストをリスト 1 に示す．プログラムの説明は，ソースリストの中にコメントとして記載した．

コメントの追加 [SK12]: 図番号を付与した図は本文中で必ず参照する．

コメントの追加 [SK13]: フローチャートの作成にはどのようなツールを使用してもよいが，フローチャートの書き方に関する資料（flow_guide.pdf）を参照すること．また，作成ツールとしては以下のサイトが便利である．
https://www.draw.io/

コメントの追加 [SK14]: フローチャートなど，カラーを使用する必要性のない図は，基本的に白地に黒の単色で作成する．

コメントの追加 [SK15]: 図の下に図番号とタイトルをつける．

コメントの追加 [SK16]: 実装した関数についての説明を求められた場合は，関数の宣言部を明記した上で，各引数，戻り値，関数の動作についてそれぞれ説明することが望ましい．図や表などを使用してもよい．

コメントの追加 [SK17]: 表番号（またはリスト番号）を付与した表は本文中で必ず参照する．

リスト 1　階乗を求めるプログラム

```
 1:  #include <stdio.h>
 2:  /*** 定数宣言 ***/
 3:  #define MAX_INPUT 3 // 入力回数
 4:
 5:  /******************
 6:  n の階乗を求める関数
 7:  入力：整数型
 8:  戻り値：整数型
 9:  ***************** /
10:  int factorial(int n)
11:  {
12:      if (n > 0) {
13:          return n * factorial(n - 1);
14:      } else {
15:          return 1;
16:      }
17:  }
18:
19:  /******************
20:  メインルーチン
21:  ***************** /
22:  int main(void)
23:  {
24:      int data;   // 入力データ
25:      int i;     // ループカウンタ
26:
27:      for (i = 0; i < MAX_INPUT; i++) {
28:          // データの入力
29:          printf("[ %d回目 ] 正の整数を入力して下さい ", i + 1);
30:          scanf("%d", &data);
31:
32:          // 結果の表示
33:          if (data > 0) {
34:              printf("計算結果：%d! = %d¥n", data, factorial(data));
35:          } else {
36:              printf("error:入力値が不正です¥n");
37:          }
38:      }
39:      return 0;
40:  }
41:
```

コメントの追加 [SK18]: 表（リスト）の上に表（リスト）番号とタイトルをつける．

コメントの追加 [SK19]: ソースリストのフォントは等幅フォントの使用を推奨する．フォントサイズは本文より少し小さい方が可読性が高いが，本文と同じでも問題はない．

コメントの追加 [SK20]: ソースコードの各行には行番号を付与するとよい．

コメントの追加 [SK21]: プログラムの動作，処理のかたまり，変数の意味などは，演習中にコメントとしてソースコードに記入しておくと，後でソースを読み直したときに理解が容易になるとともに，レポートとして提出した際に他者が読みやすいレポートになる．

コメントの追加 [SK22]: 処理行ごとに適切なインデント（字下げ）を意識する．Visual Studio では自動調整機能（ドキュメントのフォーマット）があるため，利用を推奨する．

（参考までに，読みにくいソースリストの例をリスト 2 に示す．リスト 1 と同じプログラムでも，書き方によって，分かりやすさが格段に異なっている．）

リスト 2　読みにくいソースリストの例

```
#include <stdio.h>
#define bbbb 3
int factorial(int n)
{
if (n > 0) {
return n * factorial(n - 1);
} else {
return 1;
}
}
int main(void)
{
```

コメントの追加 [SK23]: 意味のある変数名ではないのでわかりにくい．

コメントの追加 [SK24]: インデントされていないので読みにくい．

```
        int data;
        int i;
        for (i = 0; i < bbbb; i++) {
        printf("[ %d回目 ] 正の整数を入力して下さい ", i + 1);
                scanf("%d", &data);
if (data > 0) {
        printf("計算結果：%d! = %d¥n", data, factorial(data));
                } else {
                        printf("error:入力値が不正です¥n");
                }
        }
        return 0;
}
```

コメントの追加 [SK25]: インデントがばらばらで読みにくい.

