npca

Nada Personal Computer users' Association

目 次

0 前書き …… 2

1 「録画番組パソコンで！ AVI 版。」計画再浮上！ …… 3

- エンコーダに関するちょっと良い話 …… 5
- フレームレートに関するちょっと良い話 …… 7
- ノイズ除去に関するちょっと良い話 …… 9
- 最後に …… 13

2 パソコンパーツってこんなもの …… 15

- ＣＰＵについて …… 15
- ＨＤＤについて …… 15
- メモリについて …… 16
- ディスプレイについて …… 17
- マザーボードについて …… 18

3 コンピュータ内での画像処理について …… 19

- コンピュータ内のデータの扱い …… 19
- コンピュータ上での画像データ …… 19
- 色の話 …… 20
- プログラム的な話 …… 21
- 終わりに …… 22

4 Herbert と過ごすテスト期間 …… 23

- 新言語 “H” …… 23
- 関数 …… 24
- 変数と再帰 …… 25
- 堀江理論・応用堀江理論 …… 27
- 実際の問題解説 …… 29
- よりハイレベルな問題について …… 34

5 人工無能のつくりかた -文章生成の初歩の初歩- …… 35

- 人工無能を知らないあなたへ …… 35
- ごくごく初歩の文章生成 …… 36
- マルコフ連鎖で手軽な作文 …… 37
- 手軽な改良計画 …… 38
- 今後の指針のようなもの …… 39
- おわりに …… 39

6 油断大敵ウイルスまみれ？ …… 41

- 基本知識 …… 41
- 感染までの攻防 …… 41
- まとめ〜 …… 45

7 後書き …… 46

・・・・・0・・・・・

前書き

59 回生 一同

この部誌を手に取ってくださった皆さん、こんにちは。あなたの手元にこの部誌があるということはなんとか公開にこぎつけられたんでしょう。我が npca では日々プログラム技術の向上、パソコンに関する知識の会得を目的に活動しています。その結果を文化祭作品とこの部誌とで年に一度披露している、というわけです。部員がその知恵を生かして書いたのがこの部誌。文章から疲れてるオーラが出てますが気にせず読んでください。

現代は高度情報化社会の時代です。パソコンの知識のカケラもなければ生きていけません。ですがパソコンにあまり多くは触れない人にとって、パソコンはあまりにも不思議でわからない存在です。パソコンはインターネットで 2ch が見られるかと思えば、黒画面に白文字を並べればプログラムが動いたり、最近ではタダで違法に大量のファイルが手に入ったり、自分の情報が流出したり。パソコン一台でいろんなことができたりされたりしてしまいます。でも恐れることはありません。友達はボールだけではありません。ましてや愛と勇気だけが友達なんて事はもっとありません。パソコンも友達です。この部誌がその私達の友達であるパソコンについて少しでも多く知る機会になれば幸いです。まぁ載っている内容は部員の自己満足的要素が大きくて、あまり初心者向けではないかもしれませんが。

そういえば我が部も時代の流れに乗ってウェブページが存在したことを思い出したのでアドレスを載せておきます。

`http://npca.my-sv.net/`

暇があったら訪問してくださいね。後、もっと暇だったら部室にも来てください。熱烈大歓迎です。旧校舎四階、地学教室横にカオスはいつでも存在します。場所がわかりにくいともっぱらの評判ですがだれもが一度は通る道です。部員が淋しがるので思い切ってドアをノックしてください。

長くなりましたが、何が言いたいかというと、楽しんでこの部誌を読んでください、ということです。とにかく楽しむのが npca。この部誌に最後まで楽しんでお付き合いいただければ幸いです！

ちなみに文化祭当日、部員の眼が死んでいたらやさしく微笑みかけてください。多分前日徹夜でゲーム仕上げてます。

・・・・1・・・・

「録画番組パソコンで！ AVI 版。」計画再浮上！

59 回生 ノーベル

ついにやってきましたこの季節！春！若葉！僕の誕生日！そして文化祭！

こんにちは、灘のヨン様、npca の貴公子との呼び名が高いノーベルです[1]。前回、大センセーショナルを巻き起こした記事、「録画番組パソコンで！ AVI 版。」が一年の沈黙を経て再浮上[2]！去年は AVI に関する基礎知識、録画番組をパソコンへ移す基本的な作業についての記事を書きましたが、今年はあの記事を読んで録画番組をパソコンで見るようになったあなたへ捧げる記事なわけです。僕自身の AVI 変換技術も大分上がりました。上がったはずです。上がってなかったら泣きます。というわけで今回は AVI への変換方法や使うソフトはあえて省きます。去年の記事なんか見てないよ〜という人や去年の部誌なくしちゃったよ〜というあなたは npca のホームページに去年の部誌が置いてあるので是非ダウンロードしてくださいね[3]。

気がつけば去年からもう一年。……当たり前ですね。違うかったらすごいですよね。何を言ってるんだ。その当たり前の一年の間に AVI の変換を取り巻く環境も随分と変わりました。後述の h264 の台頭、DivX と Xvid のバージョンアップ。さらには AVI にとってかわるコンテナ、MKV の出現[4]。まぁ去年あれだけ僕が待ち望んでいたデジタル放送はあまり変化しませんでしたが。

とりあえずどうでもいい話はここら辺までにして、本題に入ります。今回の変換の基本方針は「タダ」「軽い」「綺麗」「速い」「人気」の 5 つです。全部読んで字の如く、な気がしますが一応説明していきます。

「タダ」は変換ソフト、エンコーダ、再生ソフトにお金をかけないこと。とりあえず僕はいたって健全なる一般高校生なのでお金がありません。今この記事を書いてるパソコンだって親がお金を出して買ってくれた家族共用パソコンです。家でテレビ番組を録画してパソコンに保存するのは自分の趣味で、できるだけ親のスネはかじりたくありません。ってなわけで悲しい現実に立ち向かわざるを得ない高校生にはタダほど魅力的なものはないわけです。まぁこのエンコードまわりのソフトは有料のソフトと無料のソフトで性能があまり変わらなかったり、ソフト自体がタダのものしかなかったりする、っていうのも一つの理由ですが。

「軽い」には二つ意味があって、変換が軽いこと、再生が軽いことです。変換が軽い、とはいってもそこは AVI 変換、重いのは当たり前です。ただ、それでも変換作業がパソコンに負担をかけるのはできるだけ避けた方がいいです。夏場の変換作業で CPU が熱暴走〜なんて話もたまに聞きますしね。後、変換時の軽さと違い、再生時の軽さの違

---

[1] 苦情は受け付けません。

[2] ちなみに前回、あの記事を載せた後二人ほどコンタクトとってくれました。どっちも NPCA 部員だったなんて口が裂けても言えませんが。

[3] 2006/4/20 現在未確認

[4] 実は h264 も MKV も去年からあったんですが、人気がなかったので省いてました。最近ようやく有名になってきたので取り上げてみようかな、と。

いは結構でます。録画して見ることの良さの一つは自分の好きな速さで映像を見られること。たった 5 秒早回しするのに 10 秒もかかるようなエンコーダは嫌です。絶対嫌。最近のパソコンは高性能のものも増えてきたので結構無視されがちなところですが、まだまだ影響が大きいので軽さを無視するわけにはいきません。

「速い」は「軽い」の項目と重なることが多いですね。簡単に言えば変換が速いこと。少しくらい重くても速ければ精神的に楽です。AVI 変換中は基本的に作業はできないので、夜中に作業をさせます。ですからいくら重くても速ければ速いほど多くの映像を変換できます。ちなみに、AVI 変換はノイズ除去の過程で速さが著しく落ちることもあります。詳しくは後述しますが、ノイズ除去に使うフィルタの効果と速さも考慮する必要があります。

「綺麗」は変換後の映像が綺麗なこと。変換後の映像にノイズをできる限り乗せないことです。ノイズが乗る主な原因は 2 つ。番組の録画時に放送の電波自体に既にノイズが乗っている場合と、エンコーダで変換した際にビットレートが足りなかったためにノイズが載る場合。ノイズ除去の方法は詳しくは後述しますが、1 つ目の放送時にすでに乗っているノイズは返還前にフィルタを使って無理矢理取ります。まぁこの過程はデジタル放送を録画してる人や市販やレンタルの DVD の映像を取り込んだ場合は関係ありませんが。2 つ目のエンコーダが原因のノイズはエンコーダの設定変更、エンコーダ自体の変更で対処します。ちなみにこれらの対処をいくらしても元の映像と全く同じ綺麗さにはなりません。ただ、人間の目で判別できないぐらいのノイズに抑えることはできます。MP3 のビットレートをどれだけ上げてもに納得できないような、完全にプラシーボ効果に[5]はまっちゃってるような人はノイズが気になるかも。とりあえず僕には気になりませんが。

「人気」は、使うエンコーダが人気があるかどうか。「別に人気がなくったって自分で見るだけだからいい！」という人もいるかもしれませんが、人気がなくて高性能〜と一部マニアで有名になるような無名エンコーダは総じてどこかに欠点があるものです。上記の 4 つの理由以外にも実は動作が不安定だったりとか、開発が途中でいきなり打ち切られて対応するデコーダもプレーヤーもなくなってしまったりだとか。マニアぶって誰も知らないようなエンコーダを使うことも、変換技術が上がってからはいいかもしれませんが、それまでは絶対にダメです。大体みんなが使っている、ということはそれだけ性能が良い、と言うことです。どこどこのサイトだけでしか取り上げられてなかったけどそこではベタボメ〜だとかそういう情報に惑わされないようにしてください。

とまぁ、今回はこんな感じで進めていきたいと思います。ちなみに変換ソフトは「AviUtl ver0.98d」を基準に考えていきます[6]。簡単に動画を変換できますしなによりフィルタが多いのでノイズ除去とか便利ですし。後、途中でわからない単語があったら途中におまけである程度載せてあるので見てください。それ見てもわからなかったら Go ！ Go ！ Google ！

第一部を書き終えた時点で僕が既に疲れてる＆眠いので先行きが異常に暗いですがこれを読んでくれるあなた、そう、あなたです。あなたのためにがんばります[7]！次の章からはあなたを AVI 変換の初心者から中級者、上級者へと押し上げる（予定）のノーベルのちょっと良い話シリーズ。見れば変換テクが上がること間違いなし！多分！！で

[5] 「プラシーボ効果」を知らない人は「偽薬」でネット検索すると良い事あるかも。

[6] 最新版の 0.99 は YUY2 への変換に関してバグが見つかっているため。

[7] リリカルマジカルがんばります！

Coffee Break **「MKV」の香り**

実は第一章で MKV について触れておきながら、結局 MKV について触れてません。それは今回の解説ではコンテナまわりの話は省略されてしまった、というのもあるんですがフリーで思い通りにうまく MKV 出力してくれる使いやすいフリーのソフトがないんですね。あっても AVI コンテナを通して MKV コンテナに入れなおす、とか。h264 と同じで、これからの技術としては注目されますが、これもとりあえず時期尚早、としか言いようがないですね。待て。而して希望せよ。

は、あなたの後ろに立っているヘトヘトの僕の生霊を感じつつ、次の章へと向かってください！！

エンコーダに関するちょっと良い話

では早速、エンコーダについて書いていきますね。上の 5 つの条件を多く満たすエンコーダは「DivX」「Xvid」「WMV」「x264」の 4 つ[8]。「DivX」と「Xvid」については前回軽く説明しましたが、今回はこの二つも含めてより詳しく比較解説していきたいと思います[9]。後、各コーデックの説明中にノイズの種類とかが書いてあるんですがわからなければ第 4 章に載ってますのでそっちを参照してください。

名称	タダ	軽い	綺麗	速い	人気
DivX	△	◎	△	○	◎
Xvid	○	◎	○	○	○
WMV	○	△	○	△	○
x264	○	×	◎	×	△

DivX

言わずとも知れた超有名エンコーダにして一番の老舗。DVD の映画を CD 一枚に収める、というコンセプトは世界中で広く受け入れられ、現在のエンコーダの流れを作ったといっても過言ではない。らしい。ただ初めはマイクロソフトの開発した MS-MPEG4 のパクりだったとか。最新バージョンは 2006/4/20 現在で ver6.2。ただ ver5.2.1 あたりが安定動作版として人気。ちなみに無料版でも変換自体はできますが、有料版も存在します。有料版の方が圧縮率が高く、設定の自由度も高いらしいです。試したことないので実際はわかりませんが。その無料版を変換してみた感じでは世間一般で言われているほどは悪くない感じ。ただ、色の再現度やノイズは他の 3 つに比べればやはり少し劣る気が。後、実写向きとされてますが特に他の 3 つとの相違点は見つかりません。DivX が元々実写映画のエンコードに使われるために生じた噂話かも。

Xvid

[8] x264 は h264 の技術を使ったエンコーダ。h264 と実質的には変わりません。h264 の技術を使ったエンコーダは有料のものが多いんですがこれはタダ。

[9] 世の中にはエンコードの解説の載ったページがゴマンとありますがそれらに流されないためにもその解説を極力無視し、僕が自分自身で映像を変換してその結果に基づき、僕の主観に沿って評価を出しています。

