



Theoretical Science Group

理論科学グループ

部報 197号
— 追い出しコンパ号 —

目 次

スキー合宿総紀	〔渡辺 尚貴〕	1
穴埋め原稿：スキーバス帰りのルート	〔Aleph-NULL〕	5
けえきばあちいの午後	〔かも〕	6
根津研へのいざない	〔Face〕	7
J a v a 事始め	〔わたる〕	10
連絡事項		23
ASCII 日本語 T _E X でのフォント増設法	〔Aleph-NULL〕	24

スキー合宿総紀

渡辺 尚貴

冬スキーから帰ってきて2カ月ちかいという今日になって、このスキーについて語ろうではないですか。語ろうで思い出したけれど、今回のスキー合宿は前回と比べて某氏がいなかつたために明らかに静かでしたね。

TSG部長を退役して、のんびりしようとしたら、小島さんから「なおくんお願ひね」ってこれだもの。旅行の経験がほとんどない僕は???な状態がしばらく続いて朝ご飯がない、夕ご飯もないって気づいたが時既に遅し。

とにかく集合さえしっかりできればあとはバスが運んでくれると思い、バス発車が10時半だというのに集合時刻を8時半にして布告。

伝統あるTSG時刻というのはいつの頃からか忘れされていたようでした。みんなには8時半に集合開始、9時に集合完了とちゃんと伝えておいたにもかかわらず8時半に80%集合完了してしまいました。僕は8時20分におりこうに到着したのになんとあの寺川さんが既に待っていられたのには驚きました。

集合場所には参加しないはずの経澤さんがおられました。「キモ」の行くえを尋ねに寺川さんをおっかけてきただけのようでした。

安田君が8時45分くらいに「やあ」と来たけれど実は彼がビリ。この衝撃の事実に彼しばらく驚嘆。時代はかわったのだよ。

駅の前は寒いので、とことこバス発着所にのんびり移動したら5分とただずして到着。しかしながらもなかった。遠くに明かりが見えた。それをたよると、旅行会社の受け付け本部の設営であった。受け付けの人曰く、あと1時間程おまちください。

寒いよ寒いよ寒いよ～。高架の柱の影に立つが、風が心を尽き抜けるように通り抜けていった。こんなところで1時間もなにしているだあ～。時間設定間違えたかなあ。みんな来るのが早すぎるよ。「渋谷いこ渋谷」寺川さん曰く。

ペチャくちゃ話で楽しむ間に、にわかに他の団体がどどっと集まってきた。僕はちやっかり受け付けの前にばつんと立っていたので、受け付け開始1番に手続き完了。クーポン券をたくさんもらうが何これ？ 竹村さんが今年も参加されたのでチェックしてもらいました。ありがとうございました。

指定された場所でバスの乗車を待つが、あまりにも手続き処理が1番乗りだったため、またも寒いところに長々と待たさる。経澤さんが「まだ～～～」と義理がたく見送り完遂するのを待たれていまいしたが力尽きて帰られました。バスに乗ったのはそれからまだまだあとのことでした。

バスは7、8団体の40名くらいの相乗りでしたが、TSGが一番大きくバスの後ろよりに位置していたため、バスの中は至って静かなものでした。

どうゆう経路を通ったのかわかりませんが、大岩君と崎山さんは隨時現在位置をチェック

クされていました。なんちゅう趣味や。僕は眠ろうとウォークマンを聞こうとしたのですが、音が聞こえない。コネクタが壊れていたのでした。眠れないよ～。狭いよ～。周りは眠っているよ～。なのにパークリアにつくとみんながぱっと起きて買い出しに出かけて、それでいて帰るとまたすぐやすやすやねむってしまうは理解できない。

眠れないまま。とうとう着いてしまいました。ふらふらだよ～。駒場祭で僕が一度も徹夜しなかったのは、一度でも徹夜すると次の日が動けなくなるから。それなのに今日はスキーですよ。大丈夫なのかしら。

宿は在りかはリフト乗り場のすぐ側と聞いていたので、ずっとむこうにリストが見えたのでそっちに向かってとここ歩いていきました。荷物の重さが応える。

宿に着いたが、部屋はありませんでした。地下の広間に荷物だけは置かせてくれるとのこと。幹事の僕は頭がふらふらしていたので変な質問でもしたのだろうか。受け付けのお兄さんにはげんな顔で見られました。いかん。東大生ってことは知られていないよね。恥。

朝食がめちゃくちゃたかい。焼きそば600円。

さて、スキー板とウェアを借りようかと、貸し出しの部屋にいたら、すでに団体がならんでいました。その後ろに我々がならんだのだが、券をもっている僕がちょっと用で部屋をでたら、別の人達に割り込まれてしまいました。女性たちの団体だったからかマスターの性格もあってか、割り込んだ団体が先に受け付けられてしまいました。これで10分の口ス。

いきなりトラブル続きの1日目の朝ですが、9時半ころにはみんなリフトの下に集まれました。

崎山さんと大岩君がなにやらつるんでたくらんでいる様子。気をつけないと。1年ぶりのスキーだったので先ずは徐行してブルークボーゲンして勘をとりもどしていました。うしろから崎山さんがいろいろ言うけれど今僕忙しいの。

我々の宿から直接行けるコースで初級コースは実は本の数百メートルしかない様子。コースが変で、ちょっとでもさらに登ると、もとの所に帰ってこれないという作りだったのです。

今年はスキー初心者がいなかったというのが良かったかもしれない。去年は僕は初めてで板がなかなかはまらなくって、はまると板が雪にはまってうごけなくなるような状態で竹村さんに非常にいろいろ御指導してもらったのです。なんどもなんどもころんで痛くて泣きそうだった所に吉田UMBさんに「辛いこともあるさ」と慰められるほどの惨めな思いをしたものです。今年はコースが難しかったから初心者がいたら助からなかっただろう。

崎山さんと大岩君の主導で我々は、ついに山のてっぺんにまで連れられてしまいました。ここから下りる道は中級者コース。帰れないよ～とわめく僕にリフトで下りれるとの助言。まあなんとか2、3回転んだ程度で無事下まで下りられました。これで少しは上達したかしら？

今年のスキー合宿での目標はパラレルターンができるようになることでした。そのため、まずは緩斜面を斜行ブルークでてれてれ滑る練習をし、それから山スキー板の力を抜けるようにして、斜行中はパラレルになるようにしました。安田君の助言でターンの前半には力を抜くことを覚えました。するとターン前半までパラレルが保たれるのでスキーが変わりました。でもターン後半はボーゲンにならないとひっくりかえってしまいました。

1日目も日が暮れ始め、さっさと宿に帰りました。やっと部屋のキーが渡されて、荷物を地下室から3階の部屋に運ぼうとするが、激しい筋肉痛。う~。老けたのう。部屋は12人で10畳が2部屋でした。意外と広かったですね。これでドラえもんにならずにすむ。

夕食を食べて、風呂も済ませると、疲れと昨晩眠っていなかったためにとて、崎山さんの演説をこもり歌のように聞きながら座布団の上で、眠ってしまいました。他の人もみんなそれぞれ椅子の上だの畳の上だのぐったりしていましたね。

毎年ここでは駄菓子を買って、飲み食いしながらいろいろ語りあっていたのですが、今年はその気力はまったくありませんでした。昨年は某氏がまったくもうってくらいに壊れていて、まとわりつかれて大変でしたというのに。

