
卒業研究テキスト

椋山女学園大学現代マネジメント学部 三木 邦弘
令和5年4月7日版

1	ChatGPT	2	3	運用コスト	18
1.1	ゲームプログラムを作る	3	3.1	様々なレベルのサービス	19
1.2	高級かな漢字変換	9	3.2	紹介サイトの構築	20
1.3	CharGPT の実行方法	10	4	アフィリエイト	21
2	卒論の書き方	14	4.1	どのくらい儲かるのか?	21
2.1	卒論用フォルダーについて	14	4.2	ASP への登録	22
2.2	タイトルや要旨など	14	4.3	Amazon や楽天への登録	22
			4.4	Google AdSense	23

よく分からない事も答えてますが、追加学習なしのものの回答と比べるとともです。追加学習データにあった「いい天気だね」を問いかけると、

雨が心配な方はお気を付けください。

そうですね。晴れ男

それはよかったね！

のように返ってきました。教えた通りの答えはなかなか返ってきません。一方回答を見ていると、なんとなく回答者の雰囲気を感じられます。500 例ほどで回答者の振りができると言うのはなぜでしょうね。

表 1.1 会話テキストデータの最初の 5 件と「天気」が出てくる 2 件

質問	答え
お腹が鳴る	何か召し上がりますか？
だるい	それは心配です。私にできることがあれば、何でもお申し付けください。
眠れない	何も心配する必要はありません。私がついています。ずっと、ずっと、一緒です。
押すなよ、絶対押すなよ	はい、絶対押しません！
見せられないよ	目をつぶって待っています。
いい天気だね	それは良かったです！ 月は見えますか？
明日の天気は？	WEB 検索してみてもいいかがでしょうか。

1.1 ゲームプログラムを作る

昔コンピュータが初めて登場した時から、コンピュータを利用するにはプログラムを作らなければならない、という問題がありました。最初は利用者がプログラムを作っていました。やがてコンピュータの利用が広がるに従い、プログラムを作る人と使う人が分かれてきました。Excel などの表計算ソフトがなかった頃は、簡単な計算もプログラムの作成が必要でした。表計算ソフトが利用できるようになり、簡単な計算を行うことは容易になりましたが、複雑な計算をさせるためには表計算ソフトの使い方を学ぶ必要があります。最近日本では小学生からプログラミングを学ばせるようになりましたが、表計算ソフトも十分使いこなせない人が多いことから、プログラミング嫌いを大量生産するのではないかと心配もあります。一方 ChatGPT の助けがあれば、誰でもプログラムを作ることができるようになるかもしれません。

ここでは簡単なゲームプログラムを ChatGPT を利用して作成する例を紹介します。日経 XTECH の 2023 年 8 月 10 日の大森敏行による「ChatGPT でシューティングゲームを自作、欲しいソフトを誰でもつくれる時代に」^{*3}の記事をもとにしています。ただし元の記事はプログラミング言語として python を使用したものでしたが、「プログラミング基礎」の授業で JavaScript をやっていますので、JavaScript で作成するものにしました。また元の記事ではどのバージョンの ChatGPT を使用したのか書かれていませんでしたが、これは「gpt-4-0613」というバージョンのものを使用しています。

質問: JavaScript を使ってシューティングゲームを作りたいと思います。どのような手順で作ったら良いでしょうか (52 tokens)

^{*3} <https://xtech.nikkei.com/atcl/nxt/column/18/00682/080700130/>

回答：シューティングゲームを JavaScript で作る際の基本的な手順は以下の通りです：

1. 環境設定：JavaScript の開発に必要なツールをインストールします。最低限、テキストエディタ（例：VS Code）とウェブブラウザ（例：Chrome）が必要です。
 2. 設計：ゲームの詳細な設計を行います。どのようなキャラクターや敵がいて、どのように動くのか、スコアはどのように計算されるのか等を決定します。
 3. コーディング：JavaScript でゲームのコードを書き始めます。以下は一般的なステップです：
 - 画面（キャンバス）の作成：HTML5 の<canvas>要素を使用してゲーム画面を作成します。
 - キャラクター・敵・その他のオブジェクトの作成：オブジェクト指向プログラミングを用いてゲーム内の各オブジェクト（キャラクター、敵、弾、背景など）を定義します。
 - キーボード・マウス入力処理：JavaScript のイベントリスナーを使用して、ユーザーからの入力（キーボードやマウスの操作）を取得します。
 - ゲームループの作成：ゲームの進行を制御するためのゲームループを作成します。これは一定間隔で画面を更新し、オブジェクトの動きなどゲームの状態を更新するものです。
 - 衝突判定の実装：キャラクター、敵、弾などのオブジェクト同士の衝突を検出し、それぞれに応じた動きや結果（敵を倒す、ダメージを受ける等）を実装します。
 - スコア計算・画面表示の実装：スコアの計算方法の実装と、それを画面に表示するロジックを作成します。
 4. デバッグとテスト：コーディングが完了したら、ゲームが正しく動作するかテストします。エラーを修正し、パフォーマンス問題を解決します。
 5. フィードバックと改善：プレイテストを行い、ユーザーからのフィードバックをもとにゲームを改善していきます。
- 以上が基本的なステップとなります。ただし、実際には詳細な機能やゲームのルールによりステップは変わる可能性があります。最後に、ゲーム開発は簡単なものから始めて徐々に難易度を上げていくと良いでしょう。（833 tokens）

質問：キー入力で左右に動かせる青い自機を表示し、キー入力で緑色の弾を撃てるようにしてください（939 tokens）

回答：以下に、HTML Canvas と JavaScript を使用してキー入力で自機を左右に動かし、弾を撃つ機能を実装した基本的なコード例を示します。

```
html
<!DOCTYPE html>
<html>
  <head>
    <title>シューティングゲーム</title>
    <style>
      body { background: #000; color: #fff; }
      #gameCanvas { background: #fff; }
    </style>
  </head>
  <body>
```

```
<canvas id="gameCanvas" width="800" height="600"></canvas>
<script>
  var canvas = document.getElementById('gameCanvas');
  var ctx = canvas.getContext('2d');

  var ship = { x: 400, y: 550, width: 50, height: 50, color: 'blue' };
  var bullets = [];
  var bulletSpeed = 5;

  function drawShip() {
    ctx.fillStyle = ship.color;
    ctx.fillRect(ship.x, ship.y, ship.width, ship.height);
  }

  function drawBullets() {
    ctx.fillStyle = 'green';
    bullets.forEach(function(bullet) {
      ctx.fillRect(bullet.x, bullet.y, bullet.width, bullet.height);
    });
  }

  function updateBullets() {
    bullets.forEach(function(bullet) {
      bullet.y -= bulletSpeed;
    });
    bullets = bullets.filter(function(bullet) {
      return bullet.y > 0;
    });
  }

  function animate() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
    drawShip();
    drawBullets();
    updateBullets();
    requestAnimationFrame(animate);
  }

  animate();

  window.addEventListener('keydown', function(e) {
    if(e.code === 'ArrowLeft' && ship.x > 0) {
      ship.x -= 10;
    }
  });
</script>
```

```

    }
    if(e.code === 'ArrowRight' && ship.x < canvas.width - ship.width) {
        ship.x += 10;
    }
    if(e.code === 'Space') {
        bullets.push({ x: ship.x + ship.width / 2 - 2.5, y: ship.y, width: 5, height: 10 });
    }
  });
</script>
</body>
</html>

