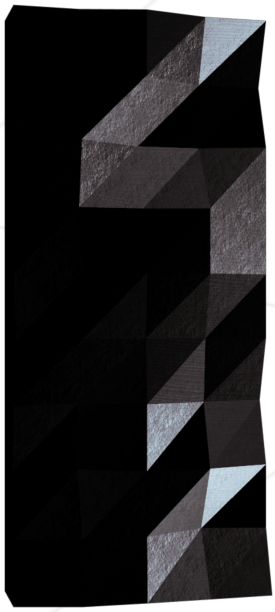

C Piscine

C 05

Summary: このドキュメントは、*42*の*C Piscine C 05*モジュール用の課題です。

Version: 7

Contents

Chapter I

Instructions

- このページのみが正式な課題の指示です。噂に惑わされないようにしてください。
- この課題は提出前に変更される可能性がありますので、気をつけてください！
- ファイルとディレクトリへの適切な権限があることを確認してください。
- すべての課題で、**提出手順**に従ってください。
- 提出した課題の評価（レビュー）は、あなたの周りにいるPiscine受験者によって行われます。
- さらに、**Moulinette**（自動採点プログラム）にも評価されます。
- **Moulinette**は非常に厳格かつ厳密に評価を行います。完全に自動化されており、交渉の余地はありません。思わぬ減点を避けるためにも、細部まで丁寧に取り組んでください。
- **Moulinette**は融通が利きません。あなたのコードがNormに準拠していない場合、理解しようとしません。**Moulinette**は**norminette**というプログラムを使用して、ファイルがNormに準拠しているかチェックします。要約：**norminette**のチェックに通らない課題を提出しても意味がありません。
- 課題は難易度が低いものから高いものへと難易度順に並んでいます。前の課題が正しく解けていない場合、後の課題が解けていても評価されません。
- 禁止されている関数を使用することは不正行為とみなされます。不正行為者は**-42**という成績がつけられ、これは交渉の余地がありません。
- 私たちが具体的に**プログラム**を求める場合のみ、**main()**関数を提出する必要があります。
- **Moulinette**は以下のフラグでコンパイルします：**-Wall -Wextra -Werror**、**cc**を使用します。
- プログラムがコンパイルできない場合、**0**の評価になります。
- 課題で指定されたファイル以外は、ディレクトリに絶対に残さないでください。

- 質問がある場合は、まず右隣の仲間に聞いてみましょう。いなければ左隣の仲間に聞いてみてください。
- あなたの助けになる参考資料は**Google / man / インターネット / ...**です。
- Intranetのフォーラムの「C Piscine」セクションをチェックしてください。
- 課題の例（出力例）をよく読んでください。課題の説明文だけではすぐに分からない要件が隠れていることがあります。
- 頭を使ってください!!!

Norminetteは、 *-R CheckForbiddenSourceHeader* をオプションに追加しなければなりません。Moulinetteも同様にこのオプションを使用します。

Chapter II

AI Instructions

Context

C Piscineは強烈な体験です。42で迎える最初の大きな挑戦として、問題解決、自律性、そしてコミュニティの世界に深く飛び込むことになります。

このフェーズの主な目的は、試行錯誤や反復、そして特に**ピアラーニング**を通じた交流によって、自分の土台を築き上げることです。

AI時代において、近道を見つけることは簡単です。しかし、AIの利用が本当にあなたの成長を助けているのか、それとも真のスキルを身につける上で障害となっていないか、よく考えることが大切です。

Piscineはまたとない人との繋がりを経験する場でもあります。今のところ、それに取って代わるものは何もありません。AIでさえも。

学習ツールとして、ICTカリキュラムの一環として、そして労働市場での期待の高まりに応えるものとして、私たちのAIに対する考え方の詳細については、Intranet上の専用FAQをご覧ください。

主なメッセージ

- 近道をせず、強固な基礎を築く。
- 技術力と実践力を真に養う。
- 真のピアラーニングを体験し、学び方や新たな問題の解決方法を学び始める。
- 結果よりも学習の過程が重要である。
- AIに伴うリスクを学び、一般的な落とし穴を避けるための効果的な管理方法と対策を身につける。

学習者のルール:

- 課題には、AIに頼る前にまず自分の頭で考えること。
- AIに直接的な答えを求めないこと。
- AIに対する42のグローバルな方針を学ぶこと。

このフェーズでの到達目標:

この基礎フェーズを終えることで、以下の点を達成できます。

- 技術とコーディングの適切な基礎を身につける。
- この段階においてAIが危険となりうる理由と、その危険性を理解する。

コメントと具体例:

- 私たちはAIの存在を知っていますし、それがあなたのプロジェクトを解決できることも知っています。しかし、あなたは学ぶためにここにいるのであって、AIが学習したことを証明するためではありません。AIが特定の問題を解決できることを示すためだけに、あなた（や私たち）の時間を無駄にしないでください。
- 42での学習は、答えを知ることではありません。答えを見つけ出す能力を養うことです。AIは直接答えを与えてくれますが、それはあなた自身の論理的思考を妨げます。そして、論理的思考には時間と努力、そして失敗が伴います。成功への道は、決して楽なものではありません。
- Exam中はAIが利用できないことを心に留めておいてください。インターネットもスマートフォンもありません。学習過程でAIに過度に頼りすぎていた場合、すぐにそのことに気づくでしょう。
- ピアラーニングは、多様な考え方やアプローチに触れる機会を与え、対人スキルや多角的な思考能力を向上させます。これは、単にボットとチャットするよりもはるかに価値があります。ですから、恥ずかしがらずに、話しかけ、質問し、共に学びましょう！
- AIは学習ツールとして、またそれ自体がトピックとして、カリキュラムの一部になります。独自のAIソフトウェアを開発する機会さえあります。私たちの段階的なアプローチについてさらに学ぶためには、Intranetで利用可能なドキュメントを参照してください。

✓ 良い実践例:

> 新しいコンセプトの学習でつまづいています。近くの人にどうアプローチしたか尋ねてみます。10分ほど話していると、突然ひらめき、理解できました。

✗ 悪い実践例:

> こっそりAIを使い、良さそうなコードをコピーします。ピアレビューの際には何も説明できず、不合格になりました。ExamではAIがなく、また行き詰まり、不合格になりました。

Chapter III

Foreword

Here is an excerpt from the lyrics of the *Harry Potter* saga:

```
Oh you may not think me pretty,
But don't judge on what you see,
I'll eat myself if you can find
A smarter hat than me.

You can keep your bowlers black,
Your top hats sleek and tall,
For I'm the Hogwarts Sorting Hat
And I can cap them all.

The Sorting Hat, stored in the Headmaster's Office.
There's nothing hidden in your head
The Sorting Hat can't see,
So try me on and I will tell you
Where you ought to be.

You might belong in Gryffindor,
Where dwell the brave at heart,
Their daring, nerve, and chivalry
Set Gryffindors apart;

You might belong in Hufflepuff,
Where they are just and loyal,
Those patient Hufflepuffs are true
And unafraid of toil;

Or yet in wise old Ravenclaw,
If you've a ready mind,
Where those of wit and learning,
Will always find their kind;

Or perhaps in Slytherin
You'll make your real friends,
Those cunning folks use any means
```

```
To achieve their ends.

So put me on! Don't be afraid!
And don't get in a flap!
You're in safe hands (though I have none)
For I'm a Thinking Cap!
```

Unfortunately, this subject has nothing to do with the *Harry Potter* saga, which is a shame, because your exercises won' t be completed by *magic*!

Chapter IV

Exercise 00 : ft_iterative_factorial

	Exercise 00
ft_iterative_factorial	
提出するディレクトリ：$ex00/$	
提出するファイル：`ft_iterative_factorial.c`	
使用可能な関数：`None`	

- 引数として与えられた数の階乗を返す反復関数を作成してください。
- 引数が無効な場合は、0を返すようにしてください。
- オーバーフローを対応する必要性はありません。オーバーフローが発生した場合、関数の戻り値は未定義となります。
- プロトタイプ例：

```
int ft_iterative_factorial(int nb);
```

Chapter V

Exercise 01 : ft_recursive_factorial

	Exercise 01
ft_recursive_factorial	
提出するディレクトリ：$ex01/$	
提出するファイル：`ft_recursive_factorial.c`	
使用可能な関数：`None`	

- 引数として与えられた数の階乗を返す再帰関数を作成してください。
- 引数が無効な場合は、0を返すようにしてください。
- オーバーフローを対応する必要性はありません。オーバーフローが発生した場合、関数の戻り値は未定義となります。
- プロトタイプ例：

```
int ft_recursive_factorial(int nb);
```