しばらくして起きて、時計をみると8時。もう夜も更けた。良い子はねましょ。人が集まっているほうの部屋に布団を展開しようとするが、多賀君と大岩君が既にねている。変な位置に布団をしくもんだからうまく展開できないじゃない。お部屋のフォーマット、不良セクタ発見、ポイ捨て除去。最適化、最適化、わっせ、わっせ。どうにか部屋に6つの布団をしきました。さっそく隅の布団に潜って、それからの記憶はなし。

朝になって、隣に寝ていた安田君が言うには、僕は激しく瞬間的にごろっと何度も移動したのでびっくりしたということでした。多賀君が崎山さんになにかされたそうですね。僕はなーにも知りません。そういうえば今年は某氏の他に、夜の2名の協奏曲もなかつたので本当に静かな夜でしたね。

2日目は、崎山さんの提案で昨日とは別のコースにいくことになりました。バスで15分くらいでしたか。いきなりジャイアントとかいうとんでもない急斜面でした。ここには下りリフトがあったので、僕は迷わずそれにのりました。

みんなで上方にまで行って、トレインになって下までおり、それからは各自好きずきでばらばらに滑りました。

僕は一人でまたのんびりパラレルの練習にいそしんでいました。片足だけで斜行できるようになり、だいぶスキーが軽くなりました。斜面の幅が狭くてちょっと練習には向いていませんでした。

午後はまた別なところを金子君と高野君とで滑ったのですが、滑れるようになり始めた高野君がよろこんで先を突っぱしって道を間違えてはぐれたこともありました。

僕のパラレルはこのころになると大体ターンのときでもパラレルのまま曲がれるようになりました。また斜滑降のときの2つの板で雪を削るようにするときのガリガリという音が、なかなかうれしかったです。

スキー合宿総紀

バスの関係上早めに引き上げなければならなかったのが残念でした。もっとすべりたかったのですが、でも次の一日があっても多分、筋肉痛で滑れなかつことでしょう。

宿に無事帰り着き急いで帰り支度をすませて我々は志賀をさりました。志賀はさつものの、どうもバスの動きが妙。北に向かっているぞ。どうやらいろいろなスキー場からの帰り客を乗せていくようです。

新宿東口バスターミナルに到着したのは朝（深夜）4時30分でした。簡単に挨拶をすませて我々は解散しました。

新宿ってこんな時間でも明るくってうるさいのですね。西武新宿線はまだ始まっていませんでした。駅の入り口がシャッターで閉まっているのでした。こんなのを見るのは初めてでした。15分程まってようやくシャッターが開き、各駅停車本川越行きにのりました。ここからわが街まで24駅です。いつのまにか寝っていたようでした。帰りのバスでもやはり眼れずにいたので、ゆったりした電車の中で安心したのでしょう。20駅をいつのまにか通り越していました。

わが街の駅をおりてもまだ真っ暗でした。田舎なので本当に真っ暗で足元が見えなくて怖いです。そらにはキラキラ夏の大三角形がみえました。もう夏が近いのですね。

空を見上げながら足をつまづきながら僕は自宅に着き、なぜか既に敷いてあった布団にもぐりこんでぐっすり眠ったのでした。

今年も楽しいスキー合宿だったではないですか。2日間しか滑れなかつたのが残念かな。来年は3日間にしよう。それからスキーのうまい人はカメラを持ってくるように。ビデオカメラでもいいですね。滑りながらでも撮れるでしょ。合宿の思い出が一人一人の心の中にしか残らないのが残念だから。

最後に今回のスキー合宿に参加された勇志たちを紹介します。

1年生	金子 濟	高野 直樹	大岩 寛	木原 英夫
2年生	渡辺 尚貴	安田 知弘	西沢 信行	多賀奈由太
3年生	寺川 愛印	竹村 太郎		
M1年生	崎山 伸夫	三瓶 均		

来年も楽しいスキー合宿にしましょう。

穴埋め原稿：スキーバス帰りのルート

Aleph-NULL

暇な記事ですねえ（＾＾；。

とりあえず帰りのバスで「部報に載せる」と宣言してしまいましたので…。さすがにうっとおしいので字も小さくしてみました。（笑）

読む気のある暇な方、道路地図をご用意ください。

ではスタート。

とりあえず熊の湯を出発。しかし行きは8時間で着いたのに帰りの旅程はなんと9時間半もある。なんじゃこりや？

【熊の湯】^{5k R292} 【蓮池】^{16k R292} 【夜間瀬】

ここまで普通のルートですねえ。でもなぜかチェーンを脱がない。ここで変だとおもうべきだった。

【夜間瀬】^{4k R403} 【分岐点】^{4k} 【北志賀竜王スキー場】

なんと、夜間瀬からバスは逆に曲がり、山をめざして一直線。がーん。

チェーンを付けたまま細い山道をぬけてきたら、北志賀に着いてしまった。すごい遠回り。北志賀って完全に尾根筋一本違うんだけどなあ。

【北志賀竜王スキー場】^{9k} 【北志賀小丸山スキー場】^{9k R403} 【夜間瀬】

やっと本道に戻った。いくら安いスキーバスだからって2時間も山道をチェーンを履いてがたがたと走られたんじゃねえ。疲れた。

【夜間瀬】^{R292,R117} 【浅野】^{R18 アップルライン} 【上信ドライブイン】

志賀スキーといえばこのドライブイン。チェーンがたがたの腰の疲れをいやして、またバスへ。次の興味はどこで高速へ乗るか。幸い最近は長野まで長野自動車道や上信越自動車道が通っているのですが…。

【上信ドライブイン】^{R18} 【長野】^{R18バイパス} 【更埴】^{R18} 【上田-西】^{R18 上田バイパス} 【上田-東】

ううむ。普通の地図帳でもはっきりとわかるような距離を延々バスは国道を走ります。そろそろみんな寝るころです。が、私は昔から車の中では寝られないのです（T_T）。ところでこの辺、本道は上田市内を突っ切っているので避けてバイパスを走るのですが、このバイパス、困ったことに上田の東側でとぎれついて国道18号に戻れないのです（＾＾；普通はここでバイパス同様に整備された県道を走って早めに右折して国道に戻るのですが…。

【上田-東】^{県道} 【バイパス終点】^{市道・町道} 【交差点】^{県道} 【交差点】^{R18} 【軽井沢ドライブイン】

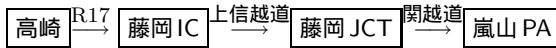
曲がらない。どんどん道も細くなってくる。どこ行く気だ？結局混雑を避けたのか名もない道を10k近く直進してました。もう一台のスキーバス以外に車はいなかった。TSGer達の「帰りは釜飯を食うぞ¹⁾」の願い空しく、バスは軽井沢手前で2度目の休憩。横川の釜飯にはとうとうありつけませんでしたとさ。折角通過したのに。

【軽井沢ドライブイン】^{R18} 【中軽井沢-西】^{碓井バイパス} 【横川】^{R18} 【高崎】

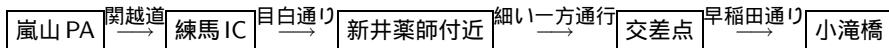
1)行きは真夜中で売っていなかった

けえきぱあちいの午後

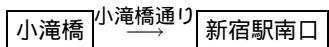
夜の碓井峠を抜け、バスはいつまでも一般道をひたはしります。とうとうこのまま東京まで一般道かと思いました。どうせ高速乗るならもっと楽に碓井峠越せるのにねえ。