```

このコードは、キャンバス上に青い自機を表示し、矢印キーを押すと左右に動くようにしています。同時に、スペースキーを押すと自機から緑色の弾が発射されるようになっています。なお、左右の移動は画面から出ないように制限を入れています。また、発射する弾は一定のスピードで上に移動し、画面から出た弾は自動的に削除されます。さらに進んだゲームの要素（敵キャラクターの導入、衝突判定、得点計算など）を取り入れるには、より詳細なコードが必要となります。（720 tokens）

質問：ふらふらと左右に動く赤い敵を 1 つ表示し、敵も黒い弾をランダムに発射するようにしてください（1714 tokens）

回答：前回と同じテンプレートを継続し、敵と黒の弾を追加します。フラフラと動く敵とその弾を作りましょう。新たに追加、または変更になる部分： - 敵のオブジェクトと弾を追加します。 - 敵がフラフラと左右に動くように更新します。 - ランダムに敵が弾を撃つようにします。

```

html
<script>
  // コードの初め部分は前回と同じため省略します...

  var enemy = { x: 375, y: 50, width: 50, height: 50, color: 'red', direction: 1 };
  var enemyBullets = [];
  var enemyBulletSpeed = 2;
  var enemySpeed = 2;

  function drawEnemy() {
    ctx.fillStyle = enemy.color;
    ctx.fillRect(enemy.x, enemy.y, enemy.width, enemy.height);
  }

  function drawEnemyBullets() {
    ctx.fillStyle = 'black';
    enemyBullets.forEach(function(bullet) {
      ctx.fillRect(bullet.x, bullet.y, bullet.width, bullet.height);
    });
  }

```

```
});  
}  
  
function updateEnemyBullets() {  
  enemyBullets.forEach(function(bullet) {  
    bullet.y += enemyBulletSpeed;  
  });  
  enemyBullets = enemyBullets.filter(function(bullet) {  
    return bullet.y < canvas.height;  
  });  
}  
  
function updateEnemy() {  
  enemy.x += enemySpeed * enemy.direction;  
  if (enemy.x < 0 || enemy.x + enemy.width > canvas.width) {  
    enemy.direction *= -1;  
  }  
  if (Math.random() < 0.01) {  
    enemyBullets.push({ x: enemy.x + enemy.width / 2 - 2.5, y: enemy.y + enemy.height, width: 5, height: 5, direction: 1 });  
  }  
}  
  
function animate() {  
  // これまでのコードは前回どおりです...  
  
  drawEnemy();  
  drawEnemyBullets();  
  updateEnemyBullets();  
  updateEnemy();  
  requestAnimationFrame(animate);  
}  
  
animate();  
  
// これ以降のコードは前回どおりです...  
</script>
```

このコードは前回のコードに敵と敵の弾の描画と更新を追加しています。敵は左右に動き、壁にぶつくと反対向きに跳ね返ります。また、一定の確率で弾を撃つようになっています。その確率は 'Math.random() < 0.01' で制御されており、この値を調整することで敵の弾の発射頻度を変更することができます。私の解答が役に立てば幸いです。なお、衝突判定等の拡張機能は自分で調整、または別途質問してください。(727 tokens)

質問: 自分の弾が敵に当たったら『You win』、敵の弾が自分に当たったら『You lose』と表示してゲームが終了するようにしてください (2503 tokens)

回答: 自機と敵の弾、敵と自機の弾の衝突判定とゲームの終了条件を追加しましょう。衝突判定は各オブジェクトの座標と大きさを比較することで行います。そして、衝突があった場合は「You win」または「You lose」と表示し、ゲームループを停止します。以下が変更後のコードです:

```
html
<script>
  // コードの初め部分は前回と同じため省略します...

  var gameOver = false;
  var result = '';

  // updateBullets, updateEnemyBullets, updateEnemy の後に次の関数を追加します:

  function checkCollision() {
    bullets.forEach(function(bullet) {
      if (bullet.x < enemy.x + enemy.width &&
          bullet.x + bullet.width > enemy.x &&
          bullet.y < enemy.y + enemy.height &&
          bullet.y + bullet.height > enemy.y) {
        result = 'You win!';
        gameOver = true;
      }
    });

    enemyBullets.forEach(function(bullet) {
      if (bullet.x < ship.x + ship.width &&
          bullet.x + bullet.width > ship.x &&
          bullet.y < ship.y + ship.height &&
          bullet.y + bullet.height > ship.y) {
        result = 'You lose!';
        gameOver = true;
      }
    });
  }

  function animate() {
    // これまであったコードは前回どおりです...

    checkCollision();
    if (!gameOver) {
```

```
requestAnimationFrame(animate);
} else {
  ctx.font = '30px Arial';
  ctx.fillText(result, canvas.width / 2, canvas.height / 2);
}
}

animate();

// これ以降のコードは前回どおりです...
</script>
```

このコードは前回のコードに衝突判定とゲーム終了条件を追加しています。衝突が検出された場合は「gameOver」変数を true に設定し、ゲームループを停止します。そして結果（勝ちまたは負け）を画面の中央に表示します。衝突判定のロジックを一部説明します。これは二つの矩形が衝突しているかどうかを決定する標準的なアルゴリズムで、次のように機能します：1. 弾の右端が敵の左端より右にある。2. 弾の左端が敵の右端より左にある。3. 弾の下端が敵の上端より下にある。4. 弾の上端が敵の下端より上にある。すべての条件が真であれば、弾と敵は衝突していると判断します。同様のロジックが自機と敵の弾の衝突判定にも適用されます。（762 tokens）

質問や回答の最後に「762 tokens」のように token の数が付いています。ChatGPT の内部では文字ではなく、ほぼ単語と同等である token という形で処理を行っています。使用料はこの token の数に比例して掛かります。数字を見ると回答の token の数は 700～900 に対して、質問の方はどんどん増えています。これは毎回これまでのやり取りを今回の質問の前にこっそり追加しているためです。この追加を行わないと、これまでの回答と無関係な回答になるために、特にプログラム作成には使えません。残念ながら質問に許される token の数には制限があり、長いやり取りになると、最初の方の部分は切り捨てられてしまいます。それを避けるためもあり、プログラムの内容が変更部分だけ示されるようです。このように JavaScript についての知識が全く無いと無理ですが、初級レベルでもあれば簡単なゲームぐらいいはなんとかなりそうです。

今回この章の例を作成するためにかなり試行錯誤を行いました。やり直す度に違うやり方の回答があったり、白地に白い自機を表示したり、バグがあって弾が出ない、敵がどこかへ行ってしまっただけで帰って来ないなど色々ありました。とりあえずはプログラムの一部についてやり方を尋ねる、というような使い方が無難でしょう。やがて改良が進み問題のないプログラムが回答されるようになると思いますが、どのようなプログラムを作りたいのかイメージを持ってないと使えないでしょう。