DivX がまだ無料だった頃、開発元で DivX を商用製品路線へ乗せるかどうかで意見が分かれ、それに反発した一部の開発者がオープンソース、フリーをコンセプトにして開発したのが Xvid。MPEG-4 特許に抵触[10)]しているため実は問題があったりするんですが、個人的な使用には問題ありません。最新バージョンは 2006/4/20 現在で ver1.10。デュアルコア対応の ver1.20 もβ版として存在します。僕も現在使っているのがこのコーデック。モスキートノイズが乗りやすい傾向にありますが色の再現度は No.1。手元の環境では画質に定評のある h264 よりも色の再現に関しては上回っています。また、モスキートノイズが乗る、といっても映像は常に動き続け、それにあわせてモスキートノイズも動き続けるのでスクリーンショットを取らない限り、日本中の腐女子を虜にするテニヌプレーヤーぐらい動体視力が良くないと気になりません。なので、綺麗なスクリーンショットを取りたい人はあまりオススメできないかもしれません。また、これは Xvid だけに言えることではないですが、やはり暗部でのブロックノイズが目立ちます。後、様々なサイトで Quantization type などのチェック項目の説明があり、それらにチェックを入れると変換に時間がかかる代わりに画質が良くなる～的な印象を与えられますが Use VHQ for bframes too 以外にチェックを入れると低速でパンするシーンで色にじみが発生したりします。おかげでどれだけ苦労したことか・・・。1 ヶ月以上研究でつぶれましたね。その研究の結果、ほとんど全てにチェックを入れないのが最適、とわかった時の悲しみといったら・・・。Encoding type は Single pass での Quality エンコードがオススメ。番組にもよりますが、CM を抜いた 25 分の番組が quantizer2～3 で 300MB 前後になる感じ。再生時の軽さや変換の速さでは定評のある DivX とほとんど同じ。個人的には今一番オススメできるエンコーダです。

WMV

マイクロソフトが開発したエンコーダで、デコーダの観点から言えば世界で一番普及しています。というのも、Windows に標準でついてくる WindowsMediaPlayer に標準でデコーダが搭載されてるんですね。自分の作った映像を友達に渡して見てもらいたい、なおかつ相手がパソコンに詳しくないのであれば一番オススメなのはこの WMV になります。特徴として、映画サイズの映像だけでなくネット配信用の低ビットレートの映像から HDTV の大きなサイズの映像にも対応していることがあげられます。最新バージョンは 2006/4/20 現在で 9。アニメの変換で人気があります。が、それほど良いエンコーダかどうかは疑問。WMV は色の再現度はともかく、画面がヌメっとして甘ったるくなることが多々あります。内部の詳しい動作についてはわかりませんが、どうやらモスキートノイズやブロックノイズを低ビットレートでも出さないようにする代わりに細部を潰しているみたいですね。それが見る側になんだかヌメっとした映像、という印象を与えるみたいです。ただ、モスキートノイズやブロックノイズが出るくらいならそれでいい！と言う人は WMV の方が良いのかもしれません。後、変換が遅くて重い。とにかく遅くて重い。今年、一度に大量の映像を WMV でエンコードする機会があったんですが遅くて重くてイライラする。DivX、Xvid の 2 倍かそれ以上時間がかかる上に CPU は常に 90%以上を単体で確保。とりあえず WMV を使おうと思ったらハイスペックのパソコンが必要かも。

x264

---

10) 2 つの権利が同一の対象に対して重なって成立しており、どちらを実施してもお互いの権利内容を実施する状態のこと。

最近頭角を現してきた、h264 の技術を利用したフリーのコーデック。h264 は MPEG-2 の二倍以上の圧縮効率を実現するといわれていて、ワンセグ放送も実は h264。離散コサイン変換 (DCT) やフレーム間予測、量子化、エントロピー符号化、算術符号化など僕みたいに少しかじっただけでは全く意味がわからない程、すごいアルゴリズムを使っているらしいです。その超最先端技術をフリーで試せるのが x264、というわけ。ただ、先ほど説明した大量のなんだかすごそうなアルゴリズムを全て効率よく使い分けることで初めて MPEG-2 の二倍以上の圧縮効率になったり、高画質になったりします。そこはフリーのコーデック。過剰な期待は禁物、と言うことを覚えておいた方がいいかも。最新バージョンは 2006/4/20 現在で Rev.503。ただバージョンアップが異常に早く、二日に一度にバージョンアップしたりするくらいせわしない。画質は Xvid が ver1.03 だった頃はこの x264 がダントツだったんですけど、Xvid の欄にも書きましたが Xvid のバージョンアップで色の再現度では抜かれた感じ。それでも細部の再現度、特にモスキートノイズが少なさでは飛びぬけていて、スクリーンショットを取るには最適。また、他のエンコーダが 16 × 16 画素ブロック単位で変換しているのに対し、h264 は 8 × 8 画素ブロック単位で変換しているのでブロックノイズも目立ちにくい。が、ブロックノイズも完全には消えるわけではないので気になるときには気になります。加えて再生がとても重い。5 秒早送りするのに 3〜4 秒もかかったらそりゃイライラしますよ。メモリが 1GB 以上ならそれもだいぶ解消されるらしいですが。ここまで紹介したコーデックの中で将来生き残るのは確実にこの x264 ですが、まだ使うには時期尚早、というのが x264 に関する大まかな感想。

というわけで各コーデックの特徴の説明終了。まとめると今、実用性があるのは Xvid、将来性があるのは x264、といった感じ。まぁここに載せた画質に関する評価は僕の主観にかなり依存してるので「ノーベル？誰だよそれ？そんな奴信用できるかっ！」って人は一度全てのエンコーダで自分自身で変換して自分の目で確かめて比べることが大切です。人によって好みは違うので選ぶエンコーダも変わってくるはず。それに色々なエンコーダを使えばそれぞれのエンコーダの良さがわかるだけでなくエンコーダそれ自身についても設定を通じてわかるようになってきます。まさに習うより慣れろ、なのでがんばって試してみてくださいね。

フレームレートに関するちょっと良い話

次はフレームレートに関するお話。実際に映像を変換して、変換後の映像が変換前よりカクカクしていて「あれっ？」と思ったことはありませんか？それは変換前と変換後でフレームレートが違う時、特に 30fps の映像を 24fps で変換してしまった際に発生する現象です。これを理解するために、まずはフレームレートについて解説していきます。

フレームレートとは単位時間当たりの画面の更新回数をさす言葉で、単位は普通「fps(Frames Per Second)」、つまり 1 秒間に何度映像を更新するかで表します。日本の番組は主に 30fps、アニメや映画は 24fps であることが多いです[11)]。24fps の番組はたいてい 4 の倍数のフレームと 5 の倍数のフレームが同じ映像になって放送されてます。「多い」と書いたのは、アニメや映画であっても全て 24fps ではない、と言うこと。最

[11)] 正確にはそれぞれ 29.97fps、23.976fps。昔ながらの方法、というかほとんど伝統ですね。この微妙な違いはきちんと変換ソフトが対処してくれるので普通は考えなくてもいいです。そういえばこの微妙な値って白黒テレビの頃からしいです。古い・・・。

Coffee Break

「AVI 変換時のパソコンのファン音」の香り

AVI 変換時に結構問題になる（っていうかうちで問題になってる）のが AVI 変換時のパソコンのファンの音。いくら夜中の涼しい時間に変換をするとはいえ、CPU を使用率 50〜100%で数時間動かし続けるので否が応でも CPU や HDD の温度は跳ね上がり、それを冷やそうとファンがうるさくなります。これが結構馬鹿にならないもので、うちは親の寝る部屋とパソコンの置いてある部屋が繋がっているので結構親はファンの音が気になってるみたいです。これを解消するには CPU の温度を下げるしかない。で、パソコンの給気口の周りに冷却材を置いてみたり、こまめにパソコンの排気口についた埃を取ったりしてみたんですが、効果は全く見られませんでした。となると、大本の CPU の使用率を減らすしかファン音を小さくする方法はない。でも変換ソフトからは CPU 使用率はいじれないし、どうしたものか・・・。と思って CPU 使用率をいじれるソフトを探して見つけたのが「BattleEncoderSirase」。これを使えば思いのままに CPU 使用率をいじれます。うちの家では-66 %、つまり 17 %まで CPU 使用率を下げないとファン音は下がりませんでしたが。ちなみに下げれば下げるほど変換にかかる時間は長くなることに注意。当たり前ですが。それでもファンの騒音に悩まされてる人にはオススメ。アドレスは最後に載せておいたので興味のある人は参照してみてください。

近は特にその傾向が強く、前編 30fps だったり、OP と ED だけ 30fps、本編は 24fps、という変則的なものもあります。この場合、24fps で変換してしまうと全体または一部が 30fps だったシーンが無理矢理 24fps に間引かれてしまい、画面がパンするシーンなどでカクカクしてしまいます。これが変換前より変換後の方がカクカクなってしまう主な原因です。とはいえ、AVI コンテナでは途中で fps を変えることは出来ません[12)]。

では、どうすればいいのか。勘のいい人は気づいたでしょう。fps の最小公倍数を取ればいいんです。つまり 120fps で変換、使わないフレームに NULL フレームを入れれば[13)]、24fps だけ、30fps だけで変換した時とサイズをそれほど変えないで映像を保存することが出来ます。実際、120fps にして増える大きさは 25 分の番組でせいぜい数 MB 程度。我慢できない大きさではありません。ただ 1 秒間にそれだけの数のフレームを読み込むので、Windows98 かそれより前のすこし古いパソコンか、XP でもスペックが異常に低いパソコンでは再生がうまくいかないかもしれません。まぁそんなパソコンで映像を変換、視聴しようとする人も少ないと思いますが。

さらにここで疑問。フレームレートは 24fps と 30fps で最小公倍数をとればいいことはわかりました。しかし、先ほど述べた通り、24fps は 4 の倍数と 5 の倍数のフレームを同じにして 30fps として放送されています。純粋な 24fps の番組ならその同じフレームをすべて統合すればいいですが、同じ方法だと 30fps の統合してはいけない部分まで統合されてしまいます。そこで 30fps と 24fps を判別することが必要になってきます。ここからものすごい暇人でないと出来ない方法、時間がない人がする方法の 2 つを説明していきます。

まず 1 つ目は、全てのフレームのフレームレート自分の目で判別する方法。とはい

12) これは AVI の規格自体が古いことが原因で、他のコンテナ (WMV、MKV) では VFR、つまり可変ビットレートの映像を入れることが出来ます。

13) 「NULLNULL だよ、ララちゃん。」「ルルちゃんだって・・・。」「あはははは！！！」ではなく何の情報も入っていない空のフレームの事。

え、数万フレーム全ての確認なんて、さすがに暇人でも目が疲れてしまうのである程度のコツと経験、知識が必要になってきます。例えばアニメでは本編だけ 24fps だったり、本編の中でも画面がパンするシーンや人物や画面の動きが激しいシーンだけ 30fps だったりと、ある程度の法則は見つけ出すことができます[14)]。最近のアニメはどんどん不規則になってきているのでどこまで通用するかはわかりませんが。そうやって 24fps と 30fps を見分けたら、次にその 2 種類をそれぞれ別々にエンコード。この際、30fps の方も 24fps の方もこの時点で 120fps で変換するところがポイント。ここで見かけ 120fps、NULL フレームを抜いて数えれば 24fps と 30fps の 2 つの映像が完成すればもう後は繋げるだけ。ここで 2 つの映像がそれぞれ 120fps でないと前述の AVI コンテナの限界から繋げられません。とにかく、この方法で作成すれば一番確実に 24fps、30fps 混合の映像を変換することが出来ます。

24fps

1					2					3					4					24fps(元)
1	*	*	*	*	2	*	*	*	*	3	*	*	*	*	4	*	*	*	*	120fps(擬似VFR)
1a		1b			2a		2b		2a'		3b		3a		4b		4a		4b'	60fps(放送)
1					2				3				4				5			30fps(キャプチャ)

30fps

1				2			3				4				5					30fps(元)
1	*	*	*	*	2	*	*	*	*	3	*	*	*	*	4	*	*	*	*	120fps(擬似VFR)
1a		1b		2a		2b		3a		3b		4a		4b		5a		5b		60fps(放送)
1				2			3				4				5					30fps(キャプチャ)

2 つ目は、変換ソフトのプラグインに判別から変換まで全て任せてしまう方法。これなら一々目を真っ赤にしてフレームレートを確認する必要はありません。ただ、そこは自動、一部間違ったフレームレートで認識してしまう可能性もあります。自分の手間と精度を天秤にかけて、自分にあった方法を選んでください。AviUtl では「AviUtl プラグイン置き場」さんの「自動フィールドシフト インタレース解除プラグイン」がオススメ。結構高精度で判別してくれます。インターレース解除もやってくれますし。使い方は省きますので、HP を参考にしてください。アドレスは最後に載せてあります。

とまぁ、こんな感じです。どれもこれも AVI コンテナの古い制約から発生する弊害のせいなわけですが。でも AVI 以外に簡単に直接出力できるコンテナも少ないですしね。ここに書いてある事が全て理解できるようになった頃には映像自体に関する知識も深まっているはず。何事もあきらめずがんばっていきましょう！

[14)] 正確には連続する 5 フレームのうち、2 コマにインタレ縞があれば 24fps、なければ 30fps というのが判別方法の基準になるみたいです。さらに厳格に見れば例外もあるそうですがそんなのは滅多にお目にかかれないので気にする必要はないかも。インタレ縞の説明はするのめんどくさいので適当にネットで検索してください。（オイ）

