

# WORD

# 2008.11 From College of Information Science

7

# 特集

# WORDのお仕事

# MAXプリン

# @雙峰祭 ~そして伝説へ~

# ハジメル Haskell

# 公衆電話巡り

# 登山 in 名古屋

Λ\_„„Λ  
(`·ω·)  
/\O==O  
/\_||\_|  
U,(( ))

# WORD の原価は 4兆ジンバブエドル号

# 4兆ジンバブエドル号 日次

MAXプリン@雙峰祭～そして伝説へ～ P. 3

WORD のお仕事 P. 17

公衆電話巡り P. 23

ハジメルハスケル P. 31

登山 in 名古屋 P. 46

編集部からのお知らせ P. 52

\$  
\$  
^\_\_,,^ \$\$\$\$\$\$\$\$\$\$\$\$\$  
( `・ω・) \$\$\$\$\$\$\$\$\$\$\$\$\$  
/\ ○==○ \$\$\$\$\$\$\$\$\$\$\$\$\$  
/ ||\_| | \$\$\$\$\$\$\$\$\$\$\$\$\$  
し'\_\_(\_) -(\_) -(\_) -(\_) -(\_)

# 学祭後記 MAXプリン@雙峰祭 ～そして伝説へ～

文 編集部 いのひろ

## この記事は

あの MAX プリンを雙峰祭で売っちゃったという、嘘のような本当の話。その舞台裏を時系列でお届けします！

さすがに 8 リットルバケツプリンを作って売るわけにもいかないので、一般的なプリンのサイズで作ることにしました<sup>1</sup>。

## 雙峰祭 1 週間前

なんと雙峰祭 1 週間前にしてまだ材料、容器などの手配をしていない状態。とりあえず準備日（10月 10 日）の 1 週間前である 3 日（金）までには手配を終えておきたかったのですが、それ以前に MAX コーヒー<sup>2</sup> が何リットル必要で、容器が何個必要なかすら把握できていなかったのです。

結局週明けの 6 日（月）に「これは結構まずいのでは！」ということになり、情報科学類生お得意のプログラミングを用いて、130cc のプリンを 300 個作るには材料がどれだけ必要なのかというのを求めました<sup>3</sup>。

実行結果はこんな感じです。このコードではプリン 1 個を 100 円で売る計算になっています。

1 日にプリン（130cc）をいくつ作りたいか入力してください

150

-----

MAX コーヒー総本数: 70.2, プリンエル総本数: 86.3636363636364

全部で MAX コーヒーは: 70.2 本, 7020 円

全部でプリンエルは: 86.3636363636364 箱, 12954.5454545455 円

2 日間での容器代金: 6300

総もうけ: 3725.45454545454

\*1 MAX プリン「ち、小さくても甘いんだからねっ！」

\*2 泣く子も黙る主原料が加糖練乳なコーヒー。ジョージア MAX コーヒー。

\*3 コードは「<http://d.hatena.ne.jp/InoHiro/20081007/1223335887>」に書いてあります。

## MAX プリン@雙峰祭 ~そして伝説へ~

大学近辺のドラッグストアなどで MAX コーヒー (500ml) が 1 本いくらするかなどのリサーチは事前に行ってるので計算 자체はすぐ出来ました。あとは電話で発注するだけです。

実は歴代の MAX プリン制作の時にお世話になっていたカワチ薬品 (つくば桜店、つくば大穂店) ではここ最近 MAX コーヒーを取り扱っておらず<sup>\*4</sup>、MAX コーヒーについてはドラッグ寺島にお世話になりました。3 ケース (72 本) 注文。「問屋に注文して、10 日（金）の午前中入荷です」と言われたはずなのに、翌日（火曜日）入荷を知らせる電話がありました。すばらしい。

プリンエル<sup>\*5</sup>ですが、これはカワチ薬品で引き続き取り扱ってくれていたため、大穂店に 88 個注文しました。

容器 (140cc の耐熱プラスチックカップ)、スプーン、フタなどはその手の専門店が Web 上にたくさんあったので、在庫を確認して 300 個分注文。その他、加熱に必要なカセットコンロのボンベなどは準備日に買いに行きました。

製品名	単価	個数	合計
MAX コーヒー (500ml)	100	72	7200
プリンエル (66g)	150	88	13200
容器 (140cc, 100 個)	1428	3	4284
スプーン (1000 本)	1869	1	1869
フタ (100 個)	420	3	1260

この記事を書いていて残念に思ったのは、せっかく 72 本も MAX コーヒーを買ったのに、MAX コーヒータワーを作らなかったことです。残念すぎる。プリンエルも 1 ケース +  $\alpha$  分あったわけですが、写真を撮るのを忘れてしました。残念すぎる。

もう一つ忘れてはいけないのは「WORD 雙峰祭特別号」の執筆。これまでの WORD から「Project MAX!-中毒者達-」、「STAR WORDS ~Episode 2008 A NEW PUDDING~」の 2 本、書き下ろし記事として「MAX プリンのおいしい作り方」、「歴代 MAX プリン図鑑」を収録する予定でした。とりあえず書き下ろし記事は 2 日くらいで書いて、編集部内でチェックしてもらいました。

「MAX プリンのおいしい作り方」では 8 リットルプリンの作り方と一般的な大きさのプリンの作り方の両方を無駄に収録。「歴代 MAX プリン図鑑」ではこれまで WORD でお届けしたプリンはもちろん、WORD では取り上げたことがなかった、「2008 年度筑波大学 AC 懇親会」の為に作られた MAX プリン（もちろん 8 リットル）の様子も収録しました。実は記事だけではなく表紙も目次も裏表紙も私が作りました。しかし、まあ、なんというか連日徹夜みたいな感じだったので、どう見ても「手抜き」みたいな記事になっております。ごめんなさいごめんなさい。

\*4 兄貴「！！…節子！ それ MAX コーヒーやない！ 練乳や！」

節子「なんでカワチ、MAX コーヒー (500ml) 取り扱ってくれないん？」

\*5 毎度おなじみのプリンの素。ハウス食品製。

\*6 カワチさんは入荷しても電話くれなかつた…



### 雙峰祭準備日（10日）

午前中はまさかの授業。

昼から準備開始です。買い出し組と準備組に別れて作業。買い出しひなたちゃん（牛久市、20歳、男性）と、かづきお氏が行ってくれることに。私は特別号の印刷などを行っていました。

夕方から机や椅子を外に出し、テントを組み立てました。しかし、学実委のテント班がこれまた修羅場っていて、テントを借りに行っても「部品が足らないので、1時間くらいしてから来てください。」と言われてしまいました。1時間後に行ったところ、まだ時間がかかると言うことで整理券を発行してもらって、また待つことに。

結局合計で2時間待ったのですが、やっと借りたテントは天井の布のサイズが間違っていて、交換。しかし交換した布もまた長さが足りません。もう一度交換しに行き、「テント貸出所にある一番大きなものです」と言われて渡された布も長さが足りない。また、私たちの他にも色々な企画で、部品や布が足りていないということだったので、次の日に持ち越しということになりました。

あとは看板を使う高さ2メートルくらいの板を借りてきたのですが、この時点では準備不足で看板に貼る紙が用意できませんでした。

### 雙峰祭1日目（11日）

現場の準備は終わったので家に帰って次の日に備えたいところでしたが、やることが山積み。特別号の綴じ込み、当日配布予定のチラシの印刷、看板に貼るA0~A1サイズの紙の印刷、調理用具の調達などなど。徹夜は必至です。前日も特別号の編集などあまり寝ていない気がしたのですが、準備不足な自分がいけないので仕方ありません。

まずは特別号の綴じ込み作業です。また特別号のほかに6号（「米どころか自分にもカビ」号）も雙峰祭で配布するために200部増刷しましたが、こちらは他の編集部員が綴じ込んでくれました<sup>7</sup>。感謝。

---

\*7 そのときの様子が「WORDのお仕事」に載っています！ 探してみてね！

## MAX プリン@雙峰祭 ~そして伝説へ~

綴じ込みが終わったら、当日配布予定のチラシを作ります。学実委の許可がないものは配布できないので、印刷の前に許可をもらう必要があります。既に数パターン許可をもらっていたのですが、もっとインパクトのあるものをということで、かづきお氏と試行錯誤。「限定 300 個」とかスイーツ(笑)な人が食いつく単語を入れないとダメだ!といったような、かづきお先生の大変参考になるアドバイスをいただき、出来たのがこれらのチラシ。



これは甘い  
MAX プリン祭 2008

MAX プリン  
1日目 (10/11) / 2日目 (10/12)  
第3エリア A棟前にて  
WORD 第6号 / 雙峰祭MAXプリン号  
無料配布中



MAXコーヒー + プリン = MAXプリン  
MAX プリン  
10/11, 12  
3A棟前にて  
情報科学類誌 WORD

「諸君はこれから、奇跡を、  
MAX プリンを見るだろう。」

雙峰祭史上もっとも甘いプリン、登場  
**300個限定生産**  
**3A棟前にて**

とりあえずこんな感じでいくつか作り、朝方 5 時頃に「腹が減った! すき家!」とすき家へ。そのついでに実委室に寄って、出来立てホヤホヤのチラシに許可をもらいます。案の定、雙峰祭

前夜<sup>\*8</sup> ということで実委室には数人の学実委の中の人たちが。

我々「えっと、チラシの許可をお願いしたいんですが…」

学実委「んとー、それは良いんですが、いま何時だと思ってるんですか？バ●なの？●ぬの？」

我々「ご、ごめんなさい…」

“じょうほう”的にはオールオッケイな時間帯<sup>\*9</sup> だったのですが、学実委の中の人が“じょうほう”な人とは限りませんよね。ましてや、連日のお仕事に疲れていたのか、かなり怖い感じでした。ごめんなさいごめんなさい。