1.2 高級かな漢字変換

ChatGPT は広く知られるようになってからまだ余り時間が経っていないので、一言でこれを示す言葉はありません。ChatGPT は「高級かな漢字変換」である、と言うのも一つの候補ではないかと思います。昔々コンピュータがアメリカからやってきと時、これで如何に日本語を扱うかが問題になりました。日本語の文字をどのような数値に対応させるか、日本語の文字をどのように表示・印刷させるか、日本語の文字をどのように入力するか。最後の問題に関しては「かな漢字変換」で対応してきました。読みを全て入力して正しい漢字に置き換えるという方式です。漢字には同音異字があるため、誤変換を完全に無くすことは不可能ですが、ある程度文の意味が読み取れるぐらいの長さの読みを入れると、かなり正確な変換が可能です。これが携帯端末が普及してくる頃には、短い読みを入力すると、その続きを含めて漢字に置き換えたものを示すようになりまし

た。読みを全て入力していた時代から、一部を入力すれば良い時代になったのです。そして ChatGPT によって、短い語句を入力するだけで長い文章が出てくるようになりました。かな漢字変換が更に進化したと言えます。

ChatGPT も「かな漢字変換」と考えると、その問題点や解決法も見えてきます。ChatGPT も時には意図しない結果を出します。これは誤変換に相当します。これを避けるには、「かな漢字変換」の時に長い読みを入れたように、十分な指示を与えます。また少ない入力を長い文章にするために盛んに話を盛ってきます。大量に学習したよくある話を入れてくるので、指示していない部分には注意します。現在のところ ChatGPT は一連のやり取りの間は、何を返したか覚えていますが、次回は忘れていきます。一番古い「かな漢字変換」でも、前回選択した同じ読みの漢字は忘れずに最初の候補として出てくるので、ChatGPT も見習って欲しいところです。

テレビドラマで会社の事務員の方は、ずっとパソコンに向かってお仕事をしています。ChatGPT で仕事内容にピッタリ合うプログラムを作成すれば、スタートボタンをクリックするだけで済むとか、ChatGPT に「いつもの文書」と指示するだけで文書ができるようになると楽になりますね。

1.3 CharGPT の実行方法

ChatGPT を実行するには、ChatGPT を開発した OpenAI に利用者登録をしてブラウザ経由で会話的に使用する他、API を利用して要求を送るという方法があります。ChatGPT を利用したシステムを作成するには後者の方法を使うしかありませんし、API での利用の際のやり取りは学習データとして利用しないと OpenAI は言っています。API の利用方法はこれまで紹介した他の API と同様の HTTP を利用してやり取りをするものです。リスト 1 は PHP を使用して問を 1 つ送り、3 つの回答を得るものです。

リスト 1 ChatGPT の API の利用例

```
1 $apikey="...";
2 $url="https://api.openai.com/v1/chat/completions";
3 $model="gpt-3.5-turbo";
4 $max_tokens=2000;
5 $n=3; // 回答の数
6 $data = array(
7     'model' => $model,
8     'messages' => [
9         ['role' => "system", 'content' => "関西弁で話して"],
10        ['role' => "user", 'content' => "マクロ経済学はどんな学問ですか？"],
11    ],
12    'max_tokens' => $max_tokens,
13    'n' => $n
14 );
15 $options=array(
16     'http' => array(
17         'method' => "POST",
18         'header' => "Content-Type: application/json\r\n".
19             "Authorization: Bearer ".$apikey,
20         'content' => json_encode($data),
21         'timeout' => 180
22     )
23 );
24 $response=@file_get_contents($url,false,stream_context_create($options));
25 if ($response===false) die("Error!");
```

```

26 $result=json_decode($response, true);
27 for ($i=0;$i<$n;$i++) {
28     echo "<H2>応答</H2>\n";
29     echo "<Div>{$result[\"choices\"][$i][\"message\"] [\"content\"]}</Div>\n";
30 }

```

- 1 行 \$apikey には OpenAI から取得した API のキーを設定します。これによって API の使用料が課金されますので、他の人に知られないように注意します。
- 3 行 \$model には利用する ChatGPT のモデルを指定します。2023 年 9 月に利用できるモデルは表のようになっています。

表 1.2 ChatGPT のモデル

モデル名	作成日	使用料 (1,000 トークンあたりのドル)
gpt-3.5-turbo	2023/03/01	0.0015 ~ 0.002
gpt-3.5-turbo-0301	2023/03/01	0.0015 ~ 0.002
gpt-3.5-turbo-0613	2023/06/13	0.0015 ~ 0.002
gpt-3.5-turbo-16k	2023/05/11	0.003 ~ 0.004
gpt-3.5-turbo-16k-0613	2023/05/31	0.003 ~ 0.004
gpt-4	2023/06/28	0.03 ~ 0.06
gpt-4-0314	2023/06/28	0.03 ~ 0.06
gpt-4-0613	2023/06/13	0.03 ~ 0.06

0301 などの日付が付いたモデルは、作成された時点のままのものです。16k の付いたものは扱えるトークンの量が多いものです。gtp-3.5 より gtp-4 の方が高性能です。

- 6-14 行 API に送る内容を\$data に設定しています。9 行目で設定情報、10 行目で質問を設定しています。後述のようにこの部分是对話を繰り返す場合は多くなります。

- 15-22 行 API に送る方法の設定などを行っています。API は JSON 形式のデータでないと受け付けないので、20 行目で変換もしています。21 行目で応答をどのくらい待つかが指定しています。この例では 180 秒にしていますが、余り短い時間にすると OpenAI のサーバーが混んでいると結果が得られないことがあります。

- 24 行 OpenAI に要求を送り、\$response に応答を入れています。@file_get_contents() のように関数名の前に「@」を付けることにより、エラーが生じてもここで中断しないようにしています。

- 26 行 OpenAI からの応答も JSON 形式なのでここで PHP で扱える配列の形式に変換しています。

- 29 行 応答が\$result["choices"][0]["message"]["content"] ~ \$result["choices"][2]["message"]["content"] に入っているため HTML のタグで囲って表示しています。ここでは使っていませんが、\$result["usage"]["prompt_tokens"] に送ったトークンの数、\$result["usage"]["completion_tokens"] に応答のトークンの数が入っています。また\$result["choices"][0]["finish_reason"] に「stop」が入っていれば正常終了ですが、「length」ならばトークン数が多くなりすぎた、「content_filter」ならば質問内容が不適当だった、「null」は処理中を示しています。

リストに色々足りない部分を追加すると、図 1.1 のような結果が得られます。

リスト 1 は ChatGPT に問いかける内容がプログラム内に記述されているので、このままでは同じ問いかけしかできませんし、異なる問いかけをする場合はプログラムを修正しなければなりません。入力用のページで入力された内容を、別の PHP のファイルで ChatGPT に送って応答を表示するという形が一般的でしょう。その場合 ChatGPT に送ってから応答が返るまでに時間がかかります。これまでのやり方、つまり Form タグ



図 1.1 API を使った ChatGPT の実行

の間の入力欄に入力してもらって、submit タイプのボタンで入力内容を送る場合、ボタンをクリックすると PHP のプログラムから応答が来るまで、ブラウザの表示内容は変わりません。これは PHP が送るべき内容がある程度溜まってから、まとめて送ろうとするためです。利用者は何も変わらないので、ちゃんと送信できたのだろうかと不安を感じます。

その解消には例えば次のような方法があります。処理待ちの間に表示させるものに、最初には表示しないというスタイル (display: none) を設定します。次の例は画像に対して設定しています。

```
<Img src='kurukuru.gif' id="kurukuru" style="display: none">
```