Coffee Break **「最近のアニメに関するヨタ話」の香り**

最近は異常に所謂萌え系アニメが増えてきましたね。別にそれはそれでいいんですけど如何せんストーリーがダメなものが多くて、なんとかならんもんかと日々嘆いてます。と、そんな話は部誌には似合わないのでもう少し専門的な話。アニメ製作現場のブームになってるのかもしれませんが、24fps **と** 30fps **の混合アニメが異常に増えてる気がします。それこそ手動での確認はできないくらい。さらには人物** 24fps**、背景** 30fps **という「変換させねーぞ！」とでも言いたいかの様な物まで。後、**16:9 **のアニメも増えてます。エンコーダの特性か知りませんが心なし** 4:3 **のアニメの方が綺麗にエンコードできる気がするので変換しにくいったらありゃしない。ハイビジョン放送を目指した物か、はたまた製作側が楽するためのものか知りませんが、アニメの大統一規格、みたいなものが登場すればこちらとしては楽なんですけどね。**120fps **とかも必要なくなるし、一々画面サイズ変更しなくて済むし。まぁ夢見たいな話ですが。**

ノイズ除去に関するちょっと良い話

最後は、ノイズ除去に関するお話。ノイズとは三省堂提供「大辞林 第二版」によると「情報理論などで、信号の性質・内容に影響を与えるおそれのあるデータの乱れ。」だそうです。そのまんまですね。この章ではそのノイズを変換時にフィルタで取る方法を載せていきます。とはいえノイズにも色々と種類があって、その種類ごとに対処法も変わってきます。それぞれのノイズに対応した AviUtl のフィルタの置いてあるアドレスも載せておくので困ったら参照してみてください。

1:エンコーダによって生じるノイズ

- ブロックノイズ
- モスキートノイズ
- 残像

エンコーダで動画を変換すると、映像のサイズは 1/10 近く小さくなります。ってことはその圧縮の過程で確実にいくつかの情報が抜け落ちています。そうして発生するのが「ブロックノイズ」「モスキートノイズ」です。まず、この 2 つのノイズの説明です。

「ブロックノイズ」はモザイク上の小さな四角が発生するノイズ。画面で暗い部分で、色が平坦に広がっている場所に発生することが多いです。これはエンコーダは普通小さなブロックに区分して、ブロック単位で変換しているために生じています。そうやってブロックで変換する際、そのブロックを変換するときに一番適した符号が圧縮に使われます。この時、隣のブロックと色がほとんど同じ場合、微妙な違いで符号が変わり、結果としてほとんど同じ色だったはずの隣のブロックと差が生じます。また、ブロックごとに変換する際、高い周波数成分を減らすんですが、それが暗い部分で起こりやすい原因だったりします。h264 ではこのブロックのサイズを面積にして 1/4 にしているので、x264 はブロックノイズが発生しにくいです。ただブロックのサイズを小さくするということは変換時の計算量を増やすことにも繋がるので x264 は他のエンコーダよりも重くて時間がかかる原因になってます。WMV がなんであんなに重いかは不明。本当になんでだろ・・・。

「モスキートノイズ」は蚊の大群がまとわりついたように見えることから名づけられたノイズ。アニメなどの塗りつぶしたような単色の部分がある場面で、それと隣接する色調の大きく違う色がある場合、例えば人の輪郭周りなどで多く発生します。発生理由はブロックノイズと似ていて、変換時に高い周波数成分が失われるためで、こちらはブロック内部で起こるために蚊の大群がまとわりついたように見えてしまいます。

どちらも映像を変換する際にビットレートが低すぎる場合に起こりやすいです。ちなみに x264 で変換された映像はこのどちらもが低く抑えられるようになっています。h264 系のエンコーダって、重くなければ本当に良いエンコーダなんだけどなぁ・・・。

2:録画時に入ってしまったノイズ

(1) 通常ノイズ

- 2D ノイズ
- 3D ノイズ
- 縦線ゴースト

どれだけすごい DVD レコーダーを使っても、地上波放送ならテレビの放送局から自分の家まで電波が来る過程で必ず電波自体が劣化します。それがこれらのノイズの元となっています。「うちの家はデジタル放送だからそんな心配ないよ〜」とのたまう、ぶるじょわじーなあなたは無視してください。

「2D ノイズ・3D ノイズ」は映像全体に入るザラザラしたノイズの事。最初に 2D ノイズ、3D ノイズと別々に扱いましたけど、実質同じです。この 2 つの違いはフィルタを使ってノイズを取る際の方法に関わってきます。「2D」「3D」とはそれぞれ「2-Dimension」と「3-Dimension」、つまり二次元と三次元の事なんですがもちろん映像に奥行きはないので三次元の方は通常の意味での三次元ではありません。三つ目の座標軸を Z 軸ではなく時間方向の軸、つまり時間軸として考えた場合の呼び名です。つまり 2D のノイズ除去フィルタとはある画素とその周りの画素を調べて、その差が著しかった場合に違いを少なくしてノイズを除去する方法で、3D のノイズ除去フィルタはある画素と同じ場所の前のフレーム、次のフレームの画素を調べて時間軸的な差が少なくなるようにしてノイズを除去する方法です。AviUtl には標準的にどちらもついていて、3D ノイズ除去フィルタはそれでいいんですが 2D ノイズ除去フィルタはそれだけでなく、「GNB の館」さんの「Wavelet_NR_Type-G」を併用することを薦めます。ウェーブレットという技術を使ったフィルタで、輪郭を残したままのノイズ除去が可能です。数値の設定項目が滅茶苦茶多いんですけど、その補助をしてくれる「TypeG_Helper」もあります。このフィルタはとても強力な上に正確ですが、エッジ部分のノイズは取りきれません。そこで AviUtl に標準でついてくるフィルタでこれを取ります。ちなみにこのウェーブレットの技術を使った 3D ノイズ除去フィルタも存在していて、精度も標準についてくるフィルタより高いんですが異常に重く、変換にかかる時間が 2 倍以上になってしまうのでオススメしません。スペックの高いパソコンなら試す価値はあると思いますが。ちなみに 3D のノイズ除去フィルタは 2D のノイズ除去フィルタより前に持ってきたほうがいい感じ。違いはほとんど無いですけど、なんとなくそんな気がするので。

「縦線ゴースト」とは映像全体に縦にかかる輝度の違う線の事で、これは AviUtl に標準で入っているフィルタでほとんど消すことが出来ます。縦線ゴーストの入る

Coffee Break　「専門用語が分からない人の避難所」の香り

詳しく説明しだすとキリがないのでそれぞれ一言で。

エンコード:映像を変換して圧縮する作業
エンコーダ:エンコードしてくれるソフト
デコーダー:エンコードされた映像を見せてくれるソフト
フレームレート:1 秒間に表示される映像のコマ数
ビットレート:1 秒間に送受信できるデータ量
インターレース:テレビの電波とかが 1 回画像を送るのに奇数行と偶数行で 2 回に分けること
コンテナ:映像の入る箱、これに映像と音声を入れて同時に視聴できるようにする
AVI:**コンテナの一種、現在主流**
MKV:**コンテナの一種、前途有望**
スクリーンショット:略称スクショ、画面保存
オープンソース:ソースがオープンなこと、ソースコード公開が主な基準
パン:パンがなければケーキを食べればいいじゃない、ではなく画面がスクロールするシーンの事
フリーウェア:タダより安い物はないソフト
CPU:**いんてるはいってる？ではなくコンピューターの中央処理装置**
HDD:**えいちでぃーでぃー、ではなくハードディスクドライブ**
僕の誕生日:4/26
ノーベル:この記事の作者、体育会系
NPCA:Nobel Puchi Central Adventure wolrd

原因を調べてみたんですが、情報が少なく見つかりませんでした・・・。後、標準で入ってるフィルタの使い方がわかりにくいので、っていうか僕はわからなかったので書いとくと、どこか真っ白、もしくは全体が灰色の場所で「ゴーストの検索」ボタンを押してください。それだけで映像全てに縦線ゴースト除去がかかります。

通常ノイズ、というか通常のノイズ除去フィルタで取るノイズは大抵これだけです。後、つい忘れがちなことですが、フィルタは「もうちょっと取った方がいいかな？」ぐらいノイズをのこしておいて変換しておいたほうが綺麗に変換できます。あまりにノイズをとりすぎると色と色の境が無くなったり、残像が出たりします。値を調節するときに見るプレビュー画面は止まっている映像なので動かして見たときと随分と印象が違い、ノイズがとても多く見えます。なのでノイズはかなり残ってるなぁ、と思う程度で充分です。注意してください。

(2) 特殊ノイズ

- 画面端のギザギザノイズ
- ゴースト
- 透過性ロゴ

これらはノイズ除去フィルタで取る物ではなかったり、特殊な場合に入るノイズのようでノイズでないものの取るものです。よくよく考えたら「ノイズ」の話に書くのもおかしいんですが、ついでなんで一緒に書いちゃいます。

「画面端のギザギザノイズ」はどこで検索しても引っかかりません。当たり前です。僕の造語です（オイ）。というのもこれは滅多に入らないノイズというか、僕以外に見たこと無いので・・・。画面端がのこぎりの歯みたいに 1 ドットずつ交互に違う色が入ってしまう現象で、これを前述のフィルタで取ろうとするとかなり強

力に除去をかけなければならなく、他の普通の映像にも影響を及ぼしてしまいます。で、どうすればいいかと言うと・・・。無理です。とれません。画面端のノイズであることを利用してクリッピングで削っちゃいます。いや、無理な物は無理なんですって。しょうがないんですって。人間、限界はありますって。

「ゴースト」はどちらかというと (1) に入ってもいいようなノイズで、真っ白な画面に黒い部分、つまり真っ白い紙に真っ黒い文字が書かれているような場面でその周りにゴーストのように同じ文字がずれて薄く表示されてしまうことです。一応 AviUtl にも標準でゴースト除去フィルタはあるんですが、全くと言っていいほど取れません。僕自身有効な対策を未だ見つけられていません。これもあきらめましょう。

「透過性ロゴ」は BS 放送などで入る右上の文字の事で、これはある程度取ることができます。というのも原理自体は簡単で、これらのロゴはロゴ自体の色と本来の色を重ね合わせて表示させているため、ロゴ自体の色を引けば元の色が現れます。MakKi さんの「透過性ロゴ フィルタ」が有効で、重宝しています。僕は BS-2 でしか使ったことはありませんが、他にも ANIMAX やスカパー！等にも対応しているそうです。録画したにはいいけど、右上に入るロゴが邪魔で邪魔でしょうがない人は試してみてはいかがでしょうか。

ここまでツラツラと長く書いてきたんですが、基本的な対策がわかってきたところで復習もかねて実際に僕が使っているフィルタとその順序を載せておきます。うちの環境にはこれがぴったりなんですが、録画環境によっては臨機応変に変化させる必要があります。学問に王道なし、変換道にノーベルあり、です。この記事を参考にがんばって練習しましょう！

1. 「ゴースト (縦線) 除去」(AviUtl 標準)
2. 「クリッピング」(AviUtl 標準)
3. 「YC 伸張フィルタ[15)]」(aLCv for MovieEdit[16)])
4. 「ノイズ除去 (時間軸) フィルタ」(AviUtl 標準)
5. 「Wavelet_NR Type-G」(GNB の館[17)])
6. 「ノイズ除去フィルタ」(AviUtl 標準)
7. 「Lanczos 3-lobed 拡大縮小」(まるも製作所[18)])
8. 「透過性ロゴ」(MakKi's SoftWare[19)])

最後に

というわけで、長かった解説もここで終わり。僕のエンコード歴 1 年の賜物ですよ。その多くの技術の内の一部をここに載せてみました。誰かの参考になれたのなら嬉しいです。後、前回の部誌の丸写しになりますが、最も重要なことを載せておきます。

---

15) 16〜235 の色範囲を 0〜255 に伸張したりする場合や 4:1:1 を 4:4:4 に補完してくれるフィルタ。簡単に言えば明るすぎたり暗すぎたり赤が薄すぎる場合に使います。実際に試してもらった方がわかりやすいのでやってみてください。

16) http://c-zone-web4654.hp.infoseek.co.jp/alcv/

17) http://homepage2.nifty.com/GNB/

18) http://www.marumo.ne.jp/

19) http://mksoft.hp.infoseek.co.jp/

巷で話題の「Winny」や「WinMX」などのファイル交換ソフトについてです。このコーナーで紹介した圧縮技術は僕のような素人にもでき、かつ実用的でサイズも軽いのでそれらのファイル交換ソフトで使用されることが多々あります。

この部誌を手にしたあなたなら大丈夫だと思いますが、ここで紹介した方法をファイル交換ソフトなどで絶対に悪用しないでください。あくまでも「録画した番組を個人で楽しむため」に書いたコーナーです。ここからはあなたのモラルや良心一つでこの技術がプラスにもマイナスにもなります。パソコンの将来のために、絶対に悪用はやめてくださいね。お願いします。

ま、去年と同じで本人はいたって「ほにゃへらぱー」なので硬く構える必要は無いですが、一応載せておきました。変換自体時間かかるので、わざわざそこまでしてファイル交換ソフトに流す人も少ないでしょうし。とにかく、映像変換というのは自分で楽しめればそれでいいんです。これさえわかれば万事 OK ！あなたの変換ライフは快適です！！周りに「そんなの意味が無い」とかいわれても気にしない！あなたはあなたの道を進んでください！！