バスはパーキングエリアに着いたのですが、運転手も添乗員もだまつたまま。どうやら時間調整みたい。30分くらいしてから、おもむろに電灯付けて「はい、ここで15分の休憩します」だって。まさか寝ずに一部始終を見てる人がいるとは思っていないだろうなあ(^^;。



すごい細い道を抜けてましたけど毎晩あんなところバスに通られたら周辺住民は迷惑なことでしょう。



なお氏の報告通り 4:30 に新宿到着。お疲れ様。(読み切った奇特な方もお疲れ様です。)

けえきぱあちいの午後

かも

去る1/14に、TSG最凶^H強の行事と呼ばれるケーキパーティーが寮食堂にて行われました。各人限界量のケーキを持ち寄って、それを心ゆくまで食べるという話に甘いものが苦手な僕は震え上がり、決して行くまいと思ったのですが、怖いもの見たさというやつでしょうか、それとも魔が差したというべきでしょうか、とにかく僕は桜上水のケーキ屋で9個の小さなケーキを買い、そしてセンター試験まっただ中の駒場キャンパスにふらふらと足を踏み入れてしまったのでした。

さて実際食べ始めてみると、目の前に山のようにそびえるケーキにびびりながらも、序盤は案外食事が進み、2・3個程度は楽に入りました。しかし5個目を過ぎたあたりからは、食べても食べてもなかなかならないケーキを見るにつれ、次第にケーキが喉を通りづらくなり、後からきた人のケーキの追加やK.O.さんのタッパーいっぱいの自家製ゼリーなどによる精神的ショックも手伝って遂に限界を迎え、7個目を片づけたところからは記憶の糸がぱつりと切れています(嘘)。

そのあとも今西さんのミントゼリーが飛び出したりしたようですが、結局持ち寄ったケーキもすべてなくなり、平穀無事に終わりを迎えたようです。ちなみに僕は7個という平凡な成績に終わりました。もうちょっと食べられるかと思ったのですが…

根津研へのいざない

Face

こんにちは。根津研会計及び根津研責任者の Face です。皆さんは根津研についてどの位知っていますか？すでに根津研に来たことがある人もいるでしょうし、逆に根津研という言葉をはじめて耳にする人もいるでしょう。というわけで、根津研とはどういう所なのかを簡単に説明したいと思います。

根津研というのは略称で、正式には「根津中央研究所」といいます。ここには98MATE (Ap2)とATマシンがあります。MO ドライブ、CD - ROMもついています。マシンのスペック、環境などの詳しい説明は省略します。そして、ゲーム機はプレイステーションにセガサターン、スーパーファミにメガドライブにネオジオと、一通りそろっています。ビデオデッキもあります。でもここまで、あまり305と変わりありませんね。

根津研が305と一番違うのは、根津研というのはアパートの部屋であるという事です。従って、便所は当然として、流し台に冷蔵庫、ガスコンロに炊飯器といった生活に必要なものも揃っています。305では冬は寒い中下からの冷たい風をこらえて椅子に座り、夏は驚異的に暑苦しい中ぎゅうぎゅう詰めにならざるを得ないわけですが、根津研では、冬はホットカーペットとこたつで夏は冷房、と空調も完璧です。さらに、根津研には布団や毛布も用意してあるので、徹っぷよ徹アニメ徹サッカーといったゴージャスなミッドナイトをエンジョイすることも出来ます。

漫画が好きな人もそうでない人も、最初に根津研に来たら、その莫大な量の漫画に驚くことでしょう。何冊位あるのかは、私にも想像がつきません。とにかく、本棚の置けるところにはすべて本棚が置いてあるという感じです。これはもう言葉では表せません。是非とも一度、見に来てください。

根津研の会員になるには、TSG のメンバーである皆さんは、

「年35,000円の会費を根津研に払う。」

これだけで、根津研の鍵がもらえて、晴れて根津研の会員になります。いきなり年35,000円払えと言われるとびっくりするかも知れませんが、

- TSG では1年2年では部費を要求されない。
- 35,000円なんて日給5,000円のバイトを1週間もすれば何とかなる。
- 外泊することを考えたら、1ヶ月に1度泊まればそれだけで元が取れる。
- 分割払いもOK。

ということを考えれば、1年で35,000円というのはそれほど大変ではないはずです。バイトを探すのが面倒で、夏休みが暇で、女の子の知り合いが欲しいという方がいましたら、日給5,000円ではありますか、結構楽しいバイトがあるので、相談に応じます。

とりあえずは2年生は

「年10,000円払えば鍵はもらえる。」

ということになります。ただ、今後の健全な根津研の運営を考えると、最終的には（年度の終わりまでには）35,000円を払っていただけると嬉しいです。また、この金額は1年間で必要な運営経費を人数で割ったものです。従って、学科の3年生をTSGに勧誘して根津研の会員になってもらえば、それだけ一人当たりの負担は軽くなります。どんどん根津研を宣伝しましょう。

実は、去年の今ごろは、「根津研がなくなる」という危機に直面していました。本来根津研とは、3年になって進学先がバラバラになってもコミュニケーションを図れる場所がほしいという発想から生まれた物でした。しかしここ2年間で、いぬ。BBSの普及や進学先がかたよるといった出来事が起り、根津研の会員が激減しました。そして根津研の存在価値が問われ、本当にあと一步で根津研が壊滅するところまで話が具体化していました。結局うちの学年の強い要請とOB・OGの方々の援助により、'95年度も根津研が存在することができました。