また後で JavaScript でスタイルを変更しやすいように id として「kurukuru」を指定しています。同様に送信ボタンにも id として「sendbtn」を指定したとします。そして送信の際は次のような go() を呼び出します。

```
function go() {
    document.getElementById('sendbtn').style.display="none";
    document.getElementById('kurukuru').style.display="inline";
    document.aaa.submit();
}
```

最初の行で送信ボタンのスタイルを「none」にしているので送信ボタンが消えます。そして次の行で処理待ちの画像のスタイルを「inline」にしているので、画像が表示されます。getElementById() はこのように id を指定したオブジェクトを取り出す関数です。name と違って web ページの中で id は一意に指定する必要があります。最後に送信を実行しています。

また送信を受けた方で対応することも可能です。PHP がまとめてブラウザに送ろうとするのを、途中で送らせる事ができます。何も送らない段階ではブラウザには既に表示された内容が残っていますが、途中で送ると表示がクリアされて、送った分だけすぐ表示^{*4}されます。次の例は処理待ちの画像を表示するタグを echo で送っていますが、この後で ChatGPT の応答待ちなどがあると、その結果が出るまでブラウザには送られません。この例では ob.flush() と flush() により PHP が貯めていた分とシステムが貯めていた分を即座にブラウザに送るので、ブラウザに画像 (kurikuru.gif) が表示されます。

```
echo "<Img src='kurukuru.gif'><Br>";
ob_flush();
flush();
```

Chat をする方法

Chat は単なる問いかけと答えではありません。それ以前の問いとその答えを踏まえたものでなければなりません。そのために Chat をする場合は、これまでのやり取りを毎回送る必要があります。リスト 1 の 9 行目では、「role」として「system」を指定することにより、状況の設定を行っています。また 10 行目では「role」として「user」を指定することにより、問いかけを行っています。同様に「role」として「assistant」を指定して過去の応答を送ることができます。例えば次のようにします。

```
['role'=>"user", 'content' => "ピンク色に合う色を 3 つ教えてください。"],
['role'=>"assistant", 'content' => "ピンク色に合う色は、次の 3 つです： グレー、白、紫。"],
['role'=>"user", 'content' => "なぜその色が合うのか理由を教えてください。"],
```

1 行目の問いを ChatGPT に送り、その応答を 2 行目で指定して、3 行目の問いを送ろうとしています。「その色」は何色なのか、どの色と合うかを尋ねているのかは、1 行目と 2 行目から分かります。その結果以下のような応答^{*5}が得られました。

ピンク色は、一般的に女性らしさや優しさを象徴する色とされています。それに対して、グレーは中立的で落ち着いた色ですので、ピンクとの相性が良いとされています。グレーはピンクの華やかさを引き立たせることができます。白は純粋さや清潔さを象徴する色ですので、ピンク色と相性が良いとされています。ピンク色と一緒に使うことで、より柔らかく明るい雰囲気を作り出すことができます。紫はロマンチックな色であり、ピンク色とも相性が良いです。ピンク色と紫を組み合わせることで、女性らしさや幻想的な雰囲気を演出することができます。これらの色が合う理由は、ピンク色の特性と相性が良く、ピンク色の魅力やイメージを引き立てる特徴を持っているからです。

Chat を続けていくと、送るべきこれまでの応答がどんどん増えていきます。その結果使用料もどんどん増えていきますし、やがて利用できるトークン数の上限に達してしまいますので、最初の方のやり取りを送らないようにするなどの対策が必要です。そうすると最初の方にあった Chat の前提となる条件などが失われる事につながります。

^{*4} 古いブラウザではある程度の量を送らないと表示されなかったので、表示されない半角の空白を大量に付けて送っていた事もありました。

^{*5} 使用したモデルは gpt-3.5-turbo-0613 です。gpt-4-0613 で尋ねたら、ホワイト、グレー、ネイビーブルーだそうです。

2. 卒論の書き方

この章では卒業論文を $\text{\LaTeX}$ でどのように記述すれば良いかについて説明します。昔卒論を担当していた時は、やはり $\text{\LaTeX}$ で卒業論文を記述してもらいました。卒論用スタイル設定ファイルを用意して、学部指定の形式になるようにし、また全員の原稿を集めて冊子を作成したりしていました。卒業論文は A4 サイズ、冊子の方は B5 サイズでしたが、冊子用スタイルに変更するだけで済みました。 $\text{\LaTeX}$ の基本的な使い方は「基礎演習」のテキストを参照してください。ここでは卒論用スタイル設定ファイルによって使える機能などについて説明します。

2.1 卒論用フォルダーについて

`mars` のデスクトップに「教材フォルダ」からサンプルフォルダ「S999」をコピーして、999 の部分を自分の学籍番号の下 3 桁に変更してください。冊子を作成する際に全員の原稿データを一箇所に集めるため、このような名前にしておいてください。サンプルフォルダの中には「`sotu23.tex`」と言うファイルがあるので、その内容を書き換える形で卒業論文を作成してください。ファイル名は変更しないでください。

2.2 タイトルや要旨など

卒業論文の形式は「卒業研究の手引」で次のように決まっています。(1) 指定ファイル表紙、(2) 要旨、(3) 中表紙、(4) 目次、(5) 卒論本文、(6) 参考文献リスト、(7) 指定ファイル裏表紙、となっています。(1) と (7) は現代マネジメント学部指定のファイルを学生会館 2 階の売店で購入して、自分で手書きする必要があります。購入は早めに、タイトルはギリギリで変わることもあるので、提出直前に書いた方が良いでしょう。また (2) は Word のファイルで別途 `S*map` で提出する必要があり、それも卒業論文と同じ日時が締切なので注意してください。

大昔は「卒業研究の手引」に原稿用紙 50 枚分以上ということが書かれていました。400 字 × 50 枚で 20,000 字に相当する量ですが、現在の「卒業研究の手引」でも 1 ページは 1,200 字程度となっているので 16.6 ページ分となります。その後この決まりが無くなったのは、20,000 字も書けないという学生が多いためだと思います。実際のところは図などが入りますし、国際コミュニケーション学部では、卒業論文は原稿用紙 100 枚分以上と言う話を聞いたことがあります。よって私のところでは本文を 17 ページ以上書くことにします。

それ以外の部分は、卒論用スタイル設定ファイルが適切な形にするので、形式の面で特に注意することはないでしょう。サンプルファイル (リスト 2) に従って、例えばサンプルファイルの「ここに 800 字程度の要旨を書け」とあるところに要旨を入力すれば、要旨が学部の指定の場所に指定の形に出てくるようになります。

リスト 2 卒論サンプルファイル

```

1 \documentclass[sotu23]{jsarticle}
2
3 \title{Web ベースの\mbox{タッチタイピング練習ソフトウェア}}
4 % または
5 %\dttitle{タッチタイピング練習ソフトウェア}{ー ブラウザでどこでも練習できる ー}
6 % 使わない方の先頭に %を付ける
7
8 \author[A20EA999]{梶山 花子}
9
10 \begin{document}
11
12 \maketitle % 表題のページの出力
13
```

```

14 \begin{abstract}
15 ここに 800字程度の要旨を書く
16 \end{abstract}
17
18 \tableofcontents % 目次の出力
19
20 \section{はじめに}
21
22 ここに卒論の本文を書く
23
24 \section{参考文献}
25
26 \begin{thebibliography}{99}
27   \bibitem{latex} Leslie Lamport 著、Edgar Cooke・倉沢良一監訳、
28     「文書処理システム\LaTeX」、株式会社アスキー、1990 年
29 \end{thebibliography}
30
31 \section{プログラムリスト}
32
33 \subsection{入り口のweb ページ: index.php}
34
35 \ListIn{../www/index.php}
36
37 \end{document}