すき屋で「たくさん食ってエネルギーを充填しないと今日一日もたない！」ということで、大盛りを注文。しかしこの大盛りによって、後に強烈な睡魔に襲われることになるを、このとき我々は考えもしなかったのです…！

大学に帰ってきてから許可をもらった原稿をスキャンして画像に。その後、計算機室で大量印刷を開始。ここで胃の中の牛丼が睡魔に変身し、襲ってきます。眠い目をこすりつつ、数パターンを印刷。気づけば既に朝の6時とか7時だった気がします。

ついに、かづきお氏が睡魔に負け、私は一人で調理用具の確保に。これまで何度も無く8リットルプリンを作るときにお世話になってきた鍋、お玉などを確保。あとはカセットコンロなどを準備して、調理が開始できる午前8時を待つだけです。

いつの間にか外は雨。8時から調理開始は絶望的なので、その時間を利用して看板に貼る紙を作ります。印刷はWORDの備品であるA1をnメートル印刷（nは任意）できるhpのプリンタを利用。上半分に先ほどの「300個限定」チラシを少し加工したものを、下半分にはインパクトを求める8リットルMAXプリンの写真をカラーで印刷したものを貼りました。

---

\*8 もう朝だけど

\*9 え、だって“じょうほう”なみんなは、朝までコーディングしたりニコニコしたりするでしょう？

## MAX プリン@雙峰祭 ~そして伝説へ~



**3A棟前で限定発売中！**

## MAX プリン@雙峰祭 ~そして伝説へ~

結局調理開始は 11 時 30 分からになりました。沸騰しない程度に熱し、次にカップへ。お玉で移すのですが、これがなかなか難しい。既に机の上は泥遊びしているような感じに。

カップに移したらある程度まとめて冷蔵庫に入れます。プリンが固まるには 2 時間程度かかることがわかつていたので<sup>\*10</sup>、出来上がるのは 14 時頃。「プリンはいつできますか?」と前号の WORD を見て来てくれたような人たちが早速いらっしゃったのですが、「すみません、売り始めるのは 14 時頃です。」としか答える事ができませんでした。すみませんでした。

そろそろ冷え固まってる頃かなあと冷蔵庫をあけてみると、冷たくなく、むしろ温かい…<sup>\*11</sup>。冷蔵庫が冷える前に暖かいプリンを入れてしまった為です。しかし、他に手ないので冷蔵庫が冷えるのを待ちます。

1 時間遅れて、15 時ごろになって、やっと販売を開始することができました。



お客様が来ません…( ´・ω・`)

しかし思うように売れ行きが伸びません。お客様が周りのサークルに持っていくかれてしまいます。

また、インパクト欲しさに作った看板ですが、人によってはただのグロ画像にしか見えなかつたようで、「なにこれ!」と驚く人もいる一方、「うわあ…」とドン引きして過ぎ去る人も少なくなかったです。特にお婆ちゃん。すごい「残念そうな顔」で看板を見ていました。

\*10 数日前に私の家の冷蔵庫で実験してみました

\*11 うわあ…冷蔵庫の中…すごくあたたかいナリ…

## MAX プリン@雙峰祭 ~そして伝説へ~

ところで WORD の切り札、メイド服の登場です。1日目は mitty 氏がメイドさんに。



中央図書館と第一エリアを結ぶ橋の上で、朝方大量に印刷したチラシを配ってもらいました。しかし、思っていたよりもチラシの効果はありませんでした。苦戦です。



夕方になるにつれお客様も少なくなり、このままではプリンが余ってしまいます。衛生上 1 日目に作ったものを 2 日目に持ち越す事はできません。価格を下げてもあまり効果はなく、19 時頃まで売り続けて 70 個程度しか売れませんでした。我々は 2 日目にリベンジを誓ったのです。

反省会（1日目夜）

1日目の問題を整理し、2日目での解決策を考えました。

問題	解決策
プリン溶液が熱いうちに冷蔵庫に入れてしまったのでなかなか冷え固まらなかった。	冷蔵庫は保存だけに利用し、プリンは冷蔵庫に入る前に氷水で冷却する。
看板に貼った写真のインパクトが強すぎて、グロ画像にしか見えなかつた。	インパクトをそのままに、グロ画像を MAX コーヒータワーの写真へ。
人手が少なく客引きができなかつた。	2日目は人員を確保して、積極的に客引きをする。
値段が高かつた。	価格を改定する。
チラシのサイズが大きかつた。	チラシのサイズを半分にする。 エコいWORDをアピール☆
チラシの効果が少ない。	チラシの裏面に簡易的な地図を載せる。 メイドさんにもっと頑張ってもらう。

雙峰祭 2日目（12日）

2日目は、前日の雨が嘘のように晴れ、朝8時から調理を開始することができました。看板は下のように改造されました。



## MAX プリン@雙峰祭 ~そして伝説へ~



これまで「“おそらく”雙峰祭史上最も甘いプリン」と控えめに書いていたのですが、MAX コーヒープリンを超える甘さを実現することは極めて困難<sup>\*12</sup> ですので、「雙峰祭史上もっとも甘いプリン」を名乗りました。

また設置位置も 1 日目よりも目立つ位置に配置したので、天の川の対岸からも「なんだ、あのカオスは！」といった具合に、非常に高い宣伝効果を発揮しました。

さて、どんどんプリンを作っていきます。

\*12 たぶん。



yasuharu 氏 によってプリンにカラメルが投入される瞬間

1日目はそのまま冷蔵庫に入れていたのですが、今日は一工夫。ホットプレートなど底が深い鍋をたくさん用意し、氷水を張ります。そしてプリン溶液を入れたカップを並べていきます。この行程で余熱を取り去るのです！！！！！11

こうして溶液がほぼ常温レベルになったところでフタをし、冷蔵庫に移します。

この戦略で10時の販売開始時点では、30個ほどすぐに売り出せる状態でした。

2日目は飛ぶように売れました。一度に10個買っていてくれる人や、「後でまた買いにくるから、2つ取っておいてくれ」と言ってくれたお客様もいました。WORD編集部のOBや、バケツプリンについて熱く語るお客様もいました。

2日目はmitty氏に代わってひなたちゃんがメイドさんに。1日目以上にたくさんのチラシを配ってもらいました<sup>\*13</sup>。

---

\*13 刷りすぎて配りきれなかったチラシが私の手元にあります

## MAX プリン@雙峰祭 ~そして伝説へ~



さすがに 1 日で 200 個以上売るのは無理ではないかと思っていたのですが、気づくと空のカップがありませんでした。

しかし最後の 1 回だけ、プリンがうまく固まりませんでした。プリンエルの分量を間違えたのか、加熱が足りなかったのか。原因は不明ですが、仕方なく最後の二十数個は「飲む MAX プリン」状態で売る事になってしまいました。ごめんなさい。

余った MAX プリン溶液は、移すカップが無いため、小さめの片手鍋に入れて冷却。



## MAX プリン@雙峰祭 ~そして伝説へ~

後日、片手鍋 MAX プリンはスタッフでおいしくいただきました。



MAX プリンを頬張る、かづきお(仮名)20歳、独身。一の矢学生宿舎にて撮影。

そして午後5時頃、手元にあった MAX プリンをすべて売る事ができました。



## MAX プリン@雙峰祭 ~そして伝説へ~

2日目はおよそ200個強、2日間で合計270個程度売る事ができました<sup>\*14</sup>。

### まとめ

売り上げはおよそ34800円。利益は殆ど出ませんでしたが、私個人としては大満足な内容でした。協力してくれた友人、献身的な学実委の皆さんに心から感謝します。また、誰よりもMAXプリンを買ってくださったあなたに感謝です m9(`・ω・')。

そして「WORDいつも読んでます」と言ってくれるお客様がたくさんいらっしゃったのが非常にうれしかったです。WORDを作っていて、これほどうれしい事はありませんでした。また「MAXプリンは前から食べたかった」と言ってくれた方、お友達数人を引き連れて「AC懇親会のときのやつですよね?」と買いに来てくれた方もいました。

WORDを第2エリアでも配布してくれという要望がありました。今のところそういう予定はありません。あくまでも情報科学類誌という位置づけですので、第3エリアのみでの配布となっています。

さて、突然ですが、この記事をもってMAXプリンネタはおしまいにします。別に誰かに「いい加減にしろ」と言われた訳ではなく、私が個人的に引き時として最適だと思ったからです。MAXプリンがこの世に誕生してから1年くらい経ちました。様々な方に記事を読んでいただき、興味をもっていただきました。アホなネタにお付き合いください、ありがとうございました。

何かイベントで作ってくれなどありましたら、「inohiro@coins.tsukuba.ac.jp」まで連絡してください。面倒臭くなかったら作ります。



いのひろの次回作にご期待ください♪

---

\*14 あれ、でもカップは300個分購入したはずなのに…

# WORDのお仕事

文 編集部 dorio、yasuharu

「WORD 編集部ってよく分かんないけど、変な人が集まってる所だよね。ゆとりだよね。」とはよく聞かれる言葉。変な人が集まっている事は否定しませんが、しかし変な人は変な人なりに頑張って活動しています。そんな変人・奇人・詰み人<sup>\*1</sup>の巣窟<sup>\*2</sup>WORD のお仕事を、少しご紹介しましょう。