4　実行結果

実行結果を以下に示す．図 2 の実行結果 1 では，MAX_INPUT 回 (=3 回) データ入力を行い，それぞれ計算結果が出力されている．[1 回目] は 12 != 479,001,600 で，正しく計算されている．[2 回目] は，13！= 1,932,053,504 という結果が出たが，正しい計算結果は 6,227,020,800 であり，異なっている．この理由については，次の章で考察する．[3 回目] は入力が負の値であったので，正しくエラーメッセージが出力されている．

図 3 の実行結果 2 は，実数値を入力した例である．この場合，入力値がこのプログラムの仕様と異なるので，正しく実行されない．

図 4 の実行結果 3 は，アルファベットを入力した例である．この場合も，入力値がこのプログラムの仕様と異なるので，正しく実行されない．

コメントの追加 [SK26]: 実行結果の章は，結果を載せるだけでなく，どのように実行（何を入力）したのか，各図はどのような結果（出力）になったものなのか，結果の正否について，文章でも説明することが望ましい.

コメントの追加 [SK27]: 実行結果の掲載方法（テキスト，図，表）は，各課題で指定があればそれに従う.

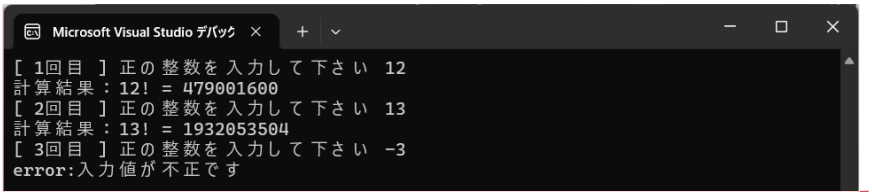

図 2　実行結果 1

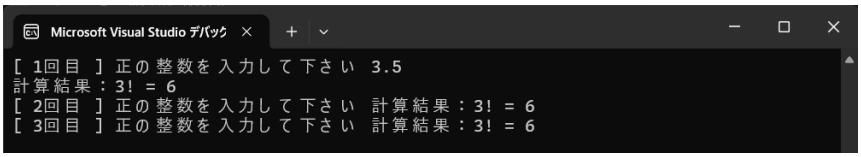

図 3　実行結果 2

図 4　実行結果 3

5　考察

このプログラムでは，13 以上の数値を入力すると正しい結果が得られなかった．これは，データを格納するために int 型を使用しているためであると考えられる．int 型で表現できる最大値は 231－1（＝2,147,483,647）であり，13!（＝6,227,020,800）はこの最大値を超えたためオーバーフローを起こしたと考えられる．このプログラムで計算できる最大値は 12! であるので，入力時に最大値のチェックを行う処理を加えれば，想定より大きい数値が入力されても対応可能なプログラムに改善されると考えられる．あるいは，関数 factorial の引数や戻り値を実数型にすれば，扱える数値の範囲を広げることができる．また，実数やアルファベットなど，想定外の入力に対するエラーチェック機能も改良の余地がある．

今回の課題では，再帰呼び出しを使うことで．単純なコードで階乗の計算を実現している．ただし再帰呼び出しを行なうプログラムの実行時には，スタックの消費量に注意する必要がある [1]．階乗計算は再帰構造を利用せずとも for 文や while 文による実装が可能であるため，スタックオーバーフロー，メモリ使用量，実行速度のいずれの面でも…（説明を省略）

6　感想（任意）

データ型と扱える数値の範囲について理解が深まった．再帰の考え方は理解するのが難しかったが，文献［2］が参考になった．

参考文献

[1] 超初心者向けプログラミング入門，https://programming.pc-note.net/，2026 年 4 月 1 日参照．

[2] 柴田 望洋，“新· 明解 C 言語 入門編，”ソフトバンククリエイティブ，2015.

コメントの追加 [SK28]: 問題点，改良の余地などについて「考察」するための章であり，主観的な「感想」は記載しないこと．

コメントの追加 [SK29]: 参考文献の章にある文献は必ず本文中のどこかで参照し，文献番号を明示すること．

コメントの追加 [SK30]: 参考文献の章は一般的には章番号を付与しないが，記載方法は各課題の様式に従うこと．

コメントの追加 [SK31]: 講義資料の URL は基本的には記載しない．
書籍の場合，著者名，書名，出版社，発行年の順に記載するのが一般的である．
サイトの場合，引用ページのタイトル，URL，参照年月日の順に記載するのが一般的である．