```

- 卒業論文のタイトルは\title{ }で指定します。長いタイトルの場合、変なところで改行されることがあります。そのような場合は改行されたくない部分を\mbox{ }で囲ってください。サブタイトルがある場合は、\dttitle{ }{ }を使ってください。
- 参考文献の後にプログラムリストを付けます。プログラムリストは付録の扱いなので、卒業論文のページ数には入りません。\\ListIn{ }で取り込むファイルを指定します。ファイル名の前には必ず「../www/」を付けてください。もし現在作成中のファイルに index.php がない場合はエラーになるので、存在するファイルの名前に書き換えてください。
- 本文中でプログラムの説明を書く場合、この本文のずっと後ろにあるプログラムリストを参照する形では分かりにくいものになってしまいます。数行であれば、\\begin{FV} ~ \\end{FV}を利用して次のような形で出すことができます。

```

for ($i=0;$i<100;$i++) {
    echo "I love you.<Br>\n";
}

```

周りの枠が不要であれば、\\begin{V} ~ \\end{V}で次のような形にすることができます。

```

for ($i=0;$i<100;$i++) {
    echo "I love you.<Br>\n";
}

```

- 長いプログラムリストの場合は、説明文中で何行目の事を説明しているのか示さないと、似たような行があると分かりにくいことがあります。そのような場合は次のようにしてください。

```
\begin{lstlisting}[caption=卒論サンプルファイル,label=latex2-samp]
\documentclass[sotu22]{jsarticle}
  中略
\title{Web ベースの\mbox{タッチタイピング練習ソフトウェア}}
\end{document}
\end{lstlisting}
```

これでリスト 2 と同じように先頭に行番号が付いた形で出るようになります。また長いリストでページに入り切らない場合も、そのまま出てくるので余分な空きができません。

-  のように周りを囲いたい場合は、`\btn{Enter}` のように書きます。
-  のように文中にボタンなどの画像を出したい場合は、`\gbtn{ファイル名}` のように書きます。大きな画像でもこれぐらいの高さに縮小されて出ます。
- `\label{~}` と `\ref{~}` を使って図や表の参照をしましょう。「~」の部分に同じ名前を複数使わないように注意します。図や表は冊子にする際に位置が変わることがありますので「次の図」の文の前に図が来ることもあります。また図の順番や追加や削除も作文中によく生じるので「図 3」のように決め打ちするのは危険です。
- プログラムリストがちゃんと印刷されているか必ず確認してください。長い行は切れてしまいます。また括弧の対応を分かり易くしてください。プログラムリストの形を整えるページを作りましたので、よろしかったらご利用ください。「リスト整形プログラム」(<http://mars.mgt.sugiyama-u.ac.jp/plist/>) 使い方は成形したいプログラムを「整形」の下大きな入力欄に貼り付けて、「整形」をクリックするだけです。整形されたプログラムの上にある「リスト全体をコピーする」をクリックすると全選択とコピーが行われます。これを元のところに貼り付ければ貼り付ければ良いでしょう。

この整形プログラムは行の長さが許せばどんどん 1 行に詰め込む傾向があるので、詰め込みすぎたら、適当に改行を入れてください。また長い文字列 (“ ~”) を勝手なところで改行してくるので、それをそのまま利用すると PHP のエラーが出ることがあります。例えば次のようになったら、

```
'header' => "User-Agent: Mozilla/5.0 (Windows NT 6.2) AppleWebKit/537.1 (KHTML,
like Gecko) Chrome/21.0.1180.75 Safari/537.1\r\n"
```

区切りの良さそうな所で改行し、「”」のペアが行内に入るように「”」を追加しましょう。

```
'header' => "User-Agent: Mozilla/5.0 (Windows NT 6.2) AppleWebKit/537.1 " .
"(KHTML, like Gecko) Chrome/21.0.1180.75 Safari/537.1\r\n"
```

- URL は長いし記号が混ざるのでそのままでは綺麗に入りません。右の余白に飛び出したりします。`\url{https://mars.mgt.sugiyama-u.ac.jp/PE/}` のように記述すると `https://mars.mgt.sugiyama-u.ac.jp/PE/` のように折れ曲がります。PDF ではリンクの設定がされます。
- 小さい図や縦長の図は場所を取るので、リスト 3 のようにして複数ある場合はまとめるようにしましょう。

リスト 3 卒論サンプルファイル

```
1 \begin{figure}[htbp]
2
3 \begin{minipage}[b]{0.45\textwidth}
4 \centering
5 \includegraphics[scale=0.7]{javascript3.png}
6 \caption{警告画面の例}
7 \label{javaalert}
```

```

8 \end{minipage}
9
10 \begin{minipage}[b]{0.45\textwidth}
11 \centering
12 \includegraphics[scale=0.7]{javascript4.png}
13 \caption{確認画面の例}
14 \label{javaconfirm}
15 \end{minipage}
16
17 \end{figure}

```

リスト 3 の 3 行目と 10 行目にある 0.45 が図の幅になります。合わせて 1 より少し小さな値になるようにします。並ぶ画像の幅が大きく異なる場合は、幅の大きさに合わせた数値にしてください。

- 要約の最初のところは、

```
\begin{abstract}
```

毎朝の洋服選びがストレスに感じるとき・・・

のように 1 行空けること。

- 本文中に<HTML>のようなタグや\$_POST['id'] 等の変数を入れたら文字化けしたりエラーになったら、
`\verb+<HTML>+` や `\verb+$_POST['id']+` のように `verb+~+` の形にするとよい。
- `lstlisting` を使用すると妙な所で切れることがある。

```

\begin{lstlisting}[caption=index.php,label=index]
<?php
for($day = 1; $day <= $day_count; $day++, $youbi++){
    $sql= "SELECT * FROM schedule WHERE userid='".$_SESSION['id']."' AND date= '".$_SESSION['date']."'

```

を処理すると、

```

3 $sql= "SELECT * FROM schedule WHERE userid= '".$_SESSION['id']
    '."' AND date= '".$_SESSION['date']."' order by date";

```

のように変数の途中で改行が入る。そこで次のように「.」のあとに空白を一つ加えると、

```
$sql= "SELECT * FROM schedule WHERE userid='".$_SESSION['id']. ' ' AND date= '".$_SESSION['date']."' o
```

次のようになおることもある。

```

3 $sql= "SELECT * FROM schedule WHERE userid= '".$_SESSION['id'].
    ' ' AND date= '".$_SESSION['date']."' order by date";

```

3. 運用コスト

大抵何かをするには費用 (コスト) ががかかります。商売であれば売上からこの費用を引いたものが利益になります。利益がマイナスになると、所持しているお金が段々減ることになるので、商売を続けることができなくなります。近年様々な分野で持続可能性 (sustainability) が重要と言われてますが、商売に限らず利益がマイナスでは続けることができません。ここでは Web 技術を利用したサービスを実現するために必要な費用について説明します。費用が分かれば、それ以上の売上が期待できるような値段を付ければ、利益を得ることができます。