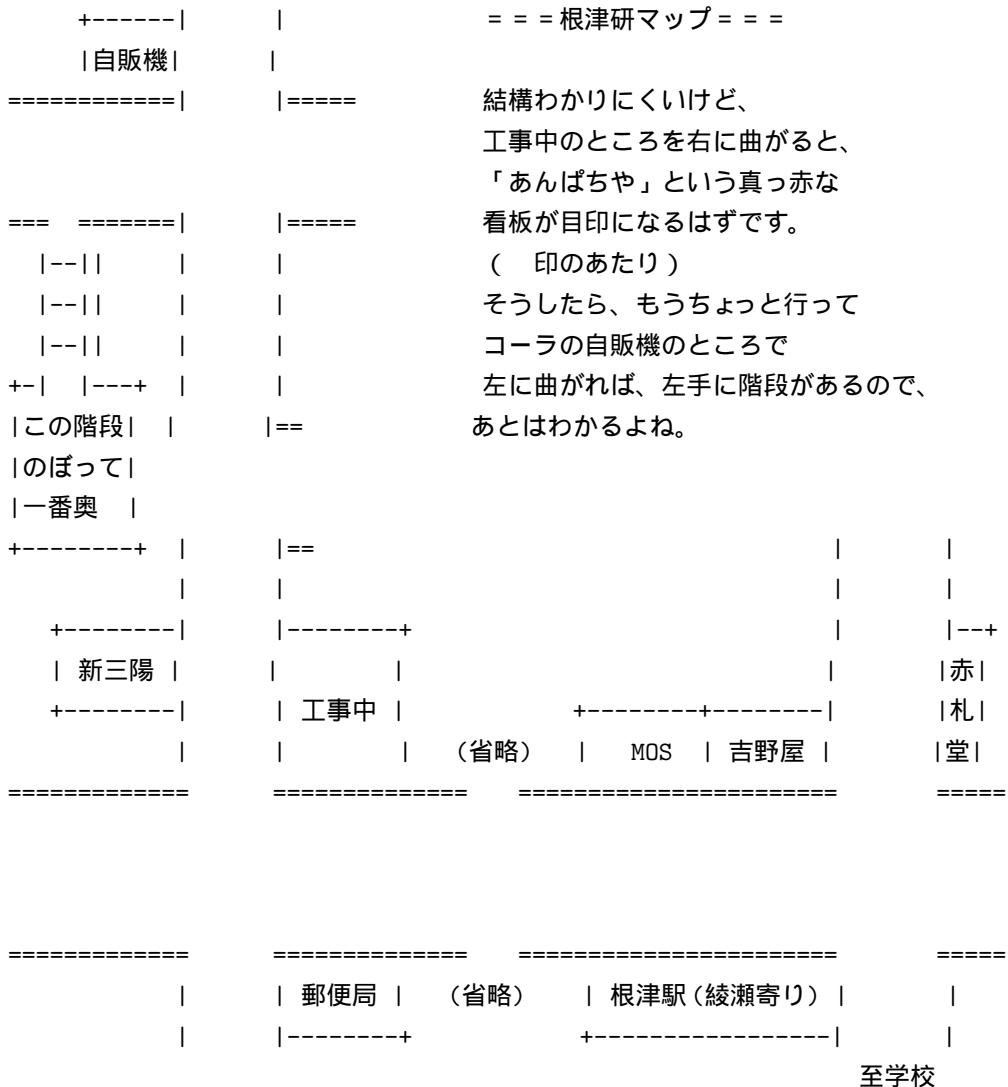
しかし、相変わらず根津研の運営は円滑とはいえない。根津研は1学年あいだが空いてしまうと、最低2年間、特にTSGの場合院への進学率が高いので、4年間はその影響を受けることになります。というわけで、そういうピンチの状態にも関わらず存続の道を選んだ我々3年生は2年生、そして1年生の人達にもっと積極的に根津研に接して欲しいと願っています。

2年生の皆さんには、まあまずは大量のこぼれた酒を吸ったあの緑色の絨毯¹⁾に腹ばいになつて漫画を読んで、ゲームをしてください。そのうち、ものがごちゃごちゃ置いてある狭いアパートが、生活に欠かせない重要な空間と思えることでしょう。根津研は本当にいいところです。みなさん、はやく根津研になじんで、根津研を好きになりましょう。

駒場生が根津研に行く場合は、電話をかけてみて誰かが出たら、これから行くということを伝えておきましょう。行ってみて誰もいなくて鍵がかかっているというのはつまらないですから。誰もでなかつたら残念だけど諦めるしかないですね。

根津中央研究所：(03) 3821 – 3384

¹⁾緑色の絨毯は、春の大掃除の際にねずみ色の新しい絨毯に敷きかえる予定です。



J a v a 事始め

わたる

最近 Java , HotJava という言葉をよく聞きます。珍しいもの好きの私はさっそく飛びついで Java の勉強を始めました。そこで Java を全く知らない状態から , ホームページにアニメーションを実現できるようになるまでを簡単に解説することにします。ただし初步的な C++ の知識があることは前提としますので御容赦下さい。

1 J a v a とは ?

「 Java とは , 異機種間ネットワークによる分散環境下でのアプリケーション開発支援を目的に , Sun Microsystems で開発された言語である。」 (C マガ 1 月号)

ようするに Internet 上で中間コードをやりとりし , 各 OS ごとに用意されたインタプリタに実行させるということです。パフォーマンスは公称で C の 20 倍です。現時点ではホームページ上に動画や音声 , ゲームなどを実現することができるもとして注目を集めています。

『 Java 』という名前の由来はジャワコーヒーです。FAQ によりますと ,

We were aiming to come up with a name that evoked the essence of the technology — liveliness, animation, speed, interactivity, and more. “Java” was chosen from among many, many suggestions. The name is not an acronym, but rather a reminder of that hot, aromatic stuff that many programmers like to drink lots of.

ちなみに発音はジャバです。開発者の方々がそう呼んでいるそうです。
それから HotJava とは Sun が Java で作った Java 対応の WWW ブラウザのことなので
すが , バージョンアップがもう随分と滞っているそうなので , 某ネスケに降伏したのか
もしれません。 (^ ^ ;)

2 言語としての特徴

Java のソースは一見すると C++ にそっくりなのですが、細部はずいぶん異なっています。以下にその概略を紹介しましょう。

- 関数はない。(メソッドに一本化。)
- struct , union はない。(クラスがあるので不要。)
- ポインタはない。
- String が用意されている。(実体は char の配列。)
- 可変数引数はない。
- goto はない。(予約語にはある。謎。)
- 多重継承はない。(interface で代換え。後述。)
- マルチスレッド対応。
- プリプロセッサがない。
- ライブラリはダイナミックに呼び出す。
- インタプリタ方式。
- ガーベッジ・コレクタ装備。(メモリ漏洩がない。)

3 用意するもの

Java の開発には JDK (Java Developer's Kit) が必要です。Sun は WIN32 用と Solaris 用を無料で配布しています。Mac 用も最近登場したらしいです。(<ftp://ftp.glocom.ac.jp/mirror/java.sun.com/>) また JDK のソースは公開されていますので、Linux , FreeBSD , IRIX などに移植されています。

4 hello,world

お約束の hello,world のプログラムを作ってみましょう。ファイルネームを `hello.java` として以下のように入力してください。

```
1: /**
2: *  hello.java --- Application example
3: */
4:
5: class HelloWorld {
6:     public static void main(String args[]){
7:         System.out.println("hello,world");
8:     }
9: }
```

C++ にとても似ていますね。次に `javac` でコンパイルします。（`javac` 自体が Java で書かれているのでちょっと遅いです。ネイティブなコンパイラの登場が待たれます。）

```
C:\Java\>javac hello.java
```

これでカレントディレクトリ上に `HelloWorld.class` というバイナリファイルが生成されました。`javac` は吐き出すファイルの名前をソースファイルからではなく、ソース内で定義されているクラスの名称で決めます。もしソース中に二つ以上のクラスがあれば、その数だけ新しいファイルが作られるのです。では実行してみましょう。

```
C:\Java\>java HelloWorld.class
```

無事に `hello,world` と表示されたでしょうか？次にソースの説明をします。

まず 1~3 行目がコメントになります。// と /* */ は全く C++ と同じです。

5 行目で `HelloWorld` クラスを宣言しています。Java のプログラミングとはクラスを定義することに他なりません。

6 行目で `main()` メソッドを宣言しています。Java インタプリタは `class` ファイルを実行する時にまず `main()` メソッドを呼び出します。`static` はこのメソッドがクラスメソッドであることを定義するものです。`static` なメソッドや変数は通常のインスタンスマソッドやインスタンス変数とは異なり、あるクラスのインスタンス全体で共有されます。`main()` の引数 `String args[]` にはコマンドラインの情報が文字列の配列（説明は後ほど）として渡されます。