## WORD 編集部って、どんな所？

至って普通の、だけど趣味に打ち込んでいる学生が集まっています。電子工作が趣味な学生やスイッチが好きな学生、プログラミングが好きな学生やサーバのセキュリティに命を賭けている学生などなど。そんな多様な趣味を持った学生が集まって、「濃い」記事が書かれています。



積まれているネットワーク機器



IP 電話

## お仕事。

WORD 編集部の本業は勿論「学類誌 WORD」の編集です。部室には、編集作業には欠かせない裁断機、巨大ホッチキス、巨大万能基盤、巨大キャパシタ(86000MFD)<sup>\*3</sup>等々がこの棚に納められています。いくらデジタル編集のこのご時世でも、手作業でやらなければならない事は沢山あるのです。

\*1 「罪人」でなく「詰み人」。仕事や課題がスタックされすぎている人。PUSH ばっかりして POP しないんだよね。みんな、計画的に POP しよう NE ☆

\*2 「すぐつ」じゃないよ。「そうくつ」よ。

\*3 原文ママ。マイクロなのかメガなのか。とりあえず手のひらよりデカイ。

## WORD のお仕事



綴じ込み作業も勿論手作業です。熟練の部員により一冊一冊丁寧に綴じ込められています。綴じ込み作業には非常に高度な技術を要するので、新入生にはまず落丁等が無いかのパリティチェックのお仕事が命じられます。野球部で言う球拾いみたいなものでしょうか。因みに、写真に写っている<sup>\*4</sup>綴じ込み作業中の人物はこの道3年のベテラン部員です。



### 編集部の環境。

以前は机の板の中に蟻が巣食い、棚の裏側には得体の知れない奇妙な虫がのたくっている魔境でした。しかし、2年歳月をかけて少しずつリニューアルされ、今ではその美しさにオバマ氏も当選してしまうほどです。

---

\*4 プライバシー保護のため、手しか写っておりません。

また、WORD 編集部はデジタル編集の設備が揃っており、締切り日前日のデスマーチにも耐えられる体制になっています。



また、ノートパソコンに FreeBSD<sup>\*5</sup> や Linux をインストールしている部員も多い為、編集ソフトを使えるようにサーバが用意されています。そして、過去の記事もサーバに納められ、すぐに参照できるようになっています。



WORD を支えるサーバ群

---

\*5 386BSD 及び 4.4BSD-Lite をベースとした、非常に堅牢な UNIX 互換 OS。その信頼性から、Yahoo! やさくらインターネット、その他多くのサーバで利用されている。

## WORD のお仕事

### 戦う WORD。

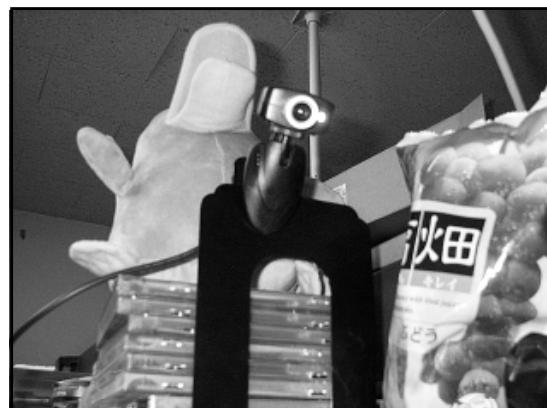
真実を伝える機関。こういった所は、いつも様々な勢力から狙われています。勿論 WORD 編集部も例外ではありません。そんな脅威への、WORD 編集部の対策はどうなのか。。

勿論完璧です。

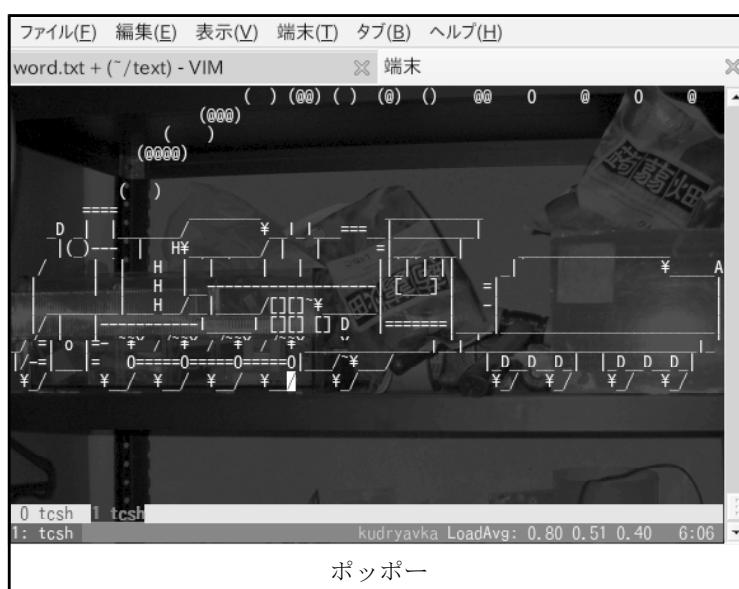
外部からの侵入者は部室内に設置されたカメラにより察知され、その犯行を未然に防ぎます。また、現代のライフラインと言えるネットワークの冗長性も確保され、サーバ群のセキュリティも固められています。

そして、各個人の意識向上への活動も行われています。個人のコンピュータでログアウトしていないものがあると、「全ての unix コマンドを sl<sup>\*6</sup> に置き換える」という警告がされるのです。

このように、堅く固められた WORD 編集部。有事の際でも情報を発信し続けられるのは我が WORD 編集部において、ほかにないでしょう。



編集室を監視する監視カメラ



\*6 コンソール上に汽車のアスキーアートが走るジョークコマンド。

ザ・癒し。

このように集中できる環境が整っている編集部。しかし、人間には心の潤いというものが需要です。しかし、そこは WORD 編集部。癒しも万全です。

例えば、Mac のハードウェアが無いと発狂する人種向けに、美しいポリバケツ G4 のハードウェアが置かれていたり、干~~ネ~~タユーモア分が足りないと息が出来ない種族向けに「まりもっこり」が置かれていたりします<sup>\*7</sup>。また美しいあきたこまちも設置され、日本人の美的感覚を刺激してくれます。



美しいポリバケツ



おいしいお米



まりもっこり

\*7 編集部入り口でお出迎えしてくれます。

## WORD のお仕事

### 我らゆとり。

複素平面を習っていなくて教授が頭を抱えたり、Neeeeet が多かったり、ネットカフェ難民になったり。色々社会にゆとり(笑)旋風を巻き起こしている注目の世代、「ゆとり」。団塊からゆとりへ。ペンからキーボードへ。世代交代の波は、いつでもどこでも起こっています。我が WORD 編集部も全ゆとり時代へと突入しています。

今日もどこかでマンナンライフの蒟蒻畑を食べながら、vim で記事を。私たちはマンナンライフの蒟蒻畑を応援しています。

### 最後に

この記事は多少の事実と多くの願望、少しの嘘と多くの愛が含まれています。



# 公衆電話巡り

文 編集部 IX



湖畔に佇むビニール傘……  
哀愁漂うその背中で彼は……  
我々に何を伝えようとしているのだろうか……

この世（鏡波）には……目には見えない闇の住人達（妖怪・一日漬し<sup>\*1</sup>、など）がいる……

彼らは時として牙をむき、君たちを襲ってくる。  
彼は……そんな奴らから君たちを守るため、地獄のそこからやってきた正義の使者、なのかもしない……<sup>\*2</sup>

\*1 何もやらずに一日が終わっていたら、こいつらの仕業です。

\*2 ぬ～べ～

## 公衆電話巡り

### ■はじめに

ある筑波大生にこんな災難が降りかかりました。

トイレでするべきことを終えて、気を抜いた瞬間……携帯電話が便器にホールインワン、即座に昇天。

彼は焦りました。彼にとって携帯電話は、ママと連絡を取る唯一の手段だったのです。

宅通の彼にとって、彼女とのコミュニケーションはとても重要なのです。私生活のほとんどは彼女のサポート無しでは成り立たないでしょう。

となると、連絡手段を確保しなくてはなりません。彼は少し考え、そして、気づきました。

**公衆電話があるじゃないかっ！！**

そんなわけで、公衆電話巡り、スタートです。

### ■決行

2008年10月某日、皆様方は勉強やサークル活動などに明け暮れていたことでしょう。そのとき見ませんでしたか？

大学の敷地内をさまよう、虚ろな目をした中肉中背の男を……

はい、それは私、IX<sup>\*3</sup>の姿であったのでしょう。

---

\*3 H：173cm、W：94kg、花も恥じらう19歳。スリーサイズはヒ・ミ・ツ

### ■図書館周辺

さて、公衆電話を巡るわけですが、近場にあることは絶対条件だと思います。なので、図書館周辺から探すことにしましょう。



人混みにまぎれ、ひっそりと佇む・広場電話 1

人通りの多い場所にあるため、記憶の片隅にある方も多いのではないでしょうか。私は真っ先にここが思い浮かび、全力で駆けつけました。

やったっ！！ これで電話が掛けられるっ！！



故・障・手・配・中 (はあと

待て、あわてるな、これは孔明の罠だ。

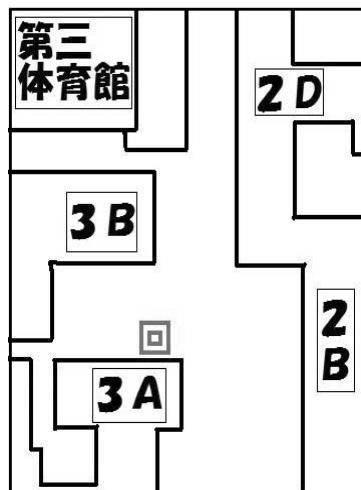
昨今、携帯電話の普及で公衆電話は激減しました。つまり、公衆電話はレアもの、特別天然記念物、ステータスかつ希少価値なのです。ましてや「故障手配中」となれば、かわいい幼馴染み並の価値があるに違いありません。そう思う事としましょう。