Web 技術を利用したサービスを実現するためには様々なものがが必要です。インターネット上のデータを収集して、加工して、提供するサービスの場合、データの収集にかかる費用、加工するための費用、提供するための費用がかかります。これらを実行するプログラムを作成する場合、作成費用が一時的にかかるだけでなく、プログラムの改訂も随時必要な事が多く、そのための費用がかかります。完成したプログラムを改訂する気がない場合でも、改訂しなければならない事が生じます。つまり Web 技術を利用したサービスを実現するプログラムは、Web サーバーのプログラムの上で動き、Web サーバーのプログラムは Linux などの OS の上で動きます。残念なことに Web サーバーや OS は日夜様々な攻撃を受けています。セキュリティホールを突かれると、サービスのために集めた個人情報などの大事なデータを盗られる可能性もあります。そのため新たにセキュリティの問題が発見されると、それを解消するためのアップデート情報が提供されるので、それを速やかに適用しなければなりません。そのための費用が必要になりますし、アップデートのために Web サーバーや OS の仕様が変わり、これまで動いていたプログラムが動かなくなることもあります。動かなくなるとサービス停止ですので、急いでプログラムの改訂をしなければなりません。

これまで授業で mars 上でプログラムを作ってきましたが、それができるようにそれなりの費用がかかっていたのです。mars の OS に関してはほぼ毎月一回以上大規模なアップデートがあります。それ以外に mars で動いている様々なプログラムのセキュリティのアップデートが週に数件あります。mars では毎朝アップデートの有無を確認して、あればアップデートするようにしています。Web サーバーや OS の機能が更新されることもあります。より機能が増える事が多いのですが、そのために昔からあった機能が新しい機能に統合されて消えることもあります。そのような機能を使っていた場合は、新しい機能を使うようにプログラムの書き換えが必要になります。

OS を動かすためにはコンピュータのハードウェアが必要です。コンピュータを購入する費用が必要です。コンピュータは電気で動きます。と言うことは電気代もかかるということです。さらにコンピュータをインターネットに接続する費用もかかります。大学や会社は既にインターネットに接続していると思いますので、大規模なデータのやり取りがなければ、既設の設備にタダ乗りできるでしょうが、大々的な動画配信をするとか、個別には少量のデータのやり取りでも相手が何百万というような事になると、インターネットの接続のために別の回線を引く必要が生じて費用がかかります。電気代や回線利用料は年中かかる上に、売上ゼロでも支払わなければなりません。

普通の商売であれば営業時間が決まっていますが、コンビニとかでなければ大抵夜中はやりません。一方 Web 技術を利用したサービスは大抵 24 時間 365 日やっています。何かトラブルがあった時、どのくらいの時間でそれに対応するかによって費用が大きく変わります。正月でも真夜中でも 30 分以内に復旧させる、などと言いつつ莫大な費用がかかります。でもサービスの利用者が早急な対応が必要と言うならば仕方ありません。費用が大きくなると利益が出なくなりますが、無料サービスに慣れた利用者が多いので、高い料金の理解を得るのは難しいでしょう。

このように考えるとなかなか大変ですが、Web 技術を利用したサービスの対象が会社内で、インターネットとはつながらなくて良いような場合は、全て用意しても余り費用がかからないこともあります。外部とつながなければ攻撃も来ませんので、気が向いた時にアップデートで済みます。ハードウェアも中古の PC で性能が足りる場合はただで済むかもしれません。また社内のサービスが夜中に止まっても、それほど怒られたりし

ないでしょう。

3.1 様々なレベルのサービス

かつては大学や会社などは自分で Web サーバーを維持・管理していました。そのためにはハードウェア以上の全ての知識も必要でした。でも考えてみるとトヨタ自動車は、車を作るためにタイヤのゴムから作っているでしょうか。タイヤはタイヤの会社から購入し、ボディを作るための鉄板は製鉄会社から購入しています。できた車を販売するのも別の会社です。同様の分業は Web 技術を利用したサービスでも行われています。例えばブログやツイッターによる情報発信ならば、そのようなサービスを行っているところに申し込むだけで情報発信が可能になります。その際に Web 技術に関する知識は必要とされません。

では卒論で作成するレベルのシステムを実現するためには、どのようなサービスが利用できるでしょうか。

1. **FaaS: Function as a Service** と呼ばれる機能 (Function) の切り売りサービスがあります。Amazon のやっている「AWS Lambda」が有名です。自分の作成したプログラムを登録して、利用者にアクセスしてプログラムを使ってもらう形態です。利用者がアクセスした分に比例した利用料を支払うことになります。利用料は非常に安く「AWS Lambda」の場合、毎月 100 万アクセスまで無料だそうです。

プログラムも提供されるサービスもあります。SaaS (Service as a Service) と呼ばれるもので、現在様々なものがあります。例えば「Windows 365」は Microsoft によるサービスですが、ブラウザを使って mars と同様な形で最新の Windows が使えるものです。自分の使いたいソフトをインストールすることもできるそうです。経理システムなどの会社の業務に必要なものを提供するサービスもあります。

2. **レンタル Web サーバー**: 授業で mars を利用しているのと同じような感じで Web サーバーを利用できるサービスです。mars を使っていた時同様に、ハードウェアや OS の事は知らなくても使えます。探せば無料のものもあります。ただ無料のものは勝手に広告が挿入されたりします。また Web サーバーの能力が低くかったり、使えるディスクの容量や回線の容量が小さかったりします。この辺りは費用が高いものほど改善されます。他に注意する点として、PHP が使えるかやデータベースに何が使えるかの確認が必要です。PHP が使えれば授業で使用した SQLite は使えますが、サービス拡大の際により高機能なデータベースが必要になった際に困るかもしれません。

インターネット上でアクセスするためには、Web サーバーにドメイン名を割り当てる必要があります。mars であれば、mars.mgt.sugiyama-u.ac.jp がドメイン名です。全く新しいものを使用したい場合は、費用が発生します。mars の場合は、既に sugiyama-u.ac.jp を相山女学園が持っていたので、学内の DNS サーバーに設定してもらうだけで費用はかかっていません。会社などでも同様でしょう。レンタル Web サーバーによっては、追加料金無しでドメイン名を提供してくれるところもあります。ただ将来別のレンタル Web サーバーを使う可能性がある場合は、自前のドメイン名を使用した方が良いでしょう。

3. **レンタルサーバー**: ハードウェアと通常 OS までは提供されるので、自分で Web サーバーに仕立てて利用します。Web サーバーとしてよく使われるので、簡単に仕立てられるようになっています。クラウド技術を利用して仮想ハードウェアが提供されるので、サービス混雑時だけ処理能力を増強するような事も可能です。費用もどの程度の能力をどのくらい使用したかで決まりますので、混雑時と閑散期の差が大きい場合は費用の節約ができます。自前のコンピュータでは混雑時に合わせた処理能力にしなければならぬので不利です。またクラウド技術を利用しているので、故障にも再起動などの短時間で対応できます。
4. **ホスティングサービス**: 自前のコンピュータを預かってくれるサービスです。社外のサーバーを利用する際に心配なのは機密データなどが漏洩しないだろうかと言う不安です。社内のサーバーと同様に管理できるのであれば多少は安心です。電気と回線ぐらいは提供されますが、自前のコンピュータですので、様々な管理は全て自前でやる必要があります。遠隔地にバックアップサーバーを用意したいと言うよう