参考にしたサイト

WikiPedia
http://ja.wikipedia.org/wiki/

BattleEncoderSirase(妖精現実)
http://mion.faireal.net/BES/

Doom9's Forum(注:英語サイト)
http://forum.doom9.net/

アニメのフレームレートまとめ wiki
http://framerate.dyndns.org/

XviD コーディックガイド (DVD ManiaX 2nd)
http://head.egoism.jp/codec/Xvid/

AviUtl のプラグイン (ICZ の剣)
http://cwaweb.bai.ne.jp/~icchan/moviefile/AviUtl_P.htm

・・・・2・・・・

パソコンパーツってこんなもの

61 回生 katukky

さて、みなさんはパソコンの中身について知っていますか？以外と知らない人も多いのでは？（自分も昔はモニターを壊せばパソコンが壊れると思っていたような人だけど）なので、パソコンの中身について語ることにしました。

ＣＰＵについて

まず、パソコンのパーツの中では最も有名と思われる CPU について。これはパソコンの脳にあたる部分です。色々なコンピューターの思考処理はほとんどすべて CPU がしています。CPU にはどんなものがあるのかというと、Windows 用にはほとんど Intel 製と AMD 製が使われており、稀に VIA 製のものが使われています。

速いということで有名な Intel 製は、その速さゆえに発熱量が多くなってしまっています。なぜかというと、一般的にクロック数（動作速度みたいなもの）を速くするには消費電力と発熱量が増すからです。あまりにも発熱量が多くなってしまったので、Intel はデュアルコアやデュアル CPU など、発熱量があまりあがらない方法で速度を上げる方向に進んでいるようです。一方、AMD 系の CPU は、クロック数よりも 1 クロックあたりに処理できるデータの量を増やす事を方針としてきたので、実際の実行速度では Intel に追いついてきている様子です。実際、同じクロック数ならば、AMD 製のほうが速いです。

また、最近出てきているデュアルコア CPU ですが、これは一つの CPU に二つの脳、つまりコアを搭載しているものです。問題点としては、2 つコアを格納するために CPU のサイズが大きくなり価格が上昇する点です。

それに対してデュアル CPU は、CPU を二つ挿してコンピューターを動かすしくみで、前述したような問題は起こらず、デュアルコアより速くなりますが、CPU2 個なのでコストが高くつくという欠点があります。

なお、Mac の PowerPC の最新機種は、デュアルコア CPU を 2 つ積んでデュアル CPU 状態にしており、非常に速くなっています。

ＨＤＤについて

次に、ハードディスク (以下 HDD) について説明します。

HDD はパソコンの情報を保存しておくところで、これが無いと Windows や Mac といった OS すら起動できません。読み書き可能で、後に出てくるメモリよりもずっと安価で大容量ですが、そのアクセス速度はあまり速いとは言えません。そのためメモリと違い、長期的に情報を置いておく場所と考えてください。

さて、HDD の性能は、主に回転数、転送規格、容量、軸受けによって決まります。回転数とは、rpm（Revolution Per Minutes）という一分間に記憶を保存しているディ

スクが何回転するかという単位で表され、7200rpm, 5400rpm, 10000rpm, 15000rpm のものがほとんどです。7200rpm は最も一般的で、最近のものはこの速度である事が多いです。5400rpm は、昔作られていたもので 7200rpm と比べて結構遅いが、価格が少々安くなっています。10000,15000rpm は、最新の物の一部はこの速度です。非常に速いのですが、割とお高くなっています。

次に、転送規格についてですが、Small Computer System Interface(以下 SCSI),Ultra AT Attachment(以下 UATA),Ultra Serial AT Attachment(以下 SATA) の 3 つが主なものです。SCSI は、大容量かつ高速にデータを転送する規格で、主に業務用サーバーなどに使われています。また多くの HDD を連結することができます。最大接続個数は色々ありますが、ここでは言及しません。UATA は、UATA33,66,100,133 という規格があり、数字は転送速度を表しています。一昔前まで、一般ユーザー向けのパソコンの HDD はほとんどこの規格を採用していました。最後に SATA です。これは最近の HDD の主流の規格です。転送速度は 150MB/秒が一般的ですが、300MB/秒も一部あります。また、さらに遅いものもあったそうです。

次に容量についてです。これはデータを容器に例えると、そこに入る水の量みたいなものです。単位は B(バイト) で、英語の文字一文字を記録するのに 1 バイト使用します。ちなみに日本語や漢字を記憶するには 2 バイト必要です。容量は少ないものだと 200MB(メガバイト：約 100 万バイト) 以下、大きいものだと 1TB(テラバイト：約 1 兆バイト) 以上のものがあります。ちなみに、$1TB = 1024GB = 1024^2MB = 1024^3KB = 1024^4B$ です。この 1024 を 1000 と換算することもあります。

次に軸受けですが、HDD をまわすモーターの回転軸の軸受け方式は、ボールベアリングという、HDD の中に入ったボールで軸受けする方式と、流体軸受けという、プラッタとヘッダ (HDD に書き込むためのもの) の間に気体や液体などの流体を挟んで、プラッタとヘッダが直接触れ合わないようにしたものがあります。流体軸受けのほうが音が小さく、モーターの寿命も長いので、最近はこれが主流のようです。

メモリについて

さて、メモリの説明へ。メモリとは、CPU がデータにアクセスする際にいちいち HDD にアクセスすると大変処理が遅くなってしまうため、一時的にデータを置いておく場所です。

メモリの種類は、一昔前の SDRAM というものと、DDR SDRAM(以下 DDR) という今でも現役の規格で、データの送受信量が SDRAM の 2 倍のものと、DDR2 SDRAM(以下 DDR2) という、DDR をさらに改良したものと、DRD RAM という Pentium 4 のための高速ですが高価なメモリとがあります。

性能は、転送速度、CL(キャス　レイテンシの略),ECC(エラーチェック＆コネクト),Reg(レジスタ), 容量, 形状で決まります。まず、転送速度についてですが、DDR1600, DDR2100, DDR2700, DDR3200, DDR24200, DDR5200 などの規格がありますが、決して数字は速さと比例しているわけではありません。それぞれ、200, 266, 333, 400, 533, 667MB/秒となっています (DDR SDRAM はデータを 1 クロックに 2 回データ転送を行えるため、転送速度は $100 * 2$,$133 * 2$……となります)。

CL には、CL2、CL3、CL2.5 などがあり、それぞれ 1,2、1,2,3 と待ってからメモリを読み書きするので、CL2 の方が CL3 と比べて性能はいいということになりますが、普通に使っている限りでは速さの違いはあまり感じないようです。また、一台のパソコ

ンに二つメモリを取り付けると下位互換といって、速度の遅いほうに CL を合わせてしまいます。忘れていましたが、HDD は 2 つ取り付けても転送速度の下位互換はありません。

ECC とは、メモリと CPU や HDD とのデータのやり取りに細かいエラーが無いかどうかを確かめる機能なのですが、エラーが起こることはそうそう無いので、家庭用ではあまり意味をなさないと思います。値段も少し高めなので、無いよりまし程度に思って下さい。ちなみに、メモリを 2 個挿したときは、両方が ECC に対応していないと機能は発揮されず、しかもマザーボードが ECC に対応していないといけません。

Reg とはこれはメモリに特殊なチップ（IC）を追加し、より大容量のメモリを搭載できるようにしたものです。しかしその分、価格は割高になります。大容量にするための技術なので、これは性能とは関係ないといえるかもしれません。

そして容量。これは HDD のところで述べたとおりのものです。32MB〜1GB 位のものがあります。普通にパソコンを使うとき安定に動作させるには 256MB、色々する人には 512MB 以上が好ましいと思います。現在は 256,512MB あたりが主流です。

最後に形状は、３つありあります。DIMM という、最近の一般のメモリ。一昔前のもので、DIMM より取り付け部のピンの数が少なく、普通２枚１組で使用する必要があり、大きさも DIMM より一回り小さい SIMM というもの。最後は RIMM といって DIMM に似ていますが互換性はありません。また、デュアルチャンネルメモリというものがあり、これは、おなじメモリを 2 つくっつけて転送速度を二倍にするというものです。同じ種類 (さっきまで述べた性能やら形状など) であり、かつマザーボードが対応していればほぼ動くきます。ほぼ、といったのは、メモリは相性に左右されやすいので、最悪の場合動かない事があります。各メーカーや店がメモリ相性交換とか言っているのはこのためです。

ディスプレイについて

ディスプレイがあってもパソコンがちゃんとデータを処理しないと、ディスプレイは映らず真っ暗だったり変になったりします。もちろんディスプレイが無くても頑張れば操作できますけどね。

で、画面に表示するデータを処理するのがグラフィックボードと呼ばれるものです。グラフィックボードは、取り付けられる場所がマザーボードには 3 つあって、ひとつは AGP スロット (スロットとは差し込む場所)、これはほとんどのグラフィックボードが接続できます。というのも、このスロットはグラフィックボード専用のスロットだからです。もうひとつが PCI スロット、これは拡張カードなどを取りつけるところで、AGP スロットよりも速度が遅いです。最後に、最先端の規格として、PCI EXPRESS というものがあります。これは、最高性能のグラフィックボード (FF11 対応とか、最新の 3D ゲームが快適に動くようなものすごいやつ) を挿すために作られた新しい規格で、これから主流になっていくであろう規格です。

さて、グラフィックボードの性能は、主に GPU、キャッシュメモリ、転送速度、DirectX に対応しているかどうか、で決まります。

キャッシュメモリと GPU は、グラフィックボード内の、メモリと CPU と思えばいいと思います。ちなみに、GPU は、送られてきたデータを画面に写すときに働きます。

転送速度というのは、マザーボードから送られてきたまたは送る情報の転送速度で、1 ×,2 ×,4 ×,8 ×,16 ×の 5 つの規格があり、1 ×は 266MB/秒, あとはそれを 2 倍ず

つしていった速度です。PCI,AGP スロットは 8 ×までしか対応していませんが、PCI EXPRESS スロットは 16 ×まで対応しています。またこれもやはり、マザーボードが対応していないと、下位互換で動いてしまったり、時には動かないということもあります。

DirectX に対応しているかは、DirectX9.0 に対応しているかと、DirectX8.1 より前かで判断できます。DirectX9.0 を使用するソフトは最近多いので、これもかなり重要です (対応していないと動かない……)。その為、最近のグラフィックボードはほぼ DirectX9.0 に対応しています。

マザーボードについて

最後にマザーボードについて。これがないとパソコンの部品が点々と孤立しているだけで、何もできないということになります。

さて、具体的にどんなものかというと、先ほどまで言って来たパーツを結びつけるものなので、当然パーツを挿す場所があります。今までに言っていない、USB や、マウス、キーボードなどを挿す場所などもほとんどのものはあります。

マザーボードはチップセットというもので性能がほぼ決まるといってよいでしょう。チップセットとは、たとえば使える CPU の種類や、グラフィックボードがどれぐらいの速度で動かせるか、などを決める部品だと思えばいいでしょう。ですからマザーボードが対応していないと、それぞれのパーツの能力が出せません。

そろそろこの文章を終わりたいと思います。みなさんも色々パソコンのデータではなくパーツなどに問題があって、遅いやらうるさいやらとなっていたら、今まで述べたことを参考にしてもらえたらと思っています。

・・・・3・・・・

コンピュータ内での画像処理について

60 回生 ぱんだ

最近は PC も普及して、みなさんもコンピュータを使う機会が増えてきたと思います。その中には、写真を取り込んで編集したり、年賀状の文面をドローソフトで作ったりと、画像を使う処理もたくさんあることでしょう。今日はその処理のことについて話したりしてみたりします。

コンピュータ内のデータの扱い

コンピュータの中では、データは全て 2 進数であらわされる、ということは聞かれたことがあるかもしれません。コンピュータ上でのデータサイズは、バイト (byte) という単位で表されます。1 バイトは、8 ビット (bit) で構成されています。1 ビットというのは、そのデータが 1 か 0 か、ということなので、1 バイトは、00000000〜11111111 まで、2 の八乗の 256 通りの情報を表すことが出来ます。(11110011、10010101、といった感じです）わかり易いように、00〜FF までの 2 桁の 16 進数で表すことが多いですね。

例を挙げると、「部誌の締め切りやべー」という文章は、テキストエディタ (メモ帳等の文章を扱うソフトです）で見ると、部誌の締め切りやべー、と見えますが、実際は、このようなデータになっています。

```
95 94 8E 8F 82 CC 92 F7 82 DF 90 D8 82 E8 82 E2 83 78 81 5B
```

また、データが大きな場合は K（キロ）や M（メガ)、G（ギガ）などをつけて表します。これは長さや重さでも使われるものと同じなのでわかりやすいと思います。「このパソコンのハードディスクは 80GB だー」とか「オレの PC はメモリが 512MB だぜっ」等と扱えるデータの量を表すときにも使われますね。

コンピュータ上での画像データ

コンピュータ上では、画像ももちろんデータの塊として扱われます。デジカメ等で取り込んだ画像も、ペインタ等で描いた画像も全部、同じようにファイルとして保存されます。

コンピュータ上での画像には、大きく分けて二つの形があります。1 つがビットマップ画像、もう一つがベクタ画像です。

ビットマップ画像とは、画像を色のついた点 (ドット) の羅列として表現したデータのことです。小さな点をものすごい数集めれば画像になりますからね。ラスタとも呼ばれることがあります。

ビットマップ画像は、写真などをあらわすのに良く使われます。1 ドット単位で色を変えたりできるので、細かい描写に適しているからです。ただ、ドット単位でしかデータが入っていないため、極端に拡大するとドットが目立ちますし、逆に縮小するとドット当たりのデータが失われたりします。点を大量に集めているので、サイズもベクタ画像に比べるとかなり大きいことが多いです。