## 公衆電話巡り

### ■ JYG（常陽銀行）ATM周辺

全く、ひどい目に遭いました。私は何も悪くないのに……。

気を取り直して、公衆電話巡りを続けましょう。大丈夫、近場の公衆電話はもう一ヵ所あります。



道行く学生達を見守るナイスガイ・広場電話2

ここも比較的通りの多い場所にあるため、主に第三エリアを拠点としている方はよく見かける光景であります。また、実際に使ったことのある方にしか判らないと思いますが、ここには可愛いマスコットキャラクターがいるのです。



女郎蜘蛛たん<sup>\*4</sup>

ね？ かわいいでしょ？

---

\*4 クモ綱クモ目アシナガグモ科の大型のクモ。

そのままでも十分かわいいが、擬人化できるとなお良い。

ここは故障中でなかったので、可愛いマスコットキャラを眺めながら、無事に連絡をとることができました。

**■公衆電話巡り終了のお知らせ？**

遂に、ママに連絡をとることが出来た IX。目的を達成し、彼は心地よい安堵感に浸っていた。しかし、その一方で、彼の中にはある感覚が芽生えていた。

あれ……公衆電話がかわいく見えてきたぞ。

そう、彼の公衆電話巡りはまだ終わらないのです。

**■関鉄バス・筑波大学西停留所周辺**

連絡は取れた。第一目的は達成出来ました。

しかし、二カ所巡っただけでは心許ない。この二カ所が故障手配中にならない保証は、どこにもないからだ。さあ、筑波大学内の**公衆電話の分布**を調べるために、公衆電話巡りを続行しましょう！！



みなぎる存在感・バス停電話 1

ここは、第三エリアから少し離れた大学西停留所の近くにある公衆電話。

個人的な見解ですが、体専や芸専のエリアには、他と異なった雰囲気が流れているような気がします。ここでも、そんな雰囲気に押され、看板の陰からの撮影です。近づいてとった写真もあったはずなのですが、データの行方がわかりません。いったい何が……。

さて、かなりの距離を歩き回り、公衆電話巡りもこれから大詰めを迎えてゆく訳なのですが… …。

公衆電話巡り



野生の SAKAMICHI Lv.100

野生の SAKAMICHI が現れた。



IXは逃げることが出来ない。

SAKAMICHI ……それは、栄養過多かつ運動不足である現代人の敵<sup>5</sup>。

しかし、今回は“元”運動部の素晴らしき身体能力の前にひれ伏してもらおうじやないか…

• • •

この坂を越えれば、公衆電話がっ！！……

もうだめ……ばたんきゅつ……

昔々あるところに、財宝（公衆電話）を求めるSAKAMICHIに戦いを挑んだ青年が居た。

戦いを終えたその姿……

無様この上なかつたと伝えられている……



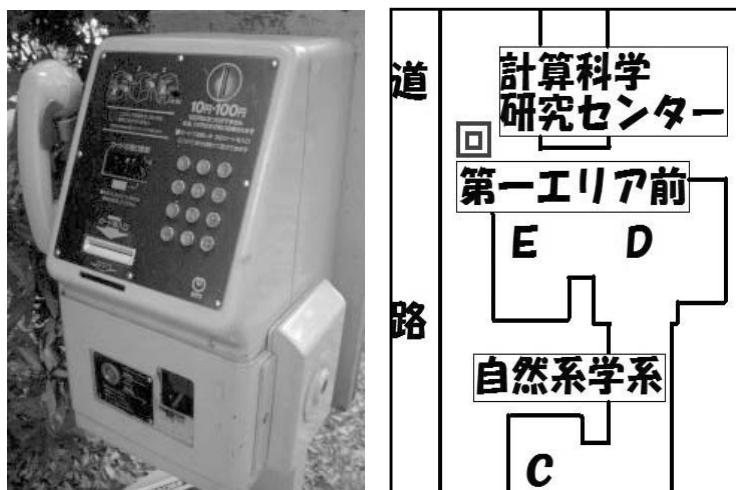
---

\*5 主に私。

■関鉄バス・第一エリア前停留所周辺

手強い相手でした。“元”運動部の身体能力をもってしても、僅差で勝利するのが精一杯でした。恐ろしい限りです。

そして、幾多の困難を乗り越えてたどり着いたのが……



新緑の巫女<sup>\*6</sup>・バス停電話 2

第三エリアよりバスで一分、徒歩三分。第一エリア前停留所の近くの公衆電話。  
雨の日には複数のビニール傘が放置されるので、傘立て的公衆電話と呼んでいます。  
また、見た目が特徴的でして……



森の～中に～昔から住んでる～♪

とってもトロックなことです。

この後、私は公衆電話巡りで酷使した体を引きずり、ふらつきながら編集部に戻りました<sup>\*7</sup>。

---

\*6 私には巫女さんに見えました。疲れすぎていたのかもしれません。

\*7 実は、全行程が徒歩。もうやめて！ IX のライフはもうゼロよ！

■あとがき

ふう……

**あらかじめ把握していた場所以外の公衆電話も探そうと思い、  
とんでもない距離を歩き回って疲れたのに、何も見つからず、  
がっかりしながら編集部に帰った  
ような気がしていたけれど、  
きっと、そんなことは全然無かつたんだＺＥ☆**

この記事は、携帯電話をトイレに落とした“携帯再起不能事件”が引き金となっております。実際に携帯は再起不能となり、五千円かけて修理するというかたちとなりました。今では、新品同様になり、絶賛稼働中です。

この機会に、少しでも良いので公衆電話をのことを思い出していただけたら幸いです。

# ハジメルaskell

文 編集部 goth

---

## ——ハジメルaskell——

ハジメマシテな方もハジメマシテでない方もご機嫌よう。

一部噂では変態言語使いと呼ばれてるとか、呼ばれてないとか噂されてる goth です。

そんな goth が皆様に今おすすめする言語、それはばり Haskell。

関数型で純粋、強い静的型付けを持つ非正格な----おっと、そこの御仁「不正確」ではなく「非正格」でござるよ。

携帯電話で打とうとしたら「ちかめにしぬぬ」<sup>\*1</sup>とか出て、「すわっ、我が余命幾ばくか!?!」となってしまう Haskell という名前は論理学者の Haskell B.Curry 氏が由来とか。

誰?とかいうツッコミの答えは置いておきまして、なんかすごそうな言語をチャチャッと紹介しちゃいましょう。

---

## ——入手とインストール——

とか説明してると長くなるんで省略。…したいのですが、そうもいかないのでご説明をば。

現在 Haskell の処理系で一番有名なのがおそらく GHC と呼ばれる処理系。

"G"から始まっているので思わず GNU 系<sup>\*2</sup>かな?とか思ってしまいますが関係が無く、"Glasgow Haskell Compiler"の略らしいです。

コンパイラというくらいなので、実行可能ファイルを作ることもできますし、それだけでなくインタプリタも備えています。

その他、Hugs と呼ばれる処理系も有名ですが、ここでは筆者が主に使ってる GHC を使いたいと思います。

各種 UNIX 系 OS ないし、各種 Linux ディストリビューションでは大体の場合パッケージ管理ツールで入手・インストールが可能ですので、以下の作業は不要になると思われます。

yum や apt, urpmi, portage, ports など身に覚えがあればそれです。

GHC の入手先ですが、Google の日本語サイト検索<sup>\*3</sup>ではトップに出てこないようですので、それっぽいサイトから巡っていただきとか、以下の URL を入力してください。

<http://www.haskell.org/ghc/>

表示されたページの左側のメニューから"Download"をクリック、次いで"Current Stable Release"のすぐ下のリンクをクリックします。原稿執筆時では 6.8.3 が現在の安定版のよう

---

\*1 ちかめにしぬぬ: 携帯電話の入力モードを日本語にしたまま "Haskell" を打とうとするところ

\*2 GNU 系: gcc や gdb、gmake など

\*3 検索: Google で調べると関連検索に "GHC 化粧品" の文字が…。奥様、化粧品は "DHC" でございます。

## ハジメル Haskell

です。

あとは、"Binary Packages"からお使いの環境にあったファイルをダウンロードしてください。

「ワシはソースパッケージ使いたいんじやー」という方もいらっしゃるかもしれません  
が、GHC に限っては"Binary Package"をおすすめします。と、いうか、**GHC のコンパイル  
に GHC が必須<sup>\*4</sup>**なので、おとなしく"Binary Package"を落としましょう。それからソースを  
コンパイルです。

Windows 版はインストーラになっているので、特に問題なくインストールできると思われます。

端末から実行したい場合には、インストールされたディレクトリにパスを通す必要があります。「マイ コンピュータ」を右クリックして「プロパティ」を選択、タブから「詳細設定」の「環境変数」と書かれたボタンをクリック。

ユーザなりシステムの環境変数の"Path"に"C:\Program Files\ghc-6.8.3\bin"を加えます。それぞれのパスが";"(セミコロン)で区切られて並べてられていますので、";"をいれるのをお忘れなく。