な場合に使えます。

高度なサービスほど高価になります。また高度なサービスの利用に慣れてしまうと、その仕組みなどの理解が進まず、いつまでも同じサービスに依存してしまう危険性が高くなります。「毎月 100 万アクセスまで無料」のような話は Amazon に限りません。いつでも違う会社のサービスに乗り換えられると言うレベルの知識を維持し、コストについては常に意識していないと、やがてカモにされます。

3.2 紹介サイトの構築

「H さんは自分の推したいアーティストの A さんの応援サイトを作りたい」と思ったと仮定して、応援サイト中身ではなくその基盤となるサーバーについて検討してみましょう。ブログで応援する事もできるでしょうが、H さんは A さんに関する様々な情報を見やすい形で提供したいと考え、形が決まっているブログではやりにくいと感じました。H さんの創意工夫を活かすには、自分で好きな構成が可能なサイトが良いと思いました。

せっかく作ったサイトは末永くあって欲しいものです。ただ IT 関係は進歩・発展・競争が激しい分野です。H さんはサーバーのハードウェアや Linux の勉強までする気はないので、レンタル Web サーバーで行こうと思いましたが、サービスを提供する会社がずっと提供してくれるとは限りません。もちろんレンタル Web サーバーと言うサービスがなくなるわけでは無いので、サービスの提供がなくなれば、別の会社のサービスへサイトを引っ越せば良いのですが、その際に URL は変わらないようにしなければなりません。

現在 URL を入力して Web ページに訪れる人はまずいません。ほとんどが Google などの検索システム経由です。そして検索システムがやっているのは、キーワードを URL に変換することです。つまりキーワードと結びついた URL が重要です。H さんの夢？としては、Google に「A さん」を入力すると自分の作ったサイトが一覧の最初に出てくるようにしたい、と言うものがあるのですが、可能でしょうか？

以前梶山が「名古屋 女子大」と入れると梶山が最初に出てくるようにお金を出した、と言う話を聞いたことがあります。Google などではそう言う商売もやっています。最近はスポンサーサイトと言う形で一覧とは少し違う形になっているようですが、Google の検索結果の最初の方に自分の Web ページを出すために SEO (Search Engine Optimization) と言うような技術もあります。ただそのような事をしなくてもユニークで価値のある内容であれば、自然に Google 検索で最初の方に出てくることもあります。「絵入源氏物語」で検索すると、2022 年 5 月の時点では Wikipedia の次の 2 番めに梶山のサイトが出てきます。宣伝も SEO も全くやっていませんが、いつの間にかこうなっていました。

せっかく Google が認めてくれても、URL が変わってしまえば苦労は水の泡ですので、レンタル Web サーバーとは別にドメインを取りましょう。もちろんその取得や維持にお金がかかります。またスマホでメールアドレスの取得の際に経験されているでしょうが、同じ名前のものは取得できません。

4. アフィリエイト

アフィリエイト (affiliate) は web ページに商品の販売などのサイトへのリンクを設けて、それを web ページを訪れた人がクリックしたり、リンク先で実際に商品を購入したりすると、報酬が得られるというものです。多くの報酬を得るためには、まず多数の訪問者が必要ですので、魅力的な web ページを作成する必要があります。また web ページの内容が更新されないと、同じ人は二度と訪れないので、内容の更新も積極的に行う必要があります。ただ自分の好きなことや趣味などの内容の web ページであれば、内容の更新なども負担には感じないでしょう。またリンク先も内容に合ったものであれば、それなりにクリックしてもらえることが期待できます。

以下の報酬例は、「【完全初心者向け】アフィリエイトとは？仕組み・始め方・やり方を図解で解説」(2020 年 12 月 11 日)^{*6}に出ていたものです。

- 転職サイトやエージェントへの登録：1 万円くらい
- FX 口座開設：2～3 万円くらい
- クレジットカード作成：数千円～1 万円くらい
- 婚活マッチングサービス登録：2000 円くらい
- 家電やガジェット購入：購入額の 2%～5% くらい
- 旅行ツアーへの申し込み：1000 円前後

web ページ以外にもメールや LINE などの SNS でもアフィリエイトは可能なようです。

4.1 どのくらい儲かるのか？

アフィリエイトで儲けるとなると、どのくらい儲かるのでしょうか。もし上記の記述を信じるならば、FX 口座開設が毎日 1 件あれば、毎月 60 万～90 万円の収入となりますが、そんなにうまくいくのでしょうか。むしろ FX 口座開設がなかなかされないから、2～3 万円と言う支払いがあると考えた方が良いでしょう。そして普通の商品が価格の 2%～5% くらいの報酬につながることから考えると、2～3 万円支払いは FX 口座を開いた人から 40 万円以上の売上があることが想定できます。そのようなカモになりたくない人は多いでしょうね。

「特定非営利活動法人 アフィリエイトマーケティング協会」が 2021 年 4 月にインターネット上で行った無記名のアンケートによると、4 割弱の回答者がアフィリエイトを始めて一年以内と回答している一方、10 年以上と言う回答も 1 割弱います。この割合は昨年調査結果とほぼ変わらない割合ですので、新たに開始する人も多いが、やめていく人もほぼ同じくらいいることが分かります。そして毎月どのくらいの収入があるかについては、表 4.1 のような回答結果でした。12% ほどの人が毎月 50 万円以上の収入を得ていることになります。ただこのような無記名・任意のアンケートでは、収入を自慢したいような人が多数回答しそうなので、この割合を単純に信じるのは危険でしょう。わざわざ嘘は付かないだろうと考えると、これだけで暮らせるくらいの収入がある人もいる、という事が分かります。

同じアンケートで、アフィリエイトをやっている期間と収入の関係をまとめたものがありました。それによるとアフィリエイトを開始したものの収入がない人の 70% が初めて 1 年以内の人でした。毎月 100 万円以上の収入のある人の半数は、5 年以上やっている人でした。(5 年以上やっている人の半数が毎月 100 万円以上というのでは無いので注意) なかなか簡単には儲からないようです。

^{*6} <https://www.xserver.ne.jp/blog/basic-affiliate/>

表 4.1 アフィリエイトによる月収

	2021 年	2020 年
収入なし	31.16%	31.55%
1,000 円未満	15.79%	16.62%
1,000 円以上～5,000 円未満	8.70%	11.35%
5,000 円以上～1 万円未満	6.67%	6.71%
1 万円以上～3 万円未満	8.55%	7.13%
3 万円以上～5 万円未満	3.13%	4.04%
5 万円以上～10 万円未満	5.58%	4.15%
10 万円以上～20 万円未満	4.01%	4.04%
20 万円以上～50 万円未満	4.59%	4.95%
50 万円以上～100 万円未満	3.65%	2.99%
100 万円以上	8.18%	6.47%

4.2 ASP への登録

ASP と言えば Application Service Provider の方が一般的なような気がしますが、ここの ASP の A は Affiliate です。ASP はアフィリエイトをやって儲けたい人とアフィリエイトをやって売上を増やしたい会社などを結ぶサービスです。ですので登録の際にどちらの立場なのかを選択するようになっています。ASP は儲けたい人に、このような商品のアフィリエイトができますよと紹介したり、その申し込みのサポートをしたり、報酬の取りまとめなどをします。売上を増やしたい会社にとっては、アフィリエイトをしてくれる人が紹介してもらえると利点の他、報酬をまとめて ASP に支払うことにより振込手数料を節約することができます。