また、ビットマップ画像には解像度というものがあります。解像度は dpi(Dots Per Inch) で表されます。これはその画像を表示する際に、一インチ当たりに何ドットが並んでいるかということで、パソコンのモニタは 72dpi です。（一インチに 72 こドットが並んでいる）印刷する時はそれより多く、300〜400dpi ほどあればいいと言われています。

ベクタ画像とは、画像を座標や数式で定義したものです。

ベクタ画像は、縮小や拡大をしてもそれに合わせてコンピュータ側で組み替えられるので、情報が失われることがありません。また、ビットマップ画像に比べ編集が楽な部分もあります。

色の話

パソコン上の色には、主に 2 つのモードがあります。それぞれ、RGB、CMYK と言います。（他にもグレースケール等がありますが）RGB とは、光の三原色、R(Red) G(Green) B(Blue) の三種類で色を表す方法です。それぞれの要素を最も強い光量で混ぜると白になる、加法混合という方法で混ぜられています。

3 つの要素を 0〜255 までの量で表すのが一般的です。つまり、256 の 3 乗で 16777216 色が、理論上は表示可能です。

わかり易いように 16 進数で書くこともあります。

例えば黒なら R、G、B 全てが 0 なので `000000` と表されます。白は全てが 256 なので `FFFFFF` という感じです。

他にも

- 赤:`FF0000`
- 黄色:`FFFF00`
- 抹茶:`C5C56A`
- 群青:`4C6CB3`

等と様々な色を表すことが出来ます。モニタは光で色を表しているので、モニタに出力する画像は基本的に RGB で作られます。

CMYK は、C(Cyan) M(Magenta) Y(Yellow) K(blacK) の四種類で色を表す方法です。RGB が光の三原色なのに対して、こちらは色の三原色＋黒で色を表しています。純粋な C、M、Y を重ねると黒になるので、原色混合と言います。理論上は CMY のみで黒も表せるのですが、印刷時には黒を表すのが難しいので、黒を含めた四つの要素を使用します。ちなみに black が K なのは Blue と間違えないようにしたからだとか。

CMYK は印刷に適しているので、印刷するとき等は基本的に CMYK を使います。また、CMYK は RGB よりも表示できる色数が少なかったりします。

プログラム的な話

画像の加工

コンピュータ上では、データで画像が表されているので、加工がとても簡単です。例えばグレースケール化を考えて見ましょう。

ここでは、NTSC 係数による加重平均法と呼ばれる方法で考えて見ます。この方法は、R、G、B それぞれの値に重み付けをして 3 で割り、平均を取りグレースケール化する方法です。この方法以外にも、重みをつけない中間値法や単純平均法と言う方法もありますが、加重平均法のほうがより自然なグレースケールを作れると言われています。

重み、と言うのは人間が色の違いによって感じる明るさの違いのことです。

例えば、`FF0000` と `0000FF` はコンピュータ上では同じ明るさですが、赤と青では人間は青のほうが暗く（重く）感じます。そこで青の明度を優先的に黒めに変換するようにするのです。

加重平均法では、グレースケール変換後の値 Y は

$$Y = (0.298912 * R + 0.586611 * G + 0.114478 * B)$$

と言う式で求めることが出来ます。（R、G、B はそれぞれ前の色の R、G、B 成分）

画像の各ピクセルの色に対して全てこの処理を行うことにより、画像全体をグレースケール化することが出来ます。

例えば、`FF0000` と赤いピクセルは

$$0.298912 * 256 = 76.521472$$

の明度となりますね。

画像の圧縮

コンピュータ上で画像を扱う際、全ての画像をそのまま利用していたのではサイズが大きくなってしまうので、そのサイズを小さくして使用しようと言うので、様々な圧縮法が考えられてきました。

ここで jpg の圧縮法について述べようかと思ったのですが、色々と事情があって（汗）、代わりに各圧縮法のサンプルでも置いときます。

まずサンプル画像を bmp で用意してみました。サイズは 117 KB です。（ちなみに印刷じゃわかりにくいかもしれませんがカラーです）


jpg:62.1KB


png:29.1KB

とまぁ圧縮したら画質はほぼ変わらず、かなりサイズが下がってますね。てなわけで、bmp のまま保存するのはあれなので、画像は圧縮して保存しましょう。ちなみに jpg は不可逆圧縮（元の画像からデータが一部失われているので完全には元に戻せない圧縮）なので、個人的には png のほうがお勧めですよ。

終わりに

てなわけでコンピュータ上での画像処理について書いてみました。なにぶん急いで書いたので結構かけなかったこととかもあったのですが、まぁその辺は気合でどうにか・・・この記事を読んで、画像を扱うときに色々と考えたりしてみたりしてくれたりするとうれしかったりします。

・・・・4・・・・・

Herbert と過ごすテスト期間

59 回生 Faey

今年も、昨年に続いてプログラミング・コンテスト “Imagine Cup 2006”が開催された。このコンテストはいくらかの部門に分かれている。Software Design 部門や Web Development 部門のようなソフトウェア開発の斬新さを競う部門。制作者の表現力を競う ShortFilm 部門。それから、昨年度我々が出場した Visual Gaming 部門。Project Hoshimi と銘打たれたこの競技には、治療用ナノマシンを制御するプログラムを作り、その優秀さを競うという設定がつけられ、後半、運の要素が大きいと批判されながらも、世界各国のプログラマが熱い戦いを繰り広げた。新聞でご存じかもしれないが、私の後輩がこの部門で優勝を果たしている。

さて、今年度も Visual Gaming 部門は開催された。ところが、競技内容は、数点の追加要素を除けば昨年度とほとんど同じ、という内容であった。常に新しきを求める我々にとって、この部門はもはや興味を引くものではなかった。

そこで我々が目をつけたのが、Algorithm 部門であった。昨年度のこの部門は、アルゴリズムの問題をどれだけ多く解けるかを競う、という試験のような代物であった。その上、この問題はすべて英語で記されていた。学生として英語を多少は解するとはいえ、技術英語を読み解きながら英語圏の人間と戦うのは、非常に不利といわざるを得ない。我々は参加を断念した。

ところが、今年度は違っていた。大会側が用意した全く新しい言語に基づく、母国語に依存しない、純粋なアルゴリズム・コンテスト———

“Herbert”

それが、今年度の挑戦者に課された課題の名前だった。課題の内容は、
「平面を動くロボットを制御するアルゴリズムを最小字数で記述せよ」
この内容は、我々の心を動かすに十分な蜜を含んでいた。

こうして、学年末考査も間近に迫った 3 月初旬、我々の戦いは幕を開けた。

新言語 “H”

ロボットを制御するために大会側が提示した言語は、“H”という名を冠していた。言語の基本的な要素———変数・関数という概念の他には、ロボットの移動命令しか持たない。それは平面を移動するという目的に特化された言語であった。

この言語を用いれば、「4 ます前へ進む」という命令は、たった 4 文字で表現できる。

```
ssss          (4)
```

おわかりのように、1 つの s が 1 ますの前進 (go Straight) を示している。他にも、右折 (turn Right) を示す命令 r、左折 (turn Left) を示す命令 l が存在し、この三つの組み合わせによってロボットは平面を自由自在に行動する。例えば、「右斜めに 2 ます移動する」という命令は、（ジグザグに進むことになるが）以下のようになる。

```
srslsrsl
```
(8)

あるいは、障害物の位置によっては、以下のように動いてもよい。結局のところ、たどり着く位置は同じである。

```
rslsrsls
```
(8)

ところで、この命令が「右斜めに 4 ます移動する」であった場合、どうすればよいだろう。素直に 4 回「右斜めに移動する」という命令を書けばよいのだろうか。

```
srslsrslsrslsrsl
```
(16)

コンテストの趣旨を思い出していただきたい。このコンテストは、“最小字数で”アルゴリズムを書くことを求めている。この趣旨に則るならば、「4 文字を要する命令」が「4 回繰り返され」、16 文字もの文字数を消費していることは、非常に無駄であると言わざるを得ない。そこで登場するのが、**関数**という概念である。

関数

a という名前をつけた関数を考えてみよう。“H”では、関数は小文字で書く。この関数は、srsl、つまり、斜めに移動する、という機能を備えているとしてほしい。つまり、a と書くことで、srsl と書いたと同じことである、とするのである。すると、先の 16 文字は次のように表せる：

```
a:srsl
aaaa
```
(9)

ごらんのように、一行目が関数の定義、即ち、a は、srsl と同じだという宣言である。この新たに定義した命令 a を 4 回書くことで、srsl を 4 回書いたことと同じと見なすのである。“H”では、:,()+-の 6 つの記号を文字と数えないから、この命令は 9 文字に縮んだことになる。ちなみに、かけ算と割り算は使用できない。

このようにして、アルゴリズムの文字数を縮めることができる。例えば、さらに進む数を増やして「右斜めに 9 ます移動する」とする。

```
a:srsl
aaaaaaaaa
```
(14)

しかし、a を 9 回繰り返すのは冗長であるから、新たに関数 b を定義して、

```
a:srsl
b:aaa
bbb          (12)
```

と書けば、全く同じ命令をより少ない文字数で記すことができる。これらが同じ内容であることは、b を aaa で、a を srsl で置き換えることによって簡単に示される。

しかし、この方法には限界がある。例えば、「右斜めに 23 ます移動する」という命令を書け、と言われた場合はどうすればよいだろうか。同じ方法に則って、

```
a:srsl
b:aaaaa
aaabbbb          (18)
```

と書くことは可能である。しかし、このままだと進む数やその約数に文字数が大きく左右される。それを防ぐ新たな概念が**変数**、そして**再帰**という構造である。

変数と再帰

まず、変数とは、箱である。その箱の中には、数字や命令を放り込んでおける。“H” では、この変数を関数と組み合わせてのみ使用する。例えば、次の命令を与えると、ロボットはどんな行動をとるだろうか。大文字で記された A が変数である。

```
a(A):AA
a(s)          (6)
```

1 行目で、関数 a は A という**引数**を取ることが宣言された。この場合、関数 a の意味は、

変数 A を受け取り、中に入っている命令を、2 回実行する

というものである。おわかりかもしれないが、2 行目の意味は、a という関数に直進命令 s を与えて呼び出す、というものである。つまり、この命令によって、ロボットは「2 ます直進する」ということになる。ちなみに、この際、A には命令しか入れることができない。その理由は、A に適当な数字を入れて、展開してみればよくわかる。その数字は全く意味をなさない。

この方法を用いれば、例えば「右斜めに 3 ます移動し、さらに左斜めに 3 ます移動する」という命令を、

```
a:srsl
b:slsr
aaabbb          (16)
```

と書く代わりに、

```
a(A):AAA
a(srsl)a(slsr)          (15)
```

と書くことができる。これで 1 文字縮めることができた。
　ちなみに、関数は引数を複数取ることができる。例えば以下のように：

```
a(A,B):AAABBB
a(srsl,slsr)          (18)
```

この例では望ましくないが、2 つ、あるいは 3 つの引数をとることで縮められる問題も存在した。

　さて、やっかいなのが再帰である。これはプログラマにとってもなかなか御しがたい概念で、しかし非常に重要な概念の一つである。幸いにして、この “H” は単純な言語であるから、再帰も用途が限られ、さほど難しくないと思われる。では、具体的な例を挙げながら説明してみよう。
　まず、再帰とは、文字どおり「再び帰る」———つまり、関数が「自分自身を呼び出す」ことである。これだけでは何のことかわからないだろうから、再帰の例を一つ挙げる。

```
a(A):srsla(A-1)
a(23)                 (11)
```

ごらんの通り、関数 a(A) は、その最後で自分自身 a(A-1) を呼び出している。この命令を与えると、ロボットはどう動くだろうか。一つ一つ展開していくと、この命令は次のように働くことがわかる。

```
srsla(22)
srslsrsla(21)
    ···
srslsrsl ··· srslsrsla(0)
```

実は、“H” は、関数の引数が 0 以下になると自動的にその関数の実行をやめるという特性を持つ。関数 a はそれぞれ 1 回ずつ「右斜めに進む」という命令を持っていることを考えると、この命令は
「23 回右斜めに進む」
という意味であることがわかる。
　おわかりだろうか。この再帰を使うと、実行回数を数字で指定することができるのである。これが、先の疑問の答え———回数が増えたときに、どのように字数を減らすか、である。数字は何桁であろうと 1 文字として扱われるため、単純な「斜めに n 回進む」という命令は、最大 11 文字で書けることが決定された。
　それでは、続いてさらに複雑な命令に挑戦してみよう。「右斜めに 13 ます、さらに左斜めに 13 ます進む」という命令を最小字数で書くには、どう書けばよいだろうか。

　まず考えられる答えは、

```
a(A):srsla(A-1)
b(B):srslb(B-1)
a(13)b(13)          (22)
```

である。ところが、関数は複数の引数を取ることができる。つまり、これを下のように書き直すことができる、ということである。

```
a(A,B):Aa(A,B-1)
a(srsl,13)a(slsr,13)          (20)
```

さて、この命令はこれ以上縮めることができないのだろうか。それが、できるのである。

堀江理論・応用堀江理論

ここから紹介するのは、我々が発見し命名した数々の特殊技巧の一つである。その他の技巧は、コンテストの問題を解説するうちに紹介する。

堀江理論とは、再帰の性質を利用した繰り返しの技巧である。なぜ堀江と名前がついているのかについては深く問わないでほしいが、関数の働きをみれば、わかる人にはわかるだろう。