Chapter VI

Exercise 02 : ft_iterative_power

	Exercise 02
ft_iterative_power	
提出するディレクトリ：$ex02/$	
提出するファイル：`ft_iterative_power.c`	
使用可能な関数：`None`	

- 引数として与えられた数のべき乗の値を返す反復関数を作成してください。
- べき乗が0未満の場合は、0を返すようにしてください。
- オーバーフローを対応する必要性はありません。
- 0の0乗は、1を返すようにしてください。
- プロトタイプ例：

```
int ft_iterative_power(int nb, int power);
```

Chapter VII

Exercise 03 : ft_recursive_power

	Exercise 03
ft_recursive_power	
提出するディレクトリ：$ex03/$	
提出するファイル：`ft_recursive_power.c`	
使用可能な関数：`None`	

- 引数として与えられた数のべき乗の値を返す再帰関数を作成してください。
- べき乗が0未満の場合は、0を返すようにしてください。
- オーバーフローを対応する必要性はありません。オーバーフローが発生した場合、関数の戻り値は未定義となります。
- 0の0乗は、1を返すようにしてください。
- プロトタイプ例：

```
int ft_recursive_power(int nb, int power);
```

Chapter VIII

Exercise 04 : ft_fibonacci

	Exercise 04
ft_fibonacci	
提出するディレクトリ：$ex04/$	
提出するファイル：`ft_fibonacci.c`	
使用可能な関数：`None`	

- フィボナッチ数列のn番目の要素を返す関数`ft_fibonacci`を作成してください。
  最初の要素は0番目の配置とし、フィボナッチ数列は、0, 1, 1, 2 のような順序で始まるものとします。
- オーバーフローを対応する必要性はありません。オーバーフローが発生した場合、関数の戻り値は未定義となります。
- プロトタイプ例：

```
int ft_fibonacci(int index);
```

- `ft_fibonacci`は再帰的に実装してください。
- `index`が0未満の場合、関数は-1を返すようにしてください。

Chapter IX

Exercise 05 : ft_sqrt

	Exercise 05
ft_sqrt	
提出するディレクトリ：$ex05/$	
提出するファイル：`ft_sqrt.c`	
使用可能な関数：`None`	

- 与えられた数に整数の平方根がある場合はその値を返し、平方根が無理数の場合は0を返す関数を作成してください。
- プロトタイプ例：

```
int ft_sqrt(int nb);
```

Chapter X

Exercise 06 : ft_is_prime

	Exercise 06
ft_is_prime	
提出するディレクトリ：$ex06/$	
提出するファイル：`ft_is_prime.c`	
使用可能な関数：`None`	

- 引数として与えられた数が素数である場合は1を返し、それ以外の場合は0を返す関数を作成してください。
- プロトタイプ例：

```
int ft_is_prime(int nb);
```

0と1は素数ではありません。

Chapter XI

Exercise 07 : ft_find_next_prime

	Exercise 07
ft_find_next_prime	
提出するディレクトリ：$ex07/$	
提出するファイル：`ft_find_next_prime.c`	
使用可能な関数：`None`	

- 引数として与えられた数が素数である場合は同じ数字を返し、それ以外の場合は、その数より大きい次の素数を返す関数を作成してください。
- プロトタイプ例：

```
int ft_find_next_prime(int nb);
```

Chapter XII

Exercise 08 : The Ten Queens

	Exercise 08
The Ten Queens	
提出するディレクトリ：$ex08/$	
提出するファイル：`ft_ten_queens_puzzle.c`	
使用可能な関数：`write`	

- 10x10のチェス盤上で、10体のクイーンが互いに1手で攻撃できないように配置するすべてのパターンを標準出力に出力し、その総数を返す関数を作成してください。
- この問題を解くためには再帰処理が必要です。
- プロトタイプ例：

```
int ft_ten_queens_puzzle(void);
```

- 出力例：

```
$>./a.out | cat -e
0257948136$
0258693147$
...
4605713829$
4609582731$
...
9742051863$
$>
```

- 列は左から右に進みます。
  - 最初の数字は、1列目のクイーンの行位置を表します。（インデックスは0から始まります）
  - N番目の数字は、N列目のクイーンの行位置を表します。
- 関数は、出力結果の総数を返すようにしてください。

Chapter XIII

Submission and peer-evaluation

課題は、いつも通りGitリポジトリに提出してください。リポジトリ内の提出物のみが、レビュー中の評価対象となります。ファイル名が正しいかどうか、必ず確認してください。

この課題の指示で、明示的に求められたファイルのみを提出してください。