前述のアフィリエイトマーケティング協会名のアンケートによると、満足度が高い ASP は、「A8.net」(41.5%)、「afb」(17.2%)となっています。「A8.net」の主な満足理由としては、広告数が多い(56.5%)、稼ぎやすい広告や、自分にとって稼げる広告が多い(41.7%)があり、「afb」の主な満足理由としては、専属担当者の対応が良い(早さ・丁寧さ)(55.6%)、「管理画面のレポートが見やすい」、「情報提供」があげられています。

実際に「A8.net」に登録してみました。登録の際には銀行口座の登録が必要となります。また自分のブログサイトの登録もありました。mars にブログシステムのインストールも検討しましたが、適当な中身も入れなくてはならず、ブログサイトの登録はしていません。登録をすると毎日お知らせメールが来るようになりました。例えば 2022 年 5 月 16 日には 5 通のメールが来ました。1 つのメールに 1 つの会社からの案内がありました。興味のある分野などを設定すると、来るものの内容や数が変わるのかもしれませんが。例えば先物取引の会社から次のような案内が来ました。

通常セルフバック報酬 15,000 円のところ...

口座開設お申込み後 60 日以内に、新規 1Lot 以上の取引完了で

●セルフバック報酬『40,000 円!』

4.3 Amazon や楽天への登録

ASP を通さずに直接アフィリエイトができるようにすれば、多分より多くの報酬が期待できます。ただ一般的な販売会社ではどこにアフィリエイトを申し込む窓口があるのか、アフィリエイトができるのか、分かりません。一方 Amazon や楽天であれば、これらが ASP のように代表して受け付けてくれます。ここでは Amazon

アソシエイト・プログラム (<https://affiliate.amazon.co.jp/>) を一例として説明します。(2022 年 5 月 13 日に確認した)

Amazon アソシエイト・プログラムに参加するには、Web ページ、モバイルアプリ、Facebook、Instagram、Twitter、YouTube、Tik Tok、Twitch.tv のページが必要です。Web ページであれば、オリジナルの内容が多く、内容が 60 日以内に更新されていて、運営主体が自分でなければなりません。SNS であれば「いいね！」などの評価が 500 以上などの条件があります。Amazon アソシエイト・プログラムに参加申請をし、とりあえず商品へのリンクを設定し、180 日以内に 3 件以上の販売実績があれば審査が行われ、条件を満たしていれば引き続きやれることになります。もちろん自分で買うのはだめです。

Amazon アソシエイト・プログラムでは、販売価格から出荷手数料、ギフト包装料、取扱手数料、消費税、サービス料、クレジット (Amazon ポイントを含む)、リベート、クレジットカード処理手数料などを差し引いた金額に表 4.2 の紹介料率を掛けて求めた紹介料がもらえます。ただし紹介料上限は、1 商品 1 個の売り上げにつき 1,000 円のです。

表 4.2 Amazon アソシエイト・プログラム紹介料率

紹介料率	商品カテゴリー
10.00%	Amazon ビデオ (レンタル・購入)、Amazon コイン
8.00%	Kindle 本、デジタルミュージックダウンロード、Android アプリ、食品 & 飲料、お酒、服、ファッション小物、ジュエリー、シューズ、バッグ、SaaS ストアの対象 PC ソフト
5.00%	ドラッグストア・ビューティー用品、コスメ、ペット用品
4.50%	Kindle デバイス、Fire デバイス、Fire TV、Amazon Echo
4.00%	DIY 用品、産業・研究開発用品、ベビー・マタニティ用品、スポーツ & アウトドア用品、ギフト券
3.00%	本、文房具/オフィス用品、おもちゃ、ホビー、キッチン用品/食器、インテリア/家具/寝具、生活雑貨、手芸/画材
2.00%	CD、DVD、ブルーレイ、ゲーム/PC ソフト (含ダウンロード)、カメラ、PC、家電 (含 キッチン家電、生活家電、理美容家電など)、カー用品・バイク用品、腕時計、楽器
0.50%	フィギュア
0.00%	ビデオ、Amazon ネットスーパー
2.00%	上記商品カテゴリーに含まれない商品

Amazon の商品の利率は高くなっています。通常の本より Kindle 本の方が利率が高いのは、Kindle 本をより多く販売したいという意図があるのかもしれません。

4.4 Google AdSense

Google AdSense は Google が提供しているアフィリエイトサービスです。様々な Web ページで見かけたことがあると思います。Google の「Google AdSense – ウェブサイトを収益化」^{*7}のページによると次のような利点があるそうです。

1. 数百万の広告主が広告スペースを求めて競い合うため、入札単価が高く、サイトの内容と合致した広告を、高い確率で掲載することができる。

^{*7} https://www.google.com/intl/ja_jp/adsense/start/

2. 広告ユニットのサイズが表示場所（パソコンまたはモバイル）に合わせて自動的に最適化されるので、広告がユーザーの目に入りやすい。
3. サイトにコードを1つ追加するだけで、サイトのレイアウトに適した広告が自動的に表示される。

広告が Web ページの内容に合わせて自動的に選択されるために、手間がかからないのは確かですが、自分の望まない広告が出る可能性もあります。表示場所も「自動的に最適化」されるので望ましくない場所に広告が表示されるかもしれません。このような場合はカスタマイズ機能や設定で修正できるそうです。

表示された広告がクリックされると収益が発生します。クリックして行った先で購入につながったかどうかは関係ありません。その分収益発生の可能性は高いですが、個々の収益は大きくないことが予想されます。

「Google AdSense – ウェブサイトを収益化」のページには収益額の予想を表示する機能もあります。2022年6月に地域として「アジア太平洋地域の国」とコンテンツカテゴリとして「科学」を選択したところ、月間ページビュー5万回で年間約4,000ドルと表示されました。一ヶ月で5万回ということは年間60万回になります。これで4,000ドルを割るとページビュー1回につき1/150ドルの収入になります。

Google AdSense の利用方法は次のようになります。

1. Google アカウントを取得する。
2. 収益を受け取るための銀行口座、住所、電話番号を登録する。
3. Web ページにコードを追加する。

「Google AdSense（グーグルアドセンス）とは？審査基準やポイントを徹底解説 — MarkeTRUNK」^{*8}によると、広告の表示方式としては、

- テキスト広告: 1～2行の広告タイトルと、Web サイトのアドレスを表示
- ディスプレイ広告: 広告の画像を表示
- 関連コンテンツ: 関連する記事を勧める中に、広告を混ぜて表示

などがあります。またコードを追加した後に、期間が1日～1週間と言われる Google による審査があり、審査に合格すると広告が表示されるようになります。審査の内容としては、

- 独自ドメインを取得しているか
 - プログラムポリシーに準拠しているか
- 次のようなコンテンツがポリシー違反の代表例です。
- 他社の法的権利を侵害したり、違法行為を助長するようなコンテンツ
 - 正規品と偽ったブランド商品の販売等、著作権や知的財産権を侵害するコンテンツ
 - 個人または集団に対する差別及び誹謗中傷
 - 動物への残虐行為
 - アダルトコンテンツ
- ユーザーにとって価値のある内容か
 - 問い合わせ・プライバシーポリシーが示されているか

などがあるようです。

^{*8} <https://www.profuture.co.jp/mk/column/32443>