学類の Mac には既に Ver.6.4.1 が入っているようです。

今回の記事は基本的なことしか書いてないので、学類の GHC で十分だと思います。

---

——— まずハジメニ ———

とりあえず起動してみましょう。

GHC にはインタプリタもついてますし、その場でフィードバックがあった方が学習しやすいと思いますので、インタプリタを使うことにしましょう。

Scheme で言うところの DrScheme のような GUI なインターフェースはありませんので、コンソール<sup>\*5</sup>を立ち上げて、ghci と打ってみてください。

```
GHCI, version 6.8.2: http://www.haskell.org/ghc/ :? for help
Loading package base ... linking ... done.
Prelude>
```

の様に表示されれば ok です(あ、ヴァージョンが古い)。

最後の行の"Prelude>"はプロンプトですので、さっそくここに色々と入力して結果を見てみましょう。(斜体が入力部分です)

```
Prelude> 1 + 2
3
Prelude> 3.4 * 5 + 6.7 / 8.9
17.752808988764045
```

こんな感じで計算することができます。

そんなん電卓でやれよ、と言われそうですが、もう少しやってみましょう。

---

\*4 GHC のコンパイルに GHC が必須: 卵が先か鶏が先か…。コンパイラにはよくある話。

\*5 コンソール: iTerm や XTerm など。

```
Prelude> div 10 2 -- 整数の割り算。関数呼び出し*6に括弧やカンマはいらない。
```

5

```
Prelude> div( 10, 2 ) -- というか、括弧やカンマをつけるとエラーになる。
```

No instance for (Integral (t, t1))

arising from a use of `div' at <interactive>:1:0-12

Possible fix: add an instance declaration for (Integral (t, t1))

In the expression: div (10, 2)

In the definition of `it': it = div (10, 2)

```
Prelude> div (div 100 5) 5 -- 関数に関数の結果を渡したいときは括弧で括る。
```

4

```
Prelude> div div 100 5 5 -- これはdivに4つの引数を与えたことになる。
```

<interactive>:1:0:

No instance for (Integral (a -> a -> a))

arising from a use of `div' at <interactive>:1:0-14

Possible fix:

add an instance declaration for (Integral (a -> a -> a))

In the expression: div div 100 5 5

In the definition of `it': it = div div 100 5 5

```
Prelude> 1 + 2 -- これは普通の式
```

3

```
Prelude> (+) 1 2 -- 括弧で演算子を括ると、前置記法が使える。
```

3

```
Prelude> 10 `div` 2 -- 逆に「`」*7で関数を括ると、中置記法が使える。
```

5

途中やたらけったいなエラーメッセージが出てますが、これを全部理解するには色々知識がいるので今は分からなくても大丈夫です。とりあえず、

- 数式はそのまま入力することができる
- 関数を使うときは括弧もカンマも使わずに引数を並べる
- 演算子は丸括弧で括ると関数のように使える
- 関数はバッククオートで括ると演算子のように使える

というのが、ここまでのおさらいです。

後者二つはそこまで重要なことでもないので、特に深い理解が必要というわけではありません。のですが、後々「スマート」な書き方を求めるに使うことになるので一応載せておきました。

\*6 関数呼び出し: 呼び出し: などと書くと偉い人に怒られそうだ…。「関数適用」といいます。

\*7 バッククオート: 日本語キーボードでは Shift 押しながら@キーを押すと入力できる。Shift+7 のシングルクオートとは別物なので注意。

## ハジメル Haskell

### ——関数イロイロ——

まだまだ基本的なことが続きます。お付き合いください。

+ (加算)、\* (乗算)などといった演算子や div (整数同士の除算)以外の演算子・関数、あるいは定数などを紹介していきます。

```
Prelude> 2 ^ 8 -- ハット(^)はべき乗を表します。この場合28の意。
256
Prelude> 2 ^^ 8 -- 実はべき乗は3種類あって、
256.0
Prelude> 2 ** 8 -- それぞれ微妙に型が違うのですが、
256.0
Prelude> 2 ** (0.5) -- 指数部分に非整数を使えるのは**だけだつたりします。
1.4142135623730951
Prelude> True // False -- True, Falseと言ったBool型の定数や、論理演算もあります。
True
Prelude> True && (not False) -- OR(||), AND(&&), NOT(not)など。
True
Prelude> 1 == 2 -- 等しいかどうかは==。C言語なんかと同じですが、
False
Prelude> 1 /= 2 -- 等しくないかどうかは!=ではなく/=なので注意。
True
Prelude> 1 != 2 -- こんな演算子はHaskellにはない(定義することはできますが…)。
<interactive>:1:2: Not in scope: `!='
Prelude> 1 >= 2 -- その他、大小比較等はC言語と同じです
False
Prelude> mod 10 3 -- divが除算の商をだすなら、modは余をだします。
1
Prelude> 13 `mod` 5 -- こうしても可。
3
Prelude> log 10 -- 自然対数とか、
2.302585092994046
Prelude> exp 10 -- ネーピア数(e)のべき乗などもあります。
22026.465794806718
Prelude> logBase 10 100 -- 常用対数ならlogBaseという関数が使えます。
2
Prelude> sqrt 2 -- 平方根はsqrtで求められますが、
1.4142135623730951
Prelude> 2 ** (1/3) -- **を使えば任意のべき乗根をだせます。
1.2599210498948732
```

こんなそなです。

結構クセの強い部分・弱い部分ありますが、いかがでしょうか？

ちなみに、"--"はC言語で言う"//"で、それ以降をコメントとみなす記号です。

複数行、あるいは一部分だけコメントアウトしたい場合、つまりC言語で言う「/\* some

`text */` を使いたい場合には、「`{- some text -}`」を使います。

## ——文字・モジュール——

なんか基本的な部分が多いですね…。しばしお付き合い願いたい。

C 言語では文字(`char`)と数字は同根というか、結局は同じものだったりするのですが、Haskell では文字と数字は明確に区別されます。

なので、文字と数字を比較したり、ということは許されません。

尤も、文字と数字で相互に変換する方法があるので特に困ることは無いと思います。

Prelude> `'a'` -- 文字はシングルクオートで囲む、というのはC言語同様です。

`'a'`

Prelude> `'a' < 'b'` -- 文字同士の比較はできますが、

`True`

Prelude> `'a' < 98` -- 文字と数字の比較はできません。

<interactive>:1:6:

No instance for (Num Char)

arising from the literal '98' at <interactive>:1:6-7

Possible fix: add an instance declaration for (Num Char)

In the second argument of `(<)`, namely '98'

In the expression: `'b' < 98`

In the definition of `it': it = `'b' < 98`

文字と数字を相互に変換する方法ですが、実はモジュールと呼ばれるものをロードする必要があります。C 言語でいうところの"`#include`"だと思っていただければいいと思います。

文字と数字の相互変換を行う関数は、Char モジュールに入っていますが、これは"`:module Char`"とすることでロードすることができます。

Prelude> `:module Char`

Prelude Char>

プロンプトに `Char` が追加されました。これは `Char` モジュールが読み込まれていることを示します。

勘のいい方は気がついたかもしれません、実はプロンプトに表示されていた `Prelude` もモジュールの名前だったのです。これまで説明してきた関数・演算子等はこの `Prelude` モジュールによって提供されていたものなのです。この `Prelude` には色々な関数が用意されているので、もし興味があれば調べてみるといいでしょう。

さて、コロン(:)から始まる行、実は Haskell の文法ではなくインタプリタへ指示を与えるコマンドとなります。

なので、インタプリタ以外ではこの書き方をすることができません。インタプリタ以外では"`import Char`"と書くことで `Char` モジュールをロードすることができます。というか、今気がついたのですが、インタプリタでも `import` が使えます。

ちなみに、ですが、"`:module`"は"`:m`"と省略することができます。あるいは"`:mo`"でもできますし、"`:mod`"もいけます。要は他のコマンドと区別できるならいくらでも略せる、ということです。

## ハジメル Haskell

あと、すっかり忘れていたのですが、インタプリタの終了方法です。

これまたコロンから始まるコマンドで":quit"<sup>\*8</sup>と入力することで終了できます。

某エディタ<sup>\*9</sup>みたいな終了方法ですが、GHCi は<C-c>や<C-d>でも終了できますので、まあ、説明がなくてもどうにかなるとは思います。

Char モジュールを読み込めたら、さっそく Char モジュールの中の関数を使ってみましょう。

```
Prelude> import Char
Prelude Char> chr 97 -- 文字コードが97の文字を得る。
'a'
Prelude Char> ord 'A' -- 'A'の文字コードを得る。
65
```

なんとなく VB<sup>\*10</sup> を彷彿とさせる命名ですが、こうなると気になることができます。

「Char モジュールにはあとどんな関数があるんだ?」と。

そういう場合には":browse Char"<sup>\*11</sup> とすることで、Char モジュールの関数などを列挙することができます。

```
Prelude Char> :browse Char
data Char = GHC.Base.C# GHC.Prim.Char#
type String = [Char]
chr :: Int -> Char
digitToInt :: Char -> Int
intToDigit :: Int -> Char
isAlpha :: Char -> Bool
isAlphaNum :: Char -> Bool
isAscii :: Char -> Bool
isControl :: Char -> Bool
isDigit :: Char -> Bool
isHexDigit :: Char -> Bool
isLatin1 :: Char -> Bool
isLower :: Char -> Bool
isOctDigit :: Char -> Bool
isPrint :: Char -> Bool
isSpace :: Char -> Bool
isUpper :: Char -> Bool
lexLitChar :: ReadS String
ord :: Char -> Int
```

\*8 quit: ":q"と省略できます

\*9 某エディタ: vim

\*10 VB: Visual Basic のこと。一時期筆者のソウルメイトだった言語。

\*11 browse: module と同様略すことができるのだが、Ver.6.8 から:break という先頭二文字が一致するコマンドができて、:bro まで打たないと:break になってしまふ

```
readLitChar :: ReadS Char
showLitChar :: Char -> ShowS
toLower :: Char -> Char
toUpper :: Char -> Char
```