堀江理論は、次のような構造を持つ：

```
a(A):sa(A-1)r
a(6)
```

注目していただきたいのは、再帰の後に来ている右折命令 `r` である。この 1 文字を付け加えることで、どのようなことが起きるかということは、今まで通り展開すればすぐにわかることである。

```
sa(5)r
ssa(4)rr
    ...
ssssssa(0)rrrrrr
```

おわかりだろうか。つまり、再帰の後にたった一文字 `r`（または `l`）をつけることで、ロボットは「くるくるまわる」のである。この場合、6 回右に曲がっているのであるから、ロボットはちょうど来た方向を向くことになる。回数に応じて、最後に向く方向が調節できるのが、この堀江理論の特徴である。

さて、これを応用するのが**応用堀江理論**である。だが実は、応用といってもたいしたことではない。この堀江理論は、「まわる」ことにこだわらなければ、

> 指定された回数の命令の後に、同じ回数だけ別の命令を実行できる

という意味にもとれる。これが応用堀江理論である。これを用いて、先の 20 文字の命令を縮めると、次のようになる。

```
a(A):srsla(A-1)slsr
a(13)
```
(15)

これが、右斜めに 13 回進んだ後、左斜めに 13 回進む、ということを意味するのである。この技巧を用いることによって、わずか 15 文字まで縮めることができた。

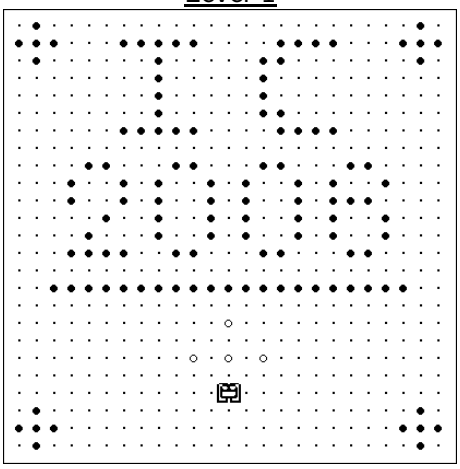
さて、ここからは実際のコンテスト問題の解説に入る。以下に示された解答より短い解答が得られた場合、

```
http://npca.my-sv.net/
```

にアクセスして、「御意見板」や「恋文投函」に書き込みをどうぞ。

実際の問題解説

Level 1

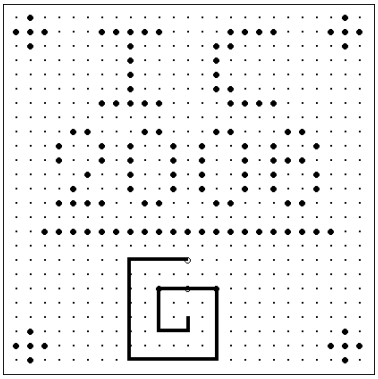


制限字数: 20 文字

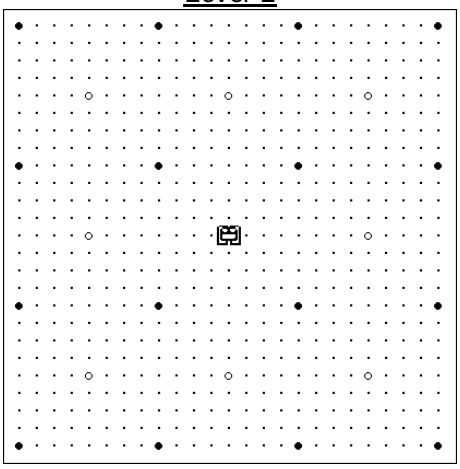
第一問で、問題のルールを説明しておこう。まず、中央下部にある不可思議な形のマークが、ロボットである。つまり、ここがスタート地点であり、このロボットが最初に向いている向きは上である。お察しの通り、このロボットは点の上にのみ立ち止まることができる。白い○をすべて踏むことによって競技は終了となり、逆に黒い●を踏むと、今まで踏んだ○がすべてリセットされてしまう。例えロボットがそこで止まらなくても、すべての○を踏んだ瞬間にロボットの動作は終了する。

この競技をするにあたって、最も重要なことは「パターン化できる」軌道を見抜くことである。本問において、踏めばよいのは下方の○ただ 4 点のみである。さて、これらをもっとも少ない文字数で踏む手順を考えてみてほしい。

もっとも効率の良い軌道を見つけることができれば、この問題は、わずか 8 文字で解くことができる。ヒントは、以下の軌道である。解答は次のページの下部に書いてあるが、挑戦したい方は、ページをめくらず、鉛筆を持ってきてじっくりと考えてみて欲しい。



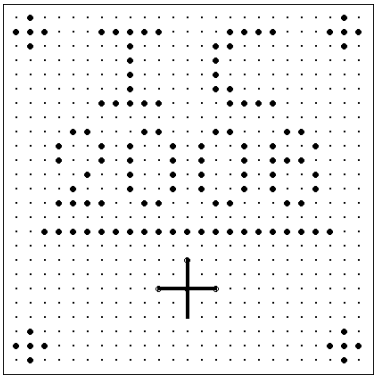
Level 2



制限字数: 20 文字

Level 1 解答

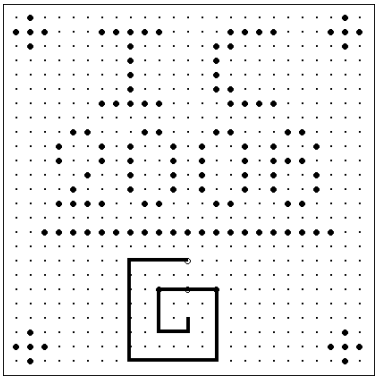
一見すると、中央の○に至り、そこから周囲 3 つの点を取りに行くのがもっとも良いルートであるように見える。その場合の最短命令は、次のようになる：



```
a:ssrrssra
ssa                    (12)
```

中央の○まで到達すれば、方向こそ違えど、残りの 3 つの点を取る動作は全く同じである。即ち、「2 ます前に進み、回れ右して 2 ます進む」動作を繰り返せばよいのである。すべての○を取るためには、戻る度に進む方向を 90 度変えてやればよい。

しかし、アルゴリズムの観点から言えば、スタート地点から最後まで全く同じ動作をした方がよい。この場合はどうすればよいだろうか？———回るのである。

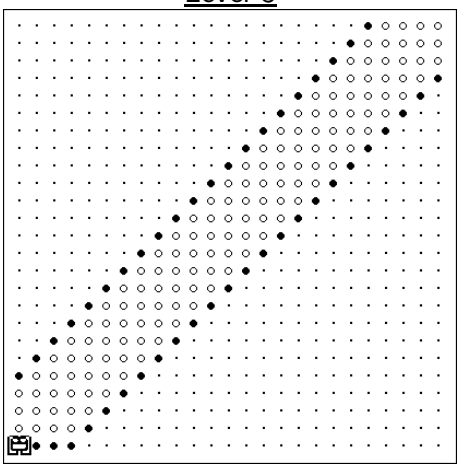


```
a(A):rAa(As)
a()                    (8)
```

最初、関数 a は空の引数を受け取るから、ただ右に曲がるだけの動作をする。その後、自分自身の引数に直進命令 s を次々と付け加えながら、徐々に大きく回っていくのである。

最初の回れ右は狙ったわけではなく、縮めると結果としてそうなることの方が多い。

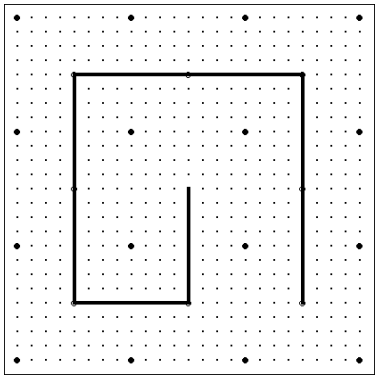
Level 3



制限字数: 13 文字

Level 2 解答

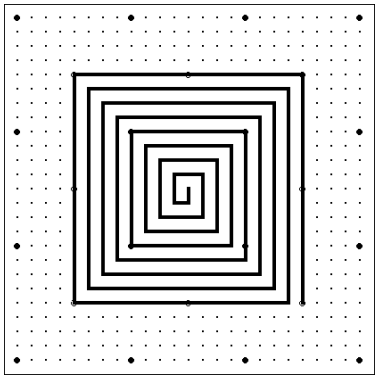
これまた一見すると、以下のように回るのが手っ取り早いように思える。その場合の最短命令は、次のようになる：



```
a:ss
b(B):BBb(aaaaB)
b(r)                    (15)
```

2 回ずつ、同じ距離を進んで右に曲がるという動作を繰り返す。それぞれで進む長さを 8 長くすれば、図のような軌道を描く。関数 a は、1 文字縮めるためのくくりだしである。

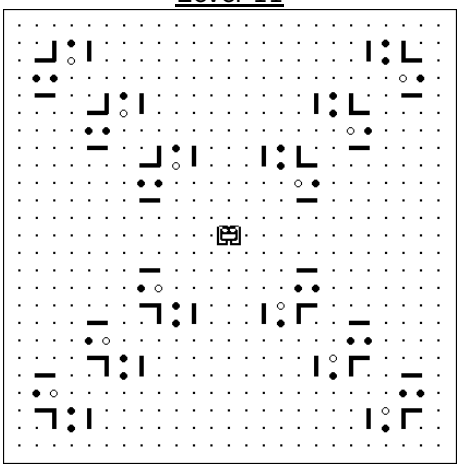
ここで、●の特性を思い出して欲しい。踏めば「今まで踏んだ○をリセットする」ということは、まだ一つも○を踏んでいない場合、踏んでも平気なのである！



```
a(A):AAa(sA)
a(r)                    (9)
```

関数 a は、自身の引数の前に前進命令 s を付け加えながら、どんどん回っていく。途中で●を踏もうがお構いなしである。

Level 11

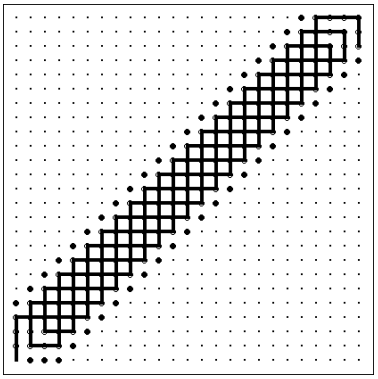


制限字数: 29 文字

Level 3 解答

この問題を解くのは少々やっかいに見える。最初と最初が多少とはいえ不規則に見えるし、斜めに移動しているだけでは制限字数を超えてしまう。

そこで目をつけるのが、制限字数である。上限がわずか 13 文字であるということは、単純なパターンを最初から最後まで、あるいはほとんどの部分を繰り返すだけで解けると考えられる。そのことを念頭に置いて、軌道を見つけると、以下のような軌道になる。



```
f:ssr
a:sfsfffa
a                              (13)
```

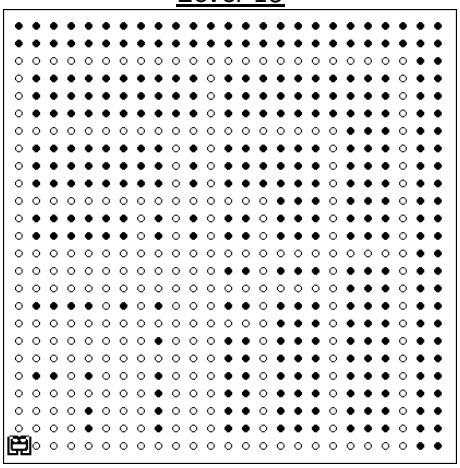
要するに、「3 ます進んで右折」*2、「2 ます進んで右折」*2 の繰り返しである。実は、本問はこれ以下の文字数で解くことはできない。

本問のように、文字数がヒントになって解けてしまう問題はいくつか存在した。Level 11, 17 などは、制限字数がそれぞれ 11 文字、8 文字しかなく、問題を解くというよりはその文字数で考えられるパターンから詰めていった方が早かった。また、Level 48 は高難度問題だが、軌道はすぐに見つかり、16 文字に至るのもすぐである。この問題が Level 48 たるゆえんは、制限字数が 15 文字であることにあるのだが、これもパターンの区切りを変えれば縮めることができる。

高難度問題になると、期間中フォーラム[1)]などに「どうしても解けない」というスレッドが立ったりしておもしろい。もちろん誰もヒントは漏らさない。

[1)] theSpoke.net の英語版に、Algorithm 部門のカテゴリがある。そこに苦悩する外国人さんが書き込むのである。

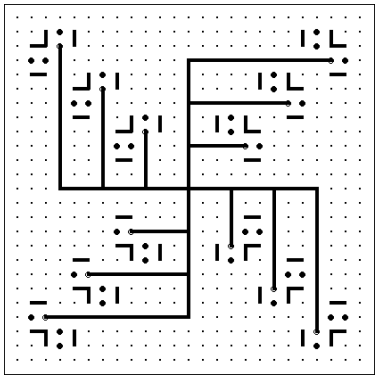
Level 15



制限字数: 24 文字

Level 11 解答

Level が二桁にもなると、だんだんと制限字数に納めるのが難しくなってくる。また、高得点を狙うためには、より少ない字数で解く必要があり、本問のような問題は障害になる。説明のために過程は省き、最短を示す。



```
a(A,N):AsArsAllAa(A,N-1)a(sssA,4)
a(r,4)                                (25)
```

一つ○を取ったら原点に戻り、同じ距離にある 4 つを取った時点でより外側の 4 つを取りに行く。これは非常に最適化されているが、ここまでしなくても解ける。関数の引数は必ずしも同じ種類でなくともよい。