7 行目で標準出力に `hello,world` を出力しています。`System` は `java.lang` パッケージ（クラスライブラリのこと）で定義されていて、全ての変数とメソッドが `static` なクラ

スです。java.lang にあるものはプログラムのどこからでも自由に使用することができ、C の標準ライブラリに相当します。ライブラリの使い方は後ほど説明します。

4.1 ちょっと道草

HelloWorld.class のサイズは 466 バイトです。とても小さいのはライブラリの動的リンクのおかげでしょう。また 16 進ダンプをしてみると面白いところがわかります。

```
3A56:0000 CA FE BA BE 00 03 00 2D-00 20 08 00 1F 07 00 17
```

先頭の 4 バイトが “CAFE BABE” になっていますね。

5 2つのプログラム：アプリケーションとアプレット

実は Java のプログラムにはアプリケーションとアプレットの 2 種類があります。アプリケーションは普通のプログラムで、C 言語の代用になるほどではありませんが、ネットワーク系を含めてたいていのことを実現できます。ただしアプリケーションそのものを Internet で送り実行させることはできません。例の hello.java はアプリケーションです。一方アプレットはブラウザ上で実行するためのプログラムで、ネットワーク上を渡ることができます。セキュリティ上の問題からアプレットの機能には厳しい制限があります。例えばブラウザ側のファイルシステムには読み書きできませんし、他のプログラムを起動させることもできません。

6 アプレットプログラミング

それではアプレットの例をいきなり見てもらいましょう。クリックした場所にマウスのボタンを押した回数が表示されます。

```
1: /**
2:  * Sample1.java --- Applet example
3:  */
4:
```

```
5: //使用するクラスライブラリを指定する。
6: import java.awt.*; //Graphics クラスなどWindow Tool Kitを提供。

7: import java.applet.*; //Applet クラスの定義を得る。
8: //全てのアプレットはApplet クラスのサブクラスである。
9: public class Sample1 extends Applet {
10:    //インスタンス変数
11:    int x,y,counter;
12:    //アプレットの初期化時に実行される。
13:    public void init(){
14:        x = 0; y = 0; counter = 0;
15:    }
16:    //アプレットの再描画時に実行される。
17:    public void paint(Graphics g){
18:        g.drawString("counter = " + counter , x, y);
19:    }
20:    //マウスのボタンが押された時に実行される。
21:    public boolean mouseDown(Event e, int X, int Y){
22:        x = X; y = Y;
23:        counter++;
24:        repaint(); //アプレットの再描画を要請する。
25:        return(true);
26:    }
27: }
```

6, 7行目の `import` は使用するライブラリを指定しています。アプレットの場合、ブラウザが用意したライブラリを実行時に読み込むことになります。Java の API は階層構造のライブラリとして整理されており、`java.awt.*` は `java.awt` 以下の全層を表しています。

9行目の `Sample1 extends Applet` は `Sample1` クラスが `Applet` クラス(`java.applet` にて定義)を継承するという意味です。アプレットを作る上でこの継承は必須であり、`Applet` クラスのメソッドをオーバーライドすることによって作成したいプログラムを形成するのです。

11行目でインスタンス変数を宣言しています。変数のアクセス指定は C++ とほぼ同じで、デフォルトでは該当パッケージ内に対して公開となります。変数の精度は C と少し異なっています。また初期値は決まっているのですが、自分で初期化するよう推奨されています。

型名	精度 (bit)	符号	初期値	
boolean	1	なし	false	真偽値 (true/false)
byte	8	あり	0	1バイト整数
char	16	なし	\u0000	文字 (Unicode対応)
short	16	あり	0	2バイト整数
int	32	あり	0	4バイト整数
long	64	あり	0	8バイト整数
float	32	あり	+0	単精度浮動小数点
double	64	あり	+0	倍精度浮動小数点

13行目の `init()` はアプレットの初期化時にブラウザによって呼び出されます。`init()` の定義がない場合は、親の `Applet` クラスにある何もしない `init()` が代わりに実行されます。

17行目の `paint()` にアプレットの描画内容を記述します。`paint()` はアプレットの始動時とウインドウの切り替えなどで画面が破損した時に呼び出されます。引数には `Graphics` オブジェクトがあり、デバイスコンテクストな情報を渡すためだと思うのですが、違うかもしれません。(^^;

18行目で `Graphics` オブジェクトの `drawString()` メソッドを呼び出しています。(メソッドの名前は小文字で始めるのが慣習です。) 名前の通り文字列を描画するもので `String` オブジェクトを引数に取ります。`String` オブジェクトは演算子 + によって連結することができ、`String` 以外のオブジェクトが+された場合は自動的に変換されます。

21行目の `mouseDown()` は `java.awt.Component` クラスのメソッドです。マウスボタンが押された時に呼び出されます。押された時の情報は引数の `Event` オブジェクトに保存されています。

24行目の `repaint()` は `java.awt.Component` クラスのメソッドで、アプレットから画面の再描画をブラウザに要求するために使います。ブラウザは再描画時に `update()` を呼び出します。`update()` はオバーライドされていない場合、`paint()` を実行するだけです。`update()` を自分で用意し、画面の描画効率を上げることができます。

以上でソースの解説は終わりです。アプレットプログラムはイベント駆動であることがわかつてもらえたでしょう。

7 アプレットとH T M L

例のアプレットをホームページに張り付けるには以下の様にします。

```
1: <applet code=Sample1.class width=300 height=300>
2: </applet>
```

これは必要最小限の設定しかしていません。URLを指定したい場合は codebase=URL を追加します。もちろん align や alt の設定もできます。アプレットにパラメータを与えることもできます。<applet></applet> の間に <param name=パラメータ名 value=値> を入れて下さい。実用例は次で示します。<applet></applet> の間に他の HTML は、アプレットがない場合の代替 HTML として解釈されます。

8 いよいよアニメーション

今度もまずソースを見て下さい。

```
1: /**
2:  ** anim.java --- 汎用アニメーションアプレット
3:  **
4:  ** Copyright 1996 Wataru Kaneko (TSG)
5:  **
6:  */
7:
8: import java.awt.*;           //For Image
9: import java.applet.*;
10:
11: public class Anim extends Applet implements Runnable{
12:     Thread kicker = null;    //同期をとるためのスレッド
13:     Image image[];          //アニメーション格納用
14:     int current_frame = 0;   //現在のフレーム番号
15:     int frames;              //全フレーム数
16:     int pause;                //フレームの表示間隔 (ms)
17:     int width = 0, height = 0; //フレームの大きさ
18:     boolean loaded = false;  //アニメーションの読み込みを完了したか
19:
20:     public void init(){
21:         String buf;
```

```
23: // アニメーションの名前を取得
24: String name = getParameter("name");
25: if(name == null) name = "";
26:
27: // フレーム数を取得
28: buf = getParameter("frames");
29: if(buf == null){
30:     System.err.println("Need param 'frames'.");
31:     System.exit(-1);
32: }
33: frames = new Integer(buf).intValue();
34:
35: // アニメーションを格納
36: image = new Image[frames];
37: for(int i=0; i<frames; i++)
38:     image[i] = getImage(getDocumentBase(), name + (i+1)
39:                         + ".gif");
40:
41: // アプレットの大きさをイメージに合わせる。
42: for(int i=0; i<frames; i++){
43:     int temp;
44:     while((temp = image[i].getWidth(this)) < 0) {}
45:     if(temp > width) width = temp;
46:     while((temp = image[i].getHeight(this)) < 0) {}
47:     if(temp > height) height = temp;
48: }
49: resize(width, height);
50: loaded = true;
51:
52: // フレームの表示間隔を取得 (defalt 1 sec)
53: buf = getParameter("pause");
54: if(buf == null) buf = "1000";
55: pause = new Integer(buf).intValue();
56:
57: public void start(){
58:     // 同期用スレッド
```

```
59:     if(kicker == null){
60:         kicker = new Thread(this);
61:         kicker.start(); //スレッド始動
62:     }
63: }
64:
65: public void stop(){
66:     if(kicker != null){
67:         kicker.stop(); //スレッド停止
68:         kicker = null;
69:     }
70: }
71:
72: /**
73: * kicker Thread本体
74: */
75: public void run(){
76:     while(true){
77:         try{
78:             Thread.sleep(pause);
79:         }
80:         catch(Exception e){
81:             System.err.println("Exception in thread 'kicker'.");
82:             System.exit(-1);
83:         }
84:         // pause ミリ秒ごとにpaint()を呼び出す。
85:         repaint();
86:         current_frame++;
87:         if(current_frame == 15) current_frame = 0;
88:     }
89: }
90:
95: }
```