なにやらコロンが二つ並んで、その後ろに Char やら矢印やらが並んでいます。実はこれは関数の型を示すのですが、説明が長くなるので後回し。

まあ、直感でなんとなく<sup>\*12</sup> 分かると思います。Char を Bool 型に変えるとか。

"is ~"とか"to ~"などは C 言語にもあるような関数ばかりなので、わざわざ例を挙げる必要もないと思われますので、さっさと文字列の話に入りましょう。

## ——文字列・リスト——

やや高度なことが出てきますのでココロシテ。

まずは文字列の話。使うだけなら簡単です。

```
Prelude> "a" -- 文字列はダブルクオートで囲む、というのはC言語同様です。
"a"
Prelude> "abc" ++ "def" -- 文字列の連結も楽々。
"abcdef"
Prelude> "abc" == "abc" -- 比較も楽々。
True
```

さて、C 言語では文字列は文字の配列でした。

Haskell でも同様に文字の集まりが文字列になるのですが、配列ではなくリストというものを使います。関数型言語の多くはこのリストという構造を持っており、かなり重要な概念であったりもします。

```
Prelude> [1,2,3] -- 大括弧で括ってカンマをで区切ればリストになります。
[1,2,3]
Prelude> [True,False] -- 数字でもBoolでも何でもリストにできます。
[True,False]
Prelude> [1,True] -- ただし、異なる型をリストで並べることはできません。
<interactive>:1:1:
    No instance for (Num Bool)
      arising from the literal `1' at <interactive>:1:1
    Possible fix: add an instance declaration for (Num Bool)
    In the expression: 1
    In the expression: [1, True]
    In the definition of `it': it = [1, True]
Prelude> [] -- 空っぽのリストもあります。
[]
Prelude> [[1,2,3],[4,5,6]] -- 「数字のリスト」のリスト、なんてことも。
```

<sup>\*12</sup> 直観でなんとなく：なんともナンセンスな文章だ。それはともかく、実は Char 以外のモジュールだともっと複雑なものが多いので直観で分からぬものの方が多いです。

## ハジメル Haskell

```
[[1,2,3],[4,5,6]]  
Prelude> [[1,2,3],1,2,3] -- 「数字のリスト」と「数字」などはダメです。  
<interactive>:1:13:  
  No instance for (Num [t])  
    arising from the literal `3' at <interactive>:1:13  
  Possible fix: add an instance declaration for (Num [t])  
  In the expression: 3  
  In the expression: [[1, 2, 3], 1, 2, 3]  
  In the definition of `it': it = [[1, 2, ....], 1, 2, ....]
```

いい感じにゲシュタルト崩壊<sup>\*13</sup>してきたところで、文字列の例。

```
Prelude> ['a','b','c'] -- 「Charのリスト」と文字列は同じもの。  
"abc"  
Prelude> ['a','b','c'] == "abc" -- だからこんなこともできます。  
True
```

まあ、だから何？という感じになりそうですが、文字列はともかく、リストはかなり大事です。

ということで次章でさらに掘り下げます。

## ——リストの続き・リストの処理・型——

あれ、この章詰め込み過ぎたかな？

まず、リストとは何か、なのですが、正確に言うのはかなり困難なのでサラッと。

```
Prelude> [] -- リストはすべてこれからハジマります。これは「空リスト」。  
[]  
Prelude> 3:[] -- コロンは演算子。「何か」とリストをくっつけるものなのです。  
[3]  
Prelude> 2:[3] -- 「空リスト」に3をくっつけて、2をくっつけて…  
[2,3]  
Prelude> 1:[2,3] -- リストをどんどん成長させることができます。  
[1,2,3]  
Prelude> 1:(2:(3:[])) -- というか、これがリストの「ホントウ」の姿なのです。  
[1,2,3]  
Prelude> [1,2,3] -- 大括弧とカンマは糖衣記法*14なのです。  
[1,2,3]
```

抽象構文木云々、遅延評価云々言い出すと長いのでこのあたりで止めておきますが、リストを構成する要素はなんと言っても":"と"[]"。この二つに帰着するのです。

ちなみに、Lispなどの他の言語では":"は"cons"、"[]"は"null"と呼ばれたりもします。筆

\*13 ゲシュタルト崩壊: 同じ文字をずっと見続けると、間違っているように見えてしまう現象。詳しくは各自でしらべてちょ

\*14 糖衣記法: 英語で言えばシンタックスシュガー。これが多いほど「甘い」言語になる

者的には断然 Haskell おすすめ<sup>\*15</sup>なので、":"と"[]"で覚えてもいいと思います。

さて、このリスト。並べるだけじゃ意味が無い。なんらかの処理を加えたいものです。  
ということで、まずは簡単なものから。

```
Prelude> sum [1,2,3] -- まずは総和。俗に言うシグマです。
6
Prelude> sum [1.5,2.7,3.8] -- 足し算できるモノならなんでも受け付けます。
8.0
Prelude> sum ['a','b','c'] -- 足し算できないCharはsumすることができません。
<interactive>:1:0:
    No instance for (Num Char)
      arising from a use of `sum' at <interactive>:1:0-16
    Possible fix: add an instance declaration for (Num Char)
    In the expression: sum ['a', 'b', 'c']
    In the definition of `it': it = sum ['a', 'b', 'c']
Prelude> product [1,2,3] -- productは乗積。すべての積。
6
Prelude> length [1,2,3,4,5] -- lengthはリストの要素の数を返します。
5
Prelude> [div 1 1,div 1 0] -- このリストは要素にエラーを含んでいますが、
[1,*** Exception: divide by zero
Prelude> length [div 1 1,div 1 0] -- lengthはちゃんと使えるのがHaskell風味。
2
Prelude> [1,2,3] ++ [4,5,6] -- さっき出てきた++、実はリストならなんでもok。
[1,2,3,4,5,6]
Prelude> ['a','b','c'] !! 0 -- 配列のようにn番目を取ってくこともできます。
'a'
```

と、まあ、わかりやすそうなものから選んできたのですが、いかがでしょうか？

すこし難易度を上げていきます。map のお話。

```
Prelude> [97,98,99] -- 例えばこの様な「数字のリスト」があったとして、
[97,98,99]
Prelude> chr 97 -- このすべての要素にchr関数を適応したいとする。
'a'
Prelude> chr 98 -- C言語の場合はforループなんかを使いますが、
'b'
Prelude> chr 99 -- Haskellではmapというものを使います。
'c'
Prelude> map chr [97,98,99] -- ね、簡単でしょ？
```

---

\*15 断然 Haskell おすすめ: とか書くと各方面からお叱りの言葉とともに物が飛んできそうだ。あと、その、ギャグなんで間に受けないでくださいね。いろんな意味で

## ハジメル Haskell

“abc”

こんな感じです。

具体的には map に第一引数に適応したい関数を、第二引数にリストを与えるべきです。

関数を引数に渡す、というのは C 言語ではあまりやらない<sup>\*16</sup> ことなので少し奇異に感じるかもしれません。が、Haskell を初め、多くの関数型言語では一般的的、というか、無いとやっていけない基本中の基本です。こういったものを「高階関数」などと呼ぶそうです。

さて、ここいらあたりで「型」の読み方について書いていきたいと思います。

まず、map の型ですが、以下のようになります。

map :: (a -> b) -> [a] -> [b]

"map :: "まではいいとして、その後の a やら b やらが型です。

まず、"(a -> b)"。

これは a を受け取ったら b というものを返す、そういう関数を表します。また、この a とか b は任意の型を受け取ることができる、ということを表します。

先ほどの例では

chr :: Int -> Char

という関数が使われていました。

chr の場合で言えば、a が Int、b が Char となっているわけです。Int を Char に変える、といった感じですね。

次に、"[a]"ですが、これは「a のリスト」を意味します。

大括弧はリストのところで出てきたものですし、直感的にもわかりやすいのではないでしょうか？

さて、ここでも a というアルファベットが出てきているのですが、これは先ほどの"(a -> b)"で a として使われた型と同じでなければいけません。つまり、chr の例では"chr :: Int -> Char"ですから a は Int です。なので、"[a]"は具体的には「Int のリスト」のことをさします。"[b]"に関しても同様です。

さて、最後になりますが、これらが"->"という記号でつながれています。

これは最初の"(a -> b)"と同様に関数を表す記号で、全体では、"(a -> b)"という「関数」、"[a]"という「a のリスト」、の二つの引数を取って、"[b]"という「b のリスト」を返す、ということを意味します。chr の例では、a が Int、b が Char ですから、「Int から Char へ変換する関数」と、「Int のリスト」、この二つを引数に取って<sup>\*17</sup>、「Char のリスト」に変換する関数となります。

\*16 C 言語ではあまりやらない：もちろん qsort など関数を引数に渡す例はいくらでもある

\*17 二つを引数に取って：実は違う。のだが、使う上ではこういう認識で構わないと思う

実際、mapにchrと「Intのリスト」を渡してみると、

```
Prelude Char> map chr [109,97,112]
"map"
```

の様に、文字列、つまり「Charのリスト」が返っていることが分かります。

さて、今度はsumの型を見てみましょう。

mapと違い、引数も一つだけですし簡単そうに思えますが、実はそうでないのがHaskell。何はともあれ実際に型を見てみましょう。

```
sum :: (Num a) => [a] -> a
```