注意して欲しいのは、ここで「自分自身を呼ぶ」つまり再帰を、自分で 2 回行っていることである。より左にある再帰の呼び出しでは、残り回数を 1 回減らして呼んでいるが、右側の呼び出しでは定数 4 を回数として指定している。つまり、これは無限ループであり、実際はすべて○を踏んでもどんどん外側の○を取りに行こうとする。一種の応用堀江理論だと言ってもよいかもしれない。

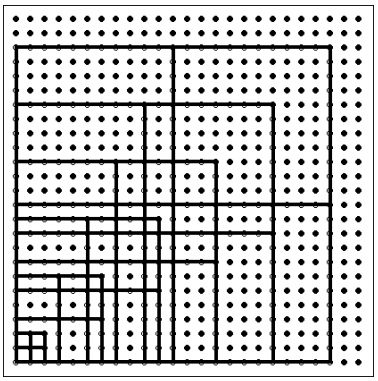
また、関数の中で、`All` という文字列があるが、これは A の中に入っている `r` を打ち消して、さらに左折しようというものである。ちなみに、関数の最初の呼び出しで `r` を入れるのはよくあることである。入れなければこうなる：

```
a(A,N):ArsArrsAlAra(A,N-1)a(sssA,4)
a(,4)                                 (26)
```

これだけで数文字縮められたりする。

Level 15 解答

この問題は、少々軌道が見つけづらいかもしれない。「田」という形をした、大きさの違う軌道がいくつも重なっている、と言えばわかるだろうか？ぐちゃぐちゃしているが、以下が軌道である。



```
a(A,N):AAlAAa(A,N-1)a(ssA,4)
a(sr,4)                                (21)
```

Level 11 のような形式で解くとなると、このような命令になる。ホッチキスの針のような形をした軌道を 4 回繰り返すことで、「田」を表現している。あとはサイズを大きくしながら繰り返すだけである。r のくくりだしを外すと軌道がよくわかる。

ところが、これには非常にすばらしい解が存在する。この軌道、よく見ると同じ命令が繰り返されることが非常に多い。関数 a が「田」一つを表すように関数を組み替えると、

```
a(A):ArAArArArAArArArAArArArAArAra(ssA)
a(s)
```

となる。a の中身を共通部分に分解すると、

```
ArAArArArAArArArAArArArAArAr
= ArAArArArAArAr ∗2
= ArAArAr ∗2 ∗ 2
= ArA ArA r ∗2 ∗ 2
```

そこで、その命令を 2 回ずつセットにして考えると、以下のようなコードが生み出される。

```
f(A):AA
a(A):f(f(f(ArA)r))a(ssA)
a(s)                                   (19)
```

よりハイレベルな問題について

紙面の都合上、問題の解説はわずか 5 問、それも前半の簡単な問題ばかりとなってしまったが、後半の難問も、軌道さえ見つければだいたいがこれらの技法で解ける。しかし、最後の 10 問ほどは、3 つの変数を使い、階差的な軌道を表すという超高難度技法を使わなければ解けないものばかりで、Level 46 などは、全参加者のうちたった 2 人しか解を導くことができなかった。

ヒントは、3 引数 A,B,C をとり、B が A に、C が B に影響されるような関数である。興味のある方は、theSpoke.net などへ行って、競技者有志が公開している問題をダウンロードし挑んでみて欲しい。

・・・・5・・・・

人工無能のつくりかた -文章生成の初歩の初歩-

60 回生 Jyakky

さわやかな初夏の風が・・・いや、違ったか。ええと、本日は灘校文化祭へようこそ。わが部は特殊すぎる事情によって五月二日も半分ほど準備に費やしている可能性が非常に濃厚ですが、ご容赦ください (ぉ

さて。編集担当の Faey さんが部長である僕の言うことを聞いてくれればこの文章は真ん中よりちょっと後ろの目立たなさそうなところに溶け込んでいるはずなのですが、いかがでしょうか。

♯ いや、別に読んでほしくないだとか書きたくなかったとか、そういうんじゃないですよ？全然。

本当は、「暗号入門」を書こうと考えていたのですが、締め切りが一週間後まで迫った段階でまだ「考えて」いる段階だったので、ちょうどその頃単純な人工無能の開発を暇つぶし程度に行っていたこともあって、人工無能について書くことにしました。国語の成績が年々下降し、とうとう通知表を夕焼け色に染める程になってしまった僕の文章なので、全体的に読みにくいこと甚だしいかと思われますが、そういう運命だと諦めていただけるとうれしいです。

ところで。この部誌というものが対象とする読者の年齢層がどの程度なのかさっぱりわからないので、どういったことを書けばいいのか混乱しているわけですが、やはりいきなりプログラムのソースコードを載せるわけにもいかないので、簡単な解説だけにとどめておくことにします。「簡単」かどうかはともかくとして。

人工無能を知らないあなたへ

♯ もしあなたが人工無能についてそれなりに知っている (or 知りたくもない) ならこの章は読み飛ばしてもいいです。

> **人工無能**
> 主にチャット等で発言の中のキーワードに反応して適当な対応を返すプログラムのこと。従来の人工知能研究とは異なり、会話における『表象の現象だけ』を考えて会話をシミュレートしようとするアプローチを取る。
> 古くは Eliza から最近では「どこでもいっしょ」のトロに至るまで、様々な人工無能がある。
>
> (はてなダイアリー キーワードより)

ええと、まず「人工知能」はわかりますよね。まぁ細かい定義までは僕も知らないのでそのあたりは Google 先生にでも聞いてもらうとして (おい)、とりあえずイメージとしては「コンピュータのくせに人間みたいに考えたりする生意気な奴」という感じで

しょうか[1]。いや、とりあえず僕はそう思っているのですが、ならば人工「無能」とはなんぞや。「能が無い」などというレッテルなど貼ってしまって外交問題に発展したりはしないのか。しないだろうけど。

・・・脱線しすぎました。とにかく、人工知能は人間のようにいろいろ考え (るように研究されてい) ますが、人工無能はなにも考えちゃいません[2]。というか、初歩的な人工無能は、覚えている言葉をランダムに組み合わせて話すことしかできません。人工知能は言葉を理解して受け答えしようとしますが、人工無能は正直そんなことあきらめています。「それっぽい」受け答えができればいいや、というコンセプトです。そのため、人工知能とは違い、「それっぽさ」を追求する研究が多く行われています。

♯ ずいぶん大雑把な書き方してしまったな・・・まぁいいか。

ごくごく初歩の文章生成

上記のように、人工無能というのは「それっぽさ」が命なので、「それっぽい」文章を自動で生成するアルゴリズムが要となるわけです。なので、以下ではそのあたりのことを少しばかり書いて行くことにします。

♯ まぁたいしたことできないのだけど。

文章を生成する、と聞いて最初に思いつくのは、「とりあえず元の文を用意しておいて、単語を適当に置き換える」というものだと思います。この場合、例えば、

元の文「A って B よね」から、
「ネコ って かわいい よね」
「夏 って 暑い よね」
「宿題 って だるい よね」
「俺 って かっこいい よね」
etc.

という感じの文章が生成できます。

♯ いや、文章の内容は気にしない方針で (ぉ

とにかく、こうすればあとはたくさんの元の文と単語を保存しておけば、基本的に文法的には完璧な文章が自動で生成できます。「しくみも簡単だし文法上も無問題！まるで魔法だ！」とか「もうこれで完成でいいんじゃないの？」とかいう声が聞こえてきそうですが、違います。断じて違います。我々の目指すところはもっとはるかなる高みにあるのです。そうだ、そうに違いない。

♯ クールダウンクールダウン。

・・・何が問題なのかというと、「型にはまった応答しかできない」ということ。「型どおりの返答ができる」というのは裏を返せばつまりはそういうことです。このままではちっとも面白くありません。それどころか退屈です。文章生成計画、早くも挫折か。

あるいは、「文法とかどうでもいいから単語並べちゃえ」というアプローチもあります。これなら、文章が一定の型にとらわれるなどということはまずありません。例えば

---

[1] 実際はちょっと違うんですが。　て。
[2] まぁ「考える」がどういうことか、は置いておい

「来たは数学でもあった返信普通と急いで」
「記号も一般的は違いかな分からないいや良くない」
etc.

という感じです。悲惨です。当然です。文法を無視してしまったら意味が成立するわけないのは火を見るより明らかです。話しかけたらこんな返事が返ってきたのでは、驚きを通り越して悲しみまで覚えます[3]。文章生成計画、早くも挫折か。

マルコフ連鎖で手軽な作文

マルコフ連鎖 (Markov Chain)
ある変数を順次、発生させるときに、現世代のパラメタの値 (のセット) のみをもとに次世代のパラメタの値を発生させる方法

(はてなダイアリー キーワードより)

・・・これだけでは何のことやらさっぱりなので、具体的にどのように文章を生成するのか説明することにします。

♯ そこ、「ページ稼ぎ」とか言わない！

・まず、文章を用意します。

例) 今日も明日も学校があるなんて信じられない。

♯ 繰り返すけれど、文章の内容は気にしない方針 d(略)

・次に、文章を形態素[4]に分割します。

例) 今日　も　明日　も　学校　が　ある　なんて　信じ　られ　ない　。

・形態素を連続する二つごとに並べます。

例)

(先頭)	今日
今日	も
も	明日
明日	も
も	学校
(省略)	(省略)
ない	。
。	(末尾)

・二つのうち前半が同じものをまとめます。

---

[3] まぁこっちのほうが面白い、という人もいるかもしれませんが。

[4] 言語の中で意味を持つ最小単位のこと。

例) 例文が短いのでいまいちわかりにくいのですが、

(先頭)	今日
今日	も
も	明日 学校
明日	も
(省略)	(省略)
ない	。
。	(末尾)

・これを多量の文章で行うと、「ある形態素の次にどのような形態素が続くか」の傾向が見えるようになってきます (「学習」みたいなものです)。

例) 使った文章は割愛しますが、

(先頭)	今日 今日 今日 もし だいたい そもそも いや
今日	も は は
(以下)	(省略)

・文頭からはじめて、次に続く形態素の中からランダムに一つ選んで文章を伸ばしていきます。

例) 今日　は　やはり　学校　が　、　しかし　実際　に　行く　。

これで完成です。簡単ですよね。ただ、これではぜんぜん意味のある文章にならないので (まぁもともと意味なんか考えてないけれど)、ここからもう少し改良することにします。

手軽な改良計画

1.二重マルコフ連鎖

マルコフ連鎖+「二重」。だいたい意味は分かると思いますが、単純マルコフ連鎖が直前の一形態素から次を決定するのに対し、手前の二形態素から次を決定する方法です。単純マルコフ連鎖はどうにも日本語と相性が悪いようなのですが、このようにすればその問題もかなり解消されます。同じように、三重にすることもできますが、あまりさかのぼり過ぎるとより多くの例文が必要になったり、保存するデータの量が増えてしまうという問題点もあります。

♯ 実装もかなり面倒になるし (ぉ

2.双方向マルコフ連鎖

マルコフ連鎖+「双方向」。これまでは文章の先頭→末尾への解析しかしませんでしたが、末尾→先頭への解析も行うようにします。このようにすれば、文章の先頭に限らず、任意の単語から前と後ろに同時に文章を延ばしていくことができます。

♯ ということは、主語を種にすればきちんとした文章が作れるということかもしれない。

3.品詞で形態素を区別する

そのままです。形態素を文字の並び+品詞で区別することによってより精度の高い文章生成が行えるようになります。

とりあえず、現在すでに実装されているのはこの三つです。他にもやらなければならないことや考えていることはいくらかあるのですが、それは次章に。

今後の指針のようなもの

さて。この章の内容は基本的に未確認情報です。実際に実装したわけではありません。ただ、これを読んだ方で、もしも人工無能を作ってみたいという人がいれば、参考になるかもしれないと思ったので書くことにしました。

♯ あと自分用メモとして。

いや、別に暗殺された天才科学者の手記とかそういうたいそうなものではないのですが。

1.前後のつながりをもっと大事に

「双方向」とはいっても、結局文章を伸ばす時には一方向のつながりしか考えていないのですが (A B → C のときの B ← C を考えていないということ)、ここで逆方向のつながりも考えるとより自然な文章が生成できるかもしれません。

2.そもそもマルコフから離れる

今のアルゴリズムは、文章の形を自然にすることにこだわりすぎて、意味をほとんど考えていないので、もしかすると新しい方針が必要かもしれない。例えば、品詞だけでマルコフ連鎖を作って、それを仮想的な「文法」として、そこに単語を当てはめていく。同時に単語と単語のつながりをうまく定義して単語のネットワークのようなものを形成して、とかも考えているのですが (というか、仮想文法を作るところまではすぐできるかもしれない)、単語同士の関連性をどのように表すかがさっぱりわからないので挫折しています (ぉ

おわりに

長かった・・・、いや、実際にはそれほど長くなかったのですが、なにぶん無理やり書いたのでどうも長く感じたわけで。それでも後半はだいぶ勢いに乗れたかな、という気はします。

さて。こんなつまらない悪文を最後まで読んでくれたあなたのことです。きっとあふれんばかりの才能の持ち主なのでしょう。いや、きっとそうです。僕がそうだといえば

そうなのです。

灘校パソコン部は、そんな才能あふれる奇特なあなたをお待ちしています。もしあなたが灘校生なら、学年は問いません。旧校舎四階、地学教室横の薄汚い小部屋まで見学に来てください。それと、あまり乗り気でないあなた。長い人生、そのくらいの寄り道も悪くないものです。ここらで、パソコン部を道連れにするのもあるいは楽しいかもしれませんよ。