次はこのアプレットを HTML に組み込んだ例です。

```
1: <applet code=Anim.class width=100 height=100>
2: <param name=name value=sample>
```

```
3: <param name=frames value=15>
4: <param name=pause value=700>
5: </applet>
```

sample1.gif , sample2.gif , . . . , sample15.gif を 700 ミリ秒の間隔で表示します。これだけ見れば何をしているのか理解できてしまう人もいるかもしれません。しかし Java プログラミングならではの細工がいろいろ施されていますので、ソースを始めから全部解説しましょう。

```
11: public class Anim extends Applet implements Runnable{
```

extends Applet は前に説明したとおり Anim クラスが Applet クラスから派生するという意味です。次の implements Runnable は、java.lang.Thread で定義された interface の Runnable を使うという意味です。Anim クラスのマルチスレッド化が可能になります。interface とはメソッドや定数の定義をまとめたもので、これを使うとクラスの継承による階層的な定義とは別に、共通の機能を持ったクラスを作つて分類することができるのです。この interface の概念はマイナーな Objective-C から採用されたものだそうです。Java は C++ のように多重継承はできませんが、この interface で補っています。

```
12:     Thread kicker = null;      //同期をとるためのスレッド
```

このアプレットではアニメーションの同期を取るためにスレッドを使っています。まず 12 行目で Thread 型の kicker を宣言していますが、ここでは Thread オブジェクトは生成されません。

```
60:         kicker = new Thread(this);
```

ここで生成されます。Java は new により改めてメモリを確保するまでクラスオブジェクトを生成しません。ようするに kicker は C++ で言うところのポインタなのです。Thread のコンストラクタはマルチスレッド化したいクラスオブジェクトを引数に要求するので、Anim クラス自身を表す this を与えています。スレッドの話は少し先に延ばして、メモリ関連のことをもう少し説明することにします。

```
13:     Image image[];           //アニメーション格納用
```

ここでは Image クラス型の配列を宣言しています。Java では配列用のメモリも実行時に動的に確保されます。

```
36:     image = new Image[frames];
```

実のところ、配列は配列型というクラスオブジェクトなのです。配列の生成時には `length` インスタンス変数が自動的に設定され、この例では `image.length=frames` となります。これにより C と違い、配列の添え字が範囲内であるのかが自動的にチェックされます。なお添え字が 0 から始まるのは C と同様です。

次にアプレットの初期化を行う `init()` の中身を見ていきましょう。

```
24:     String name = getParameter("name");
```

`String` 型の `name` を生成し HTML の以下の部分からパラメータ "sample" を受け取っています。

```
2: <param name=name value=sample>
```

`getParameter()` はパラメータの取得に失敗した場合 "" ではなく `null` を返しますので注意が必要です。

```
33:     frames = new Integer(buf).intValue();
```

ここは Java に関数がないことが弱点として現れたところだと言えるでしょう。`String` 型を `int` 型に直接変換する関数がないため、`Integer` なるクラスオブジェクトを使って解決しています。（`String` クラスに `int` へ変換するメソッドがあればすむはずですが、クラス定義が複雑になるのを嫌がったようです。）

```
38:     image[i] = getImage(getDocumentBase(), name + (i+1)
+ ".gif");
```

`image` に画像を格納しています。`getImage()` の第一引数には URL を、第二引数には画像ファイルを指定します。Java の API は GIF と JPEG をサポートしています。ところで画像の読み込みには、少なくともコンピュータにとっては長い時間がかかります。そこで `getImage()` は読み込みが完了する前に処理を返すようになっていますが、これが問題を引き起こしてくれます。

```
43:     while((temp = image[i].getWidth(this)) < 0) {}
```

ここで一見妙なことをしています。`getWidth()` は御想像の通り画像の幅を返すメソッドですが、値が `-1` になる場合があるのです。それは `image[i]` の画像がまだ完全に読み込まれていない時です。アプレットはこのような状況にも対処できるように作らねばなりません。

```
48:     resize(width, height);
```

`resize()` はウインドウ上でアプレットの占める大きさを変更します。

```
1: <applet code=Anim.class width=100 height=100>
```

ここでの設定はアプレット側の都合で変えてしまえるわけです。

```
57: public void start(){
58:     // 同期用スレッド
59:     if(kicker == null){
60:         kicker = new Thread(this);
61:         kicker.start(); //スレッド始動
62:     }
63: }
65: public void stop(){
66:     if(kicker != null){
67:         kicker.stop(); //スレッド停止
68:         kicker = null;
69:     }
70: }
```

`start()` と `stop()` はふつう対にして用います。それぞれアプレットが動き始める時と止められる時に呼び出されます。`start()` は `init()` の次に必ず実行されます。ブラウザが他のページに移動する時 `stop()` は実行されます。このアニメーションアプレットは `start()` で新しいスレッド `kicker` を作っているので、`stop()` ではそれを壊すようにしています。68行目のように `kicker` に `null` を代入してしまえば、メモリ上のゴミとなった `Thread` オブジェクトはガーベッジ・コレクタが自動的に回収してくれます。

```
75: public void run(){
76:     while(true){
77:         try{
78:             Thread.sleep(pause);
79:         }
80:         catch(Exception e){
81:             System.err.println("Exception in thread 'kicker'.");
82:             System.exit(-1);
83:         }
84:         // pause ミリ秒ごとに paint() を呼び出す。
85:         repaint();
86:         current_frame++;
87:         if(current_frame == 15) current_frame = 0;
88:     }
```

89: }

run() が kicker スレッドの本体です。kicker.start() とすると run() が別プロセスとして実行され始めるのです。kicker は一定時間ごとに repaint() を呼び出して、アプレットに次のフレームを表示するよう合図を送っています。77行目からの例外処理は文法的には C++ と同じですが、必ず必要という点が異なります。

```
91:     public void paint(Graphics g){  
92:         if(loaded)  
93:             g.drawImage(image[current_frame], 0, 0, this);  
94:     }
```