なにやらまた新しい記号が出てきました。

"=>"という記号。ひたすら矢印ばかりで嫌になってきた方もいるでしょうが我慢、我慢。しばらくの辛抱です。とりあえず、この新しく出てきた"=>"とその左側は無視して、"[a] -> a"を見てみましょう。

これは、先ほどのmapで出てきた知識だけで理解することができます。すなわち、なにかの型のaがあり、「aのリスト」を引数に取って、aを返す関数となります。

具体例をみてもその通りあることが分かります。

```
Prelude> sum [1,2,3] -- 1 + 2 + 3
6
Prelude> sum [1.5,4.7,2.8] -- 1.5 + 4.7 + 2.8
9.0
```

たしかに「Intのリスト」を渡せば、Intが返っているように見えますし、「Double(小数)のリスト」を渡せば、Doubleが返ってきてているように見えます。

問題はなさそうですが、"(Num a) =>"は何を意味するのでしょうか？

その疑問をとく鍵は次のエラーの例にあります。

```
Prelude> sum ['x','y','z'] -- だからCharは足し算できないんだってづあ !
<interactive>:1:0:
    No instance for (Num Char)
      arising from a use of `sum' at <interactive>:1:0-8
    Possible fix: add an instance declaration for (Num Char)
    In the expression: sum "abc"
    In the definition of `it': it = sum "xyz"
```

mapの場合、aやbにどんな型を入れてみても動くのですが、実はsumで出てくる'a'には入れられる型の制限があります。

その制限こそが"(Num a) =>"なのです。エラーメッセージをみてもなにやら"Num"の記述がちらほらと。この"Num"は「クラス」というもので、実はかなりの高難易度の概念<sup>\*18</sup>なので、詳しくは書きませんが、ざっくばらんに言えば「およその性質」です。

\*18 高難易度の概念: 単に筆者の説明能力が追いつかないだけ。分かってしまえばそんなに難しいわけでもない

## ハジメル Haskell

Num クラスは、足し算、ないし掛け算、符号計算<sup>\*19</sup>などを「行うことのできる性質」を表し、小数を表す Double(倍精度浮動小数)、Float(单精度浮動小数)や整数を表す Int(範囲の狭い整数。最低でも 29bit 符号付整数)、Integer(範囲の無い整数。世に言う bignum)などがこの Num クラスに属しています。

ところが文字を表す Char はこの Num クラスに属していません。一方で sum は"(Num a => "を使い、a は Num クラスに属していないといけない、というルールを定めているのです。

その結果齟齬が生じたぞ、というのが先ほどのエラー。

"No instance for (Num Char)" というのが「Char は Num のクラスに属していない(インスタンスではない)ぞ」、という意で、"arising from a use of `sum' at <interactive>:1:0-8"は「ファイル<interactive>の 1 行め、0 文字目から 8 文字目の sum ってここでこのエラーは起きてるよ」ということなのですが、今は対話モードで起動しているのでファイル名と行数は特に意味は無いです。

あと、「この関数の型はどうないやねん」という場合には":type"<sup>\*20</sup> を使います。

```
Prelude> :type length
length :: [a] -> Int
```

そんなこんなが分かれば多少はエラーもデバックのヒントになるのではないか、というところで、おさらい。

- Haskell の型は a とか b で任意の型を指す
- 型の中で出てくる a とか b は、その型の中では一意でなければならない
- "->" は関数を示すサイン。一番右にあるのが返り値で、それより左にあるのが全部引数<sup>\*21</sup>
- "=>" を使うことで a やら b に制限を加えることができる。
- chr や ord の様に Char や Int など、型名を決め打ちすることも出来るし、その組み合わせも可。 ("length :: [a] -> Int" など)

型の説明もできたので、次章以降は"[Int]"などの表記で「Int のリスト」などを書いていきます。

\*19 符号計算: 絶対値(abs)や符号取得(signum)など。符号取得は負数なら-1、ゼロなら 0、整数なら+1 を返す関数

\*20 type: 当然":t"に省略可

\*21 引数: 先にも注釈で述べたが厳密には異なる。もう少し正しい定義は(たぶん)次号の「部分適用」の項で。

## ——ソースファイルのカキカタ・コンパイル——

少し横道にそれます。

まあ、理由あっての寄り道なのですが。

今までインタプリタを使ってきましたが、ここではコンパイラを使ってみます。

Haskell の一般的な拡張子は.hs のようです。vim をはじめとしたいくつかのエディタでは .hs ファイルを読ませることで、自動で Haskell 用の拡張機能を読み込みますので、特別事情が無い限りは.hs にしておくといいでしょう。

さて、プログラミング入門の定番、Hello World のソースコードが以下のようになります。

```
main :: IO ()
main = do
    putStrLn "Hello WORD"
    return ()
```

ソースの内容はともかく、コンパイルして実行してみましょう。基本的には gcc と同じなので特に悩むところはないかと思われます。

```
$ ghc hello.hs
$ ./a.out
Hello WORD
$ ghc hello.hs -o hello # helloという実行ファイルを作る
$ ./hello
Hello WORD
$ ghc hello.hs -o hello -Wall # Warningをすべて表示する。
$ # Warningが出ないように書いたので何も出ませんが…
```

一般的にプログラムを作るとときは"-Wall"をつけろ、と言われますが現段階ではつけなくてよいでしょう。Warning にまみれてしまい、エラーが見えなくなってしまう可能性があります。

とくに ghc の出力はかなりの行数の Warning を吐くので、文字通り「埋もれて」しまうこともしばし。

で、コンパイル通らないよ！という方が出てくるかもしれません。

おそらく原因はインデントにあると思われます。というのも実は Haskell はインデント、つまり字下げ、見た目のスタイルがプログラムの挙動に影響を与えるという、トテツモナイ言語だったりなのです。

最後の二行は TAB でインデント<sup>\*22</sup> してください。

---

\*22 TAB でインデント: 8 スペース(もちろん半角で)で 1 タブの代わりにもなりますし、スペースだけでインデントしても問題ありません。ともかく 1 タブ 8 スペースのエディタで見た目の先頭を揃えれば ok です。

## ハジメル Haskell

個人的に Makefile 大好き人間なので、お手軽 Makefile も書いておきます。

```
HC=ghc
HFLAGS=
#HFLAGS=-Wall
##-Wallを有効にする場合はここに

SRCS=$(wildcard *.hs)
#SRCS=hello.hs test.hs ...
##ファイルを手動で選択する時はスペースで区切って並べる
OBJS=$(SRCS:.hs=.o) $(SRCS:.hs=.hi)
BINS=$(SRCS:.hs=)

PHONY=all
all: $(BINS)

%: %.hs
    $(HC) $(HFLAGS) $^ -o $@

PHONY+=clean
clean:
    $(RM) $(BINS) $(OBJS)

.PHONY: $(PHONY)
```

筆者愛用のものぐさ Makefile です。

とりあえずカレントディレクトリの.hs ファイルを片端からコンパイルしていきます。

これを"Makefile"という名前で保存したら、"make"ですべてコンパイル、"make clean"で発生したファイルをすべて削除できるようになります。

ということで、14 ページにしてやっと Hello World ができたのですが、  
このソースコードの説明はしません。

いや、ソースコードの説明は後でします。…多分。

というのも実はこの高々 4 行ほどのソースにはまだやっていない、関数定義のほか、Haskell に於いて最難関とも言われる「モナド」という概念が含まれるので説明は非常に困難なのです。

とりあえず、今はテンプレート的に

```
main :: IO ()
main = do
    print $ {- ここに表示したい式 -}
    return ()
```

とすれば ok です。

"putStrLn"が"print"になってたり "\$"がくっついてますが気にしないでやってください。

で、ソースコードの書き方ですが、実はいくつかのバリエーションがあつたりします。リテレイト形式と呼ばれるものや、Bird Style、あるいはインデントに依存しない書き方など様々ですが、とりあえずはインデント記述を使えばいいと思います。

---

—サイゴニ—

すいません。調子乗りすぎました。

えっと、なんか予想以上に長くなりました。HelloWorld までに実に 15 ページ。しかも HelloWorld の説明してないし…。ついで言えば、説明したいことはまだたくさんある…。

……終わるのかな？ これ。という一抹の不安を残しつつ、続きます。

*To be continued...!!*

# 登山 in 名古屋

文 編集部 dorio、yasuharu

名古屋県は愛知市<sup>\*1</sup>。別名「味噌帝国」とも言われ、スタバの入り込む隙も無い喫茶店王国。「たこ焼きが主食の地域よりも不思議だ」とも言われるこのエリア。ここには、地元の人なら誰でも知っている有名な山があります。その山は非常に険しく、山頂まで登りきった者は勇者と呼ばれます。その名も「喫茶マウンテン」。

### 事のはじまり。

とあるつくばの学園都市。私たちの住むこの街では、能力開発<sup>\*2</sup>を目的とした筑波大学を中心と据えられ、それを取り巻くように研究施設が多く設置されている。住人の多くは大学・研究施設の関係者で占められており、非常に苦々しい若々しい街となっている。そんな街の活力の源となっているのが、つくば三大重食「RanRan・夢屋・クラレット」。これらの店がある事により、つくばは今日も元気に朝を迎えるのだ。

しかし、他の大学ではどうなのか。そんな疑問が浮かんだ時、このような情報が匿名で WORD 編集部に飛び込んできた。

「名古屋には、登山ができる喫茶店があるでよー」

### そして冒険は始まる。

7月某日、JR 名古屋駅。降り立った、WORD 編集部員 dorio・かづきお・yasuharu。JST<sup>\*3</sup> 10 時集合との事であったが、いつも通り TST<sup>\*4</sup>への変換が行われ、結局 JST 11 時の集合完了となった。幸先良いスタートである。