♯ 要は見学に来てくれ、ということなんだけど (ぉ

ともかく、もう夜も遅い (AM04:09) ので、ここらでペンを置くことにします。提出期日を五日もオーバーしているのに快く編集を引き受けてくれた Faey さん、ありがとうございました。

♯ まぁ「快く」かどうかは聞いてないから知らないのだけれど。(というか多分違う)

それでは、来年こそは「暗号入門」が書ける事を祈って。

・・・・6・・・・・

油断大敵ウイルスまみれ？

59 回生 Jack

今、話題沸騰の彼をご存知ですか？ 彼は多くの人に影響を及ぼし、多数の企業が彼に出資するお金は相当な物です！ マスコミも彼をこぞって取り上げ、彼についての記事やウェブサイトが五万とあります。そう、この記事はそんな彼「コンピューターウイルス」とそれを巡る攻防についてお話しします。

なお第 1 章は基本的な知識の解説の為、分かる人は読み飛ばして頂いて構いません。

基本知識

『実行ファイル』と『データファイル』

パソコン上にあるファイル（データ）は大きく分けると 2 つに分けられます。『実行ファイル』と『データファイル』です。『データファイル』は文字や画像のデータなどが格納されています。一方『実行ファイル』にはパソコンに対する命令が格納されています。『データファイル』と『実行ファイル』を見分けるには拡張子を見るのが一番です。拡張子とはファイルの末尾についている『.txt』や『.jpg』などの文字です。この部分で Windows はファイルの種類を見分けています。『.txt』はメモ帳などで作成される、文字データで出来たファイルで、『.jpg』は写真のデータなどを格納する画像ファイルです。『実行ファイル』はファイルの末尾に『.exe』や『.com』がついているもので、コンピューターに対する命令が格納されています。また、厳密には実行ファイルではありませんが拡張子が『.vbs』や『.js』はスクリプトファイルと言って、実行ファイルに比べて機能の制限がありますが、実行ファイルのようにコンピューターへの命令が格納されています。

「ファイルの後ろにそんなのついてないよ？」といわれる方がいるかもしれませんが、それは Windows の設定で表示しないようにしているだけで、設定を変更すれば表示されます。この拡張子は非常に重要で、これを変えると Windows はそのファイルに対する動作を変更してしまいます。例えば日記を書いた『Nikki.txt』を『Nikki.mp3』にファイル名を変更すると、本来メモ帳が起動し『Nikki.txt』の内容を表示する筈が、『Nikki.mp3』を起動して音楽を再生しようとします。当然、Nikki.mp3 の中には音楽のデータは入っていないため音楽は再生されません。

感染までの攻防

知らない人はあまりいないと思いますが、コンピューターウイルスとは単純に言えば感染したコンピューターに使用者が望まない動作をさせるものです。私たちが覚えなければいけないのは「コンピューターウイルスを感染させるにはコンピューターに命令を実行させなければならない」という非常に重要な事実です。すなわちデータファイルを開いただけでは、基本的には（あくまで基本的には）ウイルスに感染しません。危険な

のは『実行ファイル』を開いた時です。実行ファイルにはコンピューターに対する命令が含まれるため、色々な悪意に満ちた行動をコンピューターに行わせる事が出来ます。例えばデータを全部消したり、無意味に CD トレイを開け閉めしまくったり。

そこでコンピューターウイルスが狙ってくるのは常に『相手にウイルスを感染させる実行ファイルを実行させる』事なのです。それを巡ってウイルス作者とウイルス対策側は常に知恵を働かせます。その知恵の数々を紹介しようというのがこの章の趣旨です。

ケース：Happy99

コンピューターウイルスが狙ってくるのは常に『相手にウイルスを感染させる実行ファイルを実行させる』事、と言いましたが、その『ウイルスを感染させる実行ファイル』はどこから来るのでしょうか？もちろん人間にとってのウイルスの様に空気に漂って来るなんていう事はありません（そんな事になると大惨事）。昔は、CD やフロッピーディスクに入っている事が多かったのですが、最近の実行ファイルは大半がインターネットからやってきます。インターネットにある『ウイルスを感染させる実行ファイル』をダウンロードして実行すれば、そこでもう負けです。ウイルスに感染してしまいます。しかし、普通は『ウイルスを感染させる実行ファイル』をインターネットで配布すればたちまち「あそこはやばい」と噂がたちばれてしまいます。そこで電子メールに実行ファイルを添付して相手にダウンロードさせる手法がかなり用いられています。

しかし、もし知らない人から「これはウイルスですよー」というタイトルのメールに添付ファイルがついていたら貴方は実行しますか？しませんよね。しないで下さい。

この Happy99 もそんなメールの添付ファイルによって広がるウイルスの一つです。このウイルスはメールに添付された『Happy99.exe』というファイルを実行すると感染します。このファイルを実行すると花火の絵が表示されるのですが、そんな物は囮で、実際はインターネットに接続する際に実行されるソフトを改ざんして、メールを送るたびに『Happy99.exe』を添付させる様にします。

そう、このウイルスの凄い所はここです。これによって受け取った側は「おっ、○○からじゃーん」と知り合いからと油断して添付ファイルを実行してしまいます。そして感染した人からも『Happy99.exe』が添付して送信され、その知り合いにも……。と、どんどんウイルスは拡大していきます。

教訓：メールに実行ファイルが添付されていても開かない。それがたとえ知り合いからのメールでも。

ケース：ラブレター

このウイルスも Happy99 と同じ様にメールで広がるウイルスです。このウイルスは感染した人から「I love you」というタイトルで送られて来て、本文には「私のラブレターを添付しました。是非読んでください」と書かれており、「LOVE-LETTER-FOR-YOU.TXT.vbs」というファイルが添付されています。

このウイルスで注目すべき点は 2 つあり、ひとつは添付ファイルである「LOVE-LETTER-FOR-YOU.TXT.vbs」の拡張子です。本来、このファイルの拡張子は「.vbs」で、スクリプトファイルです。普通スクリプトファイルを受け取ると、「うわっ、あやしー。こんなファイル開けねーよ」と言う事になるのですが、この場合 Windows で『拡張子を表示しない』という設定にしていれば本当の拡張子「.vbs」は消され、拡張子が

「.txt」であるかの様に表示されます（LOVE-LETTER-FOR-YOU.TXT）。それを実行すると、はいアウト。ウイルスに感染です。

もうひとつはそのタイトルです。「I love you」、……いい響きです。私はこの学校に入ってからラブレターなんざ貰った事がありません（同級生にもらっても嫌ですが）。もし、このウイルスの発信主がウイルスに感染した知り合いの女の子だった場合は思わず開いてしまうかもしれません。そういう人間心理をついたタイトルでこのウイルスは拡大していきました。このウイルスの作者は悪魔だと思います。

教訓：拡張子は表示する。冷静に考えて、来ないラブレターは警戒する。

ケース：山田ウイルス

ファイル共有ソフト上で配布されている事が多いウイルスです。このウイルスは正常なファイルを装っています。稚拙な物は「ファイル名.txt　　.exe」と本来の拡張子を遥か後ろにする事によって、ファイルを一覧表示した場合に「.exe」の部分を画面外にはみ出させ、無害なファイルと思わせ実行させます。

なお高度な物は、本来のソフトウェア（ゲームのインストーラー）を装い、起動すればゲームが動くのですが、それと同時にウイルスに感染させます。この場合、拡張子は「.exe」だと分かってファイルを開く為、拡張子でウイルスだと判断するのは不可能です。

近頃話題の Winny による情報流出はだいたいこのウイルスと似た構造を持っています。まあファイル共有をしない限りは触れる機会はあまり無いと思われるので、そんなに警戒しなくてもいい様な気がします。

教訓：とにかく拡張子に注意。怪しい実行ファイルは実行しない。

ケース：メリッサ

さて、今までのウイルスの拡張子は『.exe』や『.vbs』など、実行ファイルのものでした。ですがこのウイルスの拡張子は『.doc』、そう Word と同じなのです。このウイルスは Word ファイルのマクロ機能を使って命令を実行させます。そのため、一見普通の Word ファイルにしか見えません。そのためこのウイルスもかなり感染が拡大したらしいです。似たようなものに Excel のマクロ機能を使った Laroux（ラルー）というウイルスがあります。

またこのウイルスは、メールのアドレス帳にある人に自分を送信しまくります。これによって『Happy99』や『ラブレター』の様に、「知り合いからのメールやから大丈夫やろー」と油断させて感染を広げるのです。しかも拡張子が『.doc』であるため警戒はしづらいです。しかも、本文には「ご依頼の文章です」とどうとでもとれるものが入れられます。

このウイルスはウイルス対策ソフトが無いとちょっと対応しづらいですね。

教訓：がんばって！

ケース：Blaster

これはエグい。今まで紹介したウイルスはコンピューターの使用者が注意すれば防げ

ましたが、このウイルスは違います。このウイルスは Windows のセキュリティ上の欠陥を利用して、知らない内にウイルスに感染する命令が実行されます。しかもこのウイルスはコンピューターをつけてもつけても強制的に終了させるため、コンピューターに詳しく無い人にとっては対策は困難です。

とりあえず対策としては WindowsUpdate を使用して、Windows のセキュリティ上の欠陥を訂正するしかありません。パーソナルファイアーウォールやルータのパケットフィルタリングを使用する方法もありますが、少し知識が無いと難しいと思います。

まあ最近では、Windows 上での対策も進んでおり、自動的に WindowsUpdate をしてくれたり簡単なファイアーウォールを設定してくれたりはするみたいなので、そんなに心配する必要は無いと思いますが、一応自分のパソコンの状態を確認してみるのがいいと思います。

ケース：Nimda

このウイルスは色々な感染経路がありますが、私たち一般ユーザーに主に関連があるのは、『ウェブページを見ると感染する』ケースです。

私たちが Nimda に感染したウェブページにアクセスすると、Internet Explorer というウェブブラウザ（ウェブページを見るときに起動する青い『e』の奴）のセキュリティ上の欠陥を利用してウイルスに感染する命令が実行されます。これも上にある Blaster ウイルスと同じで、注意すればなんとかなるというものではありません。とにかく WindowsUpdate で欠陥を訂正して下さい。なお、この Nimda に利用されるセキュリティ上の欠陥は Internet Explorer の『Java スクリプト』や『ActiveX』といった機能に含まれる物なのでその機能を切る事によって対策は可能です。まあ一部のページが見れなくなっちゃうんですけどね。

後、もう一つ私たちに関わりがある感染経路は『メールを見ると感染する』です。添付ファイルを開いたらでは無く、メールを見ると感染してしまうため、これも注意してなんとかなるものではありません。Outlook Express などの内部で Internet Explorer を使用しているメールソフトの脆弱性を利用するため、もっとマイナーなソフトを使えば防げます。

また、メールには 2 種類あって、文字だけの普通のメールとウェブページの形式を使用した html メールがあります。このウイルスは html メールでのみ作動するため、『html メールを受信しない』設定をＯＮにしていれば防げます。

教訓：

このウイルスは非常に多くの教訓を含んでいます。まず、『WindowsUpdate をこまめに行う』です。マイクロソフトもウイルスが出たらあらかたちゃんと対応してくれています。『WindowsUpdate』を行う事は脆弱性を利用するウイルスを防ぐために必須です。

次に『Java スクリプトや ActiveX をなるべく切っておく』です。まあ、これらを利用しているページも多いため、絶対にしろとは言えないのですが……。あとこれと同じで『html メールを受信しない』もＯＮにした方がいいです。また html メールを送らないのも礼儀だと思います。

後、やはり Internet Explorer を使わないという手があります。表示できないページもありますが、ウイルスの作者は感染を拡大させるべくメジャーなソフトを狙います。

Opera や Netscape などのウェブブラウザを使っていれば感染しないウイルスも結構あると思いますよ。

まとめ〜

ウイルスの感染経路に注目して書いてきました。ウイルスに恐怖心を持たれた方がいるかもしれません。ですが、私は『ちゃんと対策していればウイルスは容易に防げる』と考えています。添付ファイルなんか開かなければいいんです。Internet Explorer も使わなきゃいいんです (Windows を使わなければもっといいんですが)。WindowsUpdate も設定すれば自動でやってくれます。さらにウイルス対策ソフトをいれれば余程運が悪くない限り大丈夫でしょう。

とにかく知識さえあればウイルスは防げます。余裕です（たぶん)。肩の力を抜いて頑張ってください。

・・・・・7・・・・・

後書き

部長代理 60 回生 ぱんだ

ういーっす。部長代理で後書きを書くことになった佐々木です。今日は部誌を手に取って最後まで読んでいただいてありがとうございます。

毎年毎年専門的な内容で、一部以外の人を置き去りにすることで有名（？）な我が部の部誌ですが、今年はそれなりに読みやすい内容になっていたかと思います。
内容を専門的にすると読む人はいなく、内容を一般的にすると読めるようにはなるけど部員側としては満足がいかない。その辺のちょうどいい間を取ろうとか思ってたり思ってなかったり。

まぁとりあえずはこれでこの部誌も終わりです。読んでくれた方々にこの部誌が少しでも役に立ちますように。

後、新入部員の勧誘に少しでも役に立ちますように。

P.S.
締め切り当日に後書きとかもう無理とか言って自分に投げた部長は反省してください・・・
ごめん・・・(Jyakky)

灘校パソコン部 2006年度
クラブ活動報告書