さて、最後に paint() です。読み込みが完了する前に表示しようとしても、何も画面には出ませんので御注意を。

9 終わり

Java はなかなか愉快な言語であるとは思いませんか？その性能もさることながら、Sun の情報公開性と無料で使わせてくれると言うところに私は魅せられてしまいました。今後 Java は WWW の Emacs-Lisp 的な地位を築いていくと思います。

ところで drawImage() が透明色付き GIF 対応していることがわかりました。これはスプライトに使えるでしょう。誰か平安京エイリアンのアプレットを作りませんか？噂ではミサイルコマンダーのアプレットが存在するらしいので、きっと可能でしょう。え、自分でやれって？ (^^;;

参考文献など

- C Magazine 1月号～3月号 短期連載 “Internet 時代を築く新言語 Java”
- The Java Language Tutorial! [<http://java.Sun.COM/tutorial/>]
- Java Application Programming Interface User's Guide
- The Java™ Language Environment: A White Paper
- Java FAQ 日本語版
[<http://www.webcity.co.jp/info/andoh/java/javafaq.html>]

- NIFTY-Serve 人工知能フォーラム (FAI) Lisp/Prolog/Smalltalk などの言語と実践
(4) 会議室

連絡事項

- 次号はオリパンフ号の予定です。原稿の割り振りは 2/22 の会合で話した通りです。
締切 3/18 12:00 必着。オリパンフに関しては原稿遅延・ブッチは巻末に載せて
公開します。
- 305 に新マシンが導入されました。NEC の PC-9821Xa/C10W です。Memory は
23.6MB に増設されています。

[利用法] A: には MS-DOS 6.2, djgpp, ASCII 日本語 TeX が導入されています。B:
は各自のホームディレクトリです。C: は Windows95 が導入されています。リセットで A:
から立ち上がる所以、初めての人は newuser でログインして登録するか、
guest でログインしてください。win コマンドで Windows95 に切り替わります。
ゲーム禁止です。その他は DOS 関連: Aleph-NULL, djgpp & Win95: わたる に
聞いてください。

Epson 486GR は従来どおりです。

ASCII 日本語 \TeX でのフォント増設法

Aleph-NULL

1 はじめに

この文章では、ASCII 版の日本語 \TeX において、標準以外のフォントを使用する方法について触れます。なお、 $\text{\LaTeX}2\varepsilon$ については触れていません。¹⁾

2 単独のフォントの場合

2.1 英文フォントを加える

例えば、Concrete Roman 10pt を文書内で使いたいとします。これが特定箇所でしか使われていないのであれば話は比較的簡単です。

まず、フォントをコントロールシーケンスに割り当てます。Concrete Roman 10pt なら、tfm ファイル名は ccr10.tfm ですので、例えば

```
\font\tenccr=ccr10
```

とします。これで、文書中で `\it` などと同じように `\tenccr` と用いることで、Concrete Roman 10pt を用いることができます。

入力：

```
\font\tenccr=ccr10
This is Computer Modern Roman 10pt.\par
{\tenccr This is Concrete Roman 10pt.}
```

出力：

```
This is Computer Modern Roman 10pt.
This is Concrete Roman 10pt.
```

¹⁾早く公式な日本語版が出るといいのになあ。

また、 \LaTeX 流には `\newfont` 命令があり、この例では

```
\newfont{\tenccr}{ccr10}
```

と指定することになります。どちらを使うかは好みでいいでしょう。

文法は以下の通りです。

font-define-sequence:

```
\font\control-sequence=font-definition  
\newfont{control-sequence}{font-definition}
```

font-definition:

```
tfm-name font-modify-postfix
```

font-modify-postfix:

```
at size  
scaled scale-factor  
scaled \magstephalf  
scaled \magstepmag
```

font-modify-postfix で、*at size* を用いた場合、フォントの tfm に書かれたサイズと比較し、適当な倍率で拡大縮小が行われます。*size* には \TeX で有効な長さを指定します（例：30pt, 5mm, 4zw, 13Q [P \TeX]）。

scaled scale-factor では、*scalefactor*/1000 倍に拡大されます。例えば、*scaled 1200* では 1.2 倍に、*scaled 850* では 0.85 倍に拡大されます。

また、*\magstepmag* では、 1.2^{mag} 倍に、*\magstephalf* では $\sqrt{1.2}$ 倍に拡大されます。これは、1.2 倍の系列がフォントの組み合わせとして適しているため、このような指定を受け入れるようになっています。実際、 \TeX の 14pt (`\Large`) は 14.4pt でデザインされているのです。

次の例は有効な宣言です。

```
\font\tenccr=ccr10          % Concrete Roman 10pt  
\font\twtyccr=ccr10 scaled \magstep4  
                          % Concrete Roman 10pt の 2.0736 倍  
\font\bigtsg=tsglogo at 160pt  
                          % TSG logo 10pt を 16 pt に 16 倍  
\font\tenccrb=ccr8 at 10pt  
                          % 8pt を 10 pt に 1.25 倍  
\font\tenccrc=ccr9 at 10pt  
                          % 9pt を 10 pt に 1.111 倍  
\newfont{\egtccsc}{ccsc10 scaled 800}%  
                          % Concrete SmallCaps 10pt の 0.8 倍
```

\TeX では改行は段落の指示を除いては空白とみなされるので、文章中に指示をおいた

```
Suddenly, you notice a cave spider  
\font\tenccr=ccr10  
at the top of ...
```

などという指定はエラーとなることに注意しなければなりません。こういった語の解釈が問題になる場合は、指定の末尾に \relax を付けるか、明示的な中括弧を用いる \LaTeX 形式の指定を用いるのが良いでしょう。

```
Suddenly, you notice a cave spider  
\font\tenccr=ccr10\relax % at との結合を防ぐ  
at the top of ...
```

また、 \font は中括弧によって宣言される局所効力を持つので。文書全体に効力を及ぼすにはプリアンプルかスタイルファイルでの指定が必要です。

\TeX は \font 命令を見つけると TFM ファイルをシステム所定のディレクトリ群か環境変数 TEXFONTS でしめされるディレクトリから検索します。適切な TFM ファイルを置いておく必要があります。また、実際に追加したフォントが使用可能であるかは、出力ドライバ次第です。

2.2 和文フォントの追加

アスキー日本語 \TeX では、和文も 1 つのフォントとして実装されています。TFM の形式は違います (JFM ファイル) が、この差異は自動的に判定されます。そのため、フォントを追加する手順は基本的に同一です。例えば、この部報でリストに使われている中ゴシックは、私の環境では

```
\font\ninmgt=skgotmaa at 9pt
```

として追加されています。しかし、通常和文フォントは METAFONT では提供されないので、初期状態で JFM ファイルは存在しません。そのため、和文フォントを追加する際には、JFM ファイル min10.tfm をコピーするのが普通です。

アスキーの日本語 \TeX には和文の追加フォント名には一定のルールがあります。それは「最初の 2 文字がメーカー、次の 3 文字が書体、その後は太さ、変倍、斜体を表す」というものです。私は過去に写植で出力してもらったことがあるので、写研の書体の名前をそのまま使ってています。(sk=写研、got=石井ゴシック系、m=中、a=正体、a=直立) 詳しくは dvout や 日本語 \TeX の付属文書を参照して下さい。もっとも自分で使う範囲ではこだわる必要性はないかもしれません。別に正方形の書体の末尾が “aa” じゃなくても \TeX 自身は文句は言いません。