しかし、北関東とも南東北とも言われるつくばから名古屋に降り立った私たちにとって、名古屋の天気はあまりにも過酷であった。じんじんと照らす太陽は体力を削いでゆき、「もうマックで食おうよ」と早くも挫折しそうであった。そんな中、とある編集部員の声とともに事態は一変した。

「うは～児童館だ～子供かわゆすなあ<sup>\*5</sup>」

そうして元気になった部員(約一名)は、サクサクと喫茶マウンテンを見つけたのであった。

---

\*1 間違える人なんて居ないと信じたいですが、本当は愛知県名古屋市ですよ？

\*2 問題解決能力。

\*3 JST (Japan Standard Time)。日本標準時。日本で標準的に用いられている時刻のこと。

\*4 TST (Tsukuba Standard Time)。筑波標準時。筑波大生でもっぱら用いられている標準時のこと。JST - 1 ~ 2 時間と言われている。なお、JST より早く設定されることは決してない。

\*5 子供好きの肩身が狭いご時世ですね。



### とまあ、そんなこんなで

喫茶マウンテンに着いて中に入ると、何やら地元のサチ充いかついお兄様方が、順番待ちコーナーにぎっしり。「うわあ…」という言葉を口に出すことも出来ず、目を合わせないように店内をぐるりと見回してみると、アベックやら学生やらサラリーマンで一杯。繁盛しているいたって普通の喫茶店だった。

「あれ、これ普通過ぎね？ これじゃ、ただの名古屋遠征喫茶店ツアーになっちゃわね？」

そんな事を呟きながら待つこと 10 分。席に案内され、メニューを渡される。結構豊富、というかちょっとした飲み屋並に……

( °Д° )



## 登山 in 名古屋

(つ Δ ⊂) ゴ シ ゴ シ

ロロ抹茶小倉スパ	800
甘口バナナスパ	800
甘口メロンスパ	800
甘口キウイスパ	800
しるこスパ	800
甘口イチゴスパ (生)	800
普通のメニュー……?	

( ; ° Δ ° )

ま、また、まだ慌てる時間じゃない。ほ、ほら、周りを見てみろよ。リーマンやらなんやらわっさり居るだろ？繁盛しているだろ？いたって普通の喫茶店のはずだ。きっと、ネタでメニューの名前を変えているだけだ。落ち着いて、注文をしてやろうじゃないか。

- ・大人のお子様ランチ
- ・納豆ピラフ
- ・甘口バナナスパ
- ・赤のコーラ
- ・青のコーラ
- ・イカスミジュース

と、無難なメニューを適当にチョイスして注文完了。後は待つだけ。10分ほど待っていると、次々と頼んだものが到着。

### ・大人のお子様ランチ

お値段800円。名前からして、お子様ランチの大量バージョンとかそんなものを予想。しかし、現実は、これ。



ミツ〇一 …… ミ〇キーじゃないか！いや、違う！可愛くねえ、全然可愛くねえ！こいつ〇ッキーじゃないか！

でも、可愛くなくとも食えるんだつたらいいか、と自分で自分を納得させて食う。

dorio 「……」

かづきお 「……」

yasuharu 「……」

何だろう、この感覚。大声で「このランチを作ったのは誰じゃー！」って叫びたくなるほどの不味さでもないし、かと言って「ウマイ！」と言いながら食える代物でもない。すごく、負けた感じがする。ライスの部分が、非常に重い。どうやつたらピラフをここまで油っこくできるのか、と言いたくなる程重い。上に乗っているハンバーグやウィンナーは普通だったので、それで納得しておく。

#### ・納豆ピラフ

お値段 750 円。茨城から来たのならこれは外せないだろう、と注文。糸引くピラフとか、納豆臭が酷いのを予想して部員同士で盛り上がる<sup>\*6</sup>。

で、到着！



わかりづらいが、納豆入り。

これはヤバい！ これはヤバい！ と言いながら食ってみる。

dorio 「……ん？ あれ、これ普通すぎね？」

かづきお 「普通だ……」

yasuharu 「普通過ぎて何も言うことがない……」

---

\*6 「キャッキャ♪ ウフフ（はあと」

## 登山 in 名古屋

### ・甘口バナナスパ

お値段 800 円。期待の星。これはヤバいだろ！ ヤバいだろ！ と、先ほどのお子様ランチ・納豆ピラフの地味さを飛ばすようにしゃべる。で、来たのがこれ。



いかにも甘そうなスパゲッティ。

一同「(うわ……普通のナポリタンスパゲッティに、生クリームとバナナ、さくらんぼを乗つけてある……)」

さすがに予想の斜め上。ケチャップ味と生クリームは、さすがに組み合はせきつくね？ と、ちょっと引きながら取り皿に取り、食べ始める。

一同「……これはッ！」

ただのナポリタンとは違っていた。甘い。やたら甘い。

麺の赤さが甘さの秘訣なのか、麺の素材が甘さの決め手なのか。そして、その味わいを花を添える生クリームとバナナ。まるで スワイーツ。でも、スパゲッティ。どこまでも、スパゲッティ。

### ・飲み物

そして、最後にやってきたのが飲み物。普通は飲み物が最初に来るんじやね？ という言葉を飲み込んで笑顔で受け取る。まずは赤と青のコーラ。

かづきお「これは！」

yasuharu「どうやって仕入れるんですかね、これ。」

dorio「特注なのかな。すげー。」

そして飲み比べ。

これコーラ？ どう考へてもソーダ水じゃね？

dorio 「これ、クレーム付けてもいいんじゃね？」  
yasuharu・かづきお 「いや、さすがに……」

そして、最後にイカスミジュース。締めは、地雷臭しかしないこの飲み物で！ 見た目は、確かに「イカスミ」を謳うだけあって、黒い。どんな味がするんだ！とワクワクしながら、レッツドリンク！

……あれ、これ砂糖水の味しかしなくね？

どうやら、このジュースは砂糖水にイカ墨を溶かしただけの飲み物のようで、砂糖水と同じ味の黒い飲み物としか表現が出来ない。

かづきお 「いや、これ違う意味で地雷だよね。」  
dorio・yasuharu 「うん……」

### 食べ終えて

食べ終えて店を出る頃には、編集部員一同のテンションは低く、どんよりとした雰囲気であった。

dorio 「……いや、きっともっと凄いのがあるんだよ。  
あれだけメニューが沢山あるんだし。またトライしてみようぜ！」  
かづきお 「え、また行くのはちょっと……」  
yasuharu 「いやあ……ねえ……」

そんなこんなで、重い腹を抱えてしょんぼり帰路につく一同であった。



登山終了。

山なし落ちなし意味なし。

## 筑波大学周辺での暴力・恐喝事件について

文 編集部 goth / 協力 三嶋

### ・事件について

既にご存知ではあると思われますが、現在、筑波大学周辺では暴力・恐喝事件が相次いでいます。犯行の手口としては、スクーターに二人乗りして近づいてきて、いきなり暴力を振るったり、人気のないところへ連れ込んで恐喝するなど、様々です。犯人はいまだ捕まっておらず、依然、事件に巻き込まれる可能性はゼロではありません。

この様な事件に巻き込まれないためにも、各自、防犯意識を持って行動してください。

### ・事件に巻き込まれないためにも

今のところ事件は深夜に起きているようです。冬至に向かい日も短くなっていますので、なるべく早い時間の帰宅を心がけ、深夜の外出を控えるようにしましょう。

どうしても深夜に外出しなければならない場合は、なるべく人通りの多い道を選ぶようにしましょう。特につくばは街灯が無い少ない道や、茂みなどの遮蔽物の多い道が多くあります。その様な場所では犯人に身を隠す場所を与えててしまうなど、事件に巻き込まれる可能性が大きくなります。また、もしできるのならば一人で外出をせずに、なるべく複数人で行動するようにしましょう。

### ・事件に巻き込まれたら…

出会い頭に暴行をうける、などのような場合では対処することはできませんが、恐喝などの場合はその後の対処によっては身を守ることができます。

大原則は犯人を刺激しないことです。たとえ運動部に属していて体力に自身があったとしても、犯人がナイフなどの凶器を持っていた場合、抵抗することでより大きな被害を出してしまう可能性があります。むしろ、犯人は凶器を持っている、と思っていたほうがよいでしょう。恐喝犯に唯々諾々と従うのは許せない、と思うことでしょうが、命よりも大事なものはありません。

もちろん、犯人をそのまま野放しにする手はありません。事件後には必ず警察・大学等の機関に通報し、少しでも多く犯人逮捕につながる情報を伝えましょう。

防犯ブザーは非常に有効な手の一つです。犯人は多くの人間に目撃されることを嫌います。よく、最近は防犯ブザーを鳴らしても誰も来てくれない、とは言われますが、大きな音を鳴らすことは犯人への牽制にも繋がります。事件に巻き込まれた時、人は意外に大声を出せないものです。最近は様々な種類の防犯ブザーがありますので、是非一度検討してみてください。

以上いくつか挙げてきましたが、何よりも大切なこと、それは

### 「自分は無関係」「自分は大丈夫」という楽観視をせず、常に防犯意識を持つこと

それが一番です。

皆様の大学生活が、どうか穏便で、楽しく、実りあるものであることをお祈りします。

編集部一同

# 情報科学類誌

# WORD

WORDの原価は4兆ジンバブエドル号

発行者

情報科学類長

編集長

柴田 泰晴

製作・編集

筑波大学情報学群  
情報科学類WORD編集部  
(第三エリアC棟212室)

印刷

情報科学類印刷室

2008年11月

初版第一刷発行