英文での注意事項はどれもそのままあてはまります。さらに、日本語フォントの出力は英文以上にドライバに依存することに注意しなければなりません。

3 文書全体を通じてフォントを置き換える

例えば、前号 196 号では出力全体に Concrete Roman を使用しました。このような処理をする場合、フォント設定全体を置換してしまうのが手っ取り早い方法です。楽である。前号で使った `ccr.sty` を見てみます。

```
\font\tenrm=ccr10
\font\ninrm=ccr9
\font\egtrm=ccr8
\font\sevrm=ccr7
```

10pt から 7pt しか変更していませんが、これだけです。 \TeX をインストールした人なら、`macros` か `jmacros` といったディレクトリに `jlfont2.tex` なるファイルがあるはずです。こいつを読むと、183行目当たりから、`\font` 命令の山があるのがわかるでしょう。こいつの名前にあわせて、`\font` 命令で新たにフォントを割り当てれば置換は完了です。ここでは `\newfont` は重複をエラーとして検出するので使えません。

4 サイズに応じてフォントを切り替える

第2節の要領でフォントを登録すると、フォントサイズを `\small` などで切り替えてもサイズは変わりません。これでは使い勝手が悪いので、きちんと連動するようにしてみましょう。

例として、`\ccr` 命令に Concrete Roman を割り当てるにします。

こんどは `jart10.sty` を覗いてみると、

```
\def\tiny{\@setsize\tiny{6pt}\vpt@\vpt}
```

なる記述があります。`\@setsize` は `jlfonts2.tex` 内で定義されている 4引数命令で、第1引数を現在のサイズとして保存、改行幅を第2引数に設定し、第3、第4引数を呼び出します。では `\vpt`, `\@vpt` はどうなっているでしょう? `jlfonts2.tex` をみてみます。

```
\def\vpt{\textfont\z@\fivrm
...
\def\pgt{\fam\gtfam\fivgt}\textfont\gtfam\fivgt%
\scriptfont\gtfam\fivgt\scriptscriptfont\gtfam\fivgt%
```

```
\@setstrut\rm\min}
```

```
\def\@vpt{}
```

なんか \pgt とかを再定義しています。実は \pgt は \gt の実体で、ここでは、「現在のフォントファミリが \gtfam で、フォントは \fivgt である」という風に再定義しています。フォントファミリーは数式モードを扱う上ですごく重要ですが、いかんせん面倒ですし、なんと 16 種しかサポートされておらず、日本語 \TeX が既に 14 種も使っているので使用は困難です。普通、フォントを使う場面は数式の外が圧倒的に多いですので、ファミリーの使用はあきらめることにします。そうすると、上の定義はこうなります。

```
\def\@vpt{  
  ...  
 \def\pgt{\fivgt}  
 \@setstrut\rm\min}  
 \def\@vpt{}
```

これなら簡単ですね。要するに「ゴシックは \fivgt だぞ」ということを再定義しているだけです。更に、 \circ の付いている方は拡張用の空定義です。これを使わない手はありません。

では早速 Concrete Roman の定義を。

```
\makeatletter  
 \font\tenccr=ccr10  
 \font\ninccr=ccr9  
 \font\egtccr=ccr8  
 \font\sevccr=ccr7  
 \@addfontinfo\@xpt{\def\pccr{\tenccr}}%  
 \@addfontinfo\@ixpt{\def\pccr{\ninccr}}%  
 \@addfontinfo\@viiipt{\def\pccr{\egtccr}}%  
 \@addfontinfo\@viiipt{\def\pccr{\sevccr}}%  
 \makeatother  
  
 \def\ccr{\protect\pccr}
```

他のサイズは自分で足してください。 $\@addfontinfo$ を使うために、コントロールシーケンス \makeatletter で \circ を一時的に使用可能にしている点に注意してください。勿論このフォントしか追加しないのなら、直接 $\@xpt$ を定義しても構いません。

また、 \ccr を直接定義しないのは、ここで定義した \pccr が fragile である（状況で定義が変わったり \TeX の内部状態を変えたりする）のが原因です。 \protect を使わな

いと、目次や脚注に使う時に障害となります。でもいちいち \protect を使わないといけないのは不便ですから、こうやって 2 段階定義をして robust なコマンドにします。

日本語フォントの追加もまったく同様です。

5 どうせなら欧文と和文くらい同時に…

\bf で日欧同時にフォントが切り替わるのだったら、中ゴシックと Typewriter とか組み合わせたいフォントはいろいろあります。こういう場合はどうやるのでしょうか。

やっぱり「技術は盗め」の格言通り、 \LaTeX の解決法をみてみましょう。

```
\def\rm{\protect\prm}\def\s{\protect\psl}%
\def\bf{\protect\pbf\protect\pgt}
```

なんだ、単純じゃないですか $\backslash(^{\wedge}_{\wedge})/$ 。万事解決。

```
\def\mgt{\protect\pmgt}%
\def\tt{\protect\ptt\protect\pmgt}%
組み合わせフォント
```

\pmgt は中ゴシックの \@addfontinfo での名前だとするとだとするとこんな感じで OK です。

また、こういったフォントの「生の名前」が出てくるのは j1fonts2.sty だけで、その他の latex.tex などは \tt などの定義をそのまま使っています。ですから、こうやって組み合わせてやれば、 \verb の中や verbatim 環境の中でも中ゴシックが使われます。

6 おわりに

\TeX の文書と言うと単調な 2 書体の文書を想像する人が普通でしょう。確かに \TeX は、目立つ文章よりも読みやすい文章を作るのがねらいですが、それでも中ゴシックや太明太明朝と言った書体はうまくつかえば全体の「黒さ」を調整し読みやすい文書の作成に役立ちます。うまく使ってやりましょう。

編集後記

みなさんがこの記事を読んでいるということは、はまった状況をなんとかぬけだして部報がでたということです。(いぬ。のlog参照)

追い出される方がいっぱい記事を書かれている昨年の188号が気に入ってる自分としては、追い出される方の原稿を載せられなかつたのが残念でなりません。なんかぜんぜん追い出しコンバの部報じゃないような感じになってしまったし、2号分の記事がのっている「はず」なのにえらく薄いし…。今回ばかりはかなり「不満のある」しあがりになってしまいましたね。この失点はオリパンフで取り戻すべし [宣言]。

昼間に Nishi さんに「そりゃー原稿が足りない時は編集長が書いて補うんだ」とか言われてしまったので、本当に原稿2本書きました(^^;。一本はスキーの帰りに宣言した帰りのバスのルート、もう一本は何人かから聞かれていたTeXのフォント追加方法です。それでも薄い部報になってしまった。

本当にちゃんとこの部報は出るのだろうか？（まだ言ってる。）

(Aleph-NULL)

理論科学グループ 部報 197号

1995年3月9日 初版 第1刷発行

発行者 金子 淣

編集者 大岩 寛

発行所 理論科学グループ

〒153 東京都目黒区駒場 3-8-1

東京大学教養学部内学生会館 305

Telephone: 03-5454-4343

(C) Theoretical Science Group, University of Tokyo, 1996.

All rights are reserved.

Printed in Japan.

理論科学グループ部報 第197号

— 追い出しコンパ号 —

1996年3月9日

THEORETICAL SCIENCE GROUP