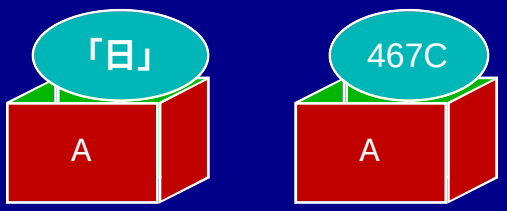
文字データの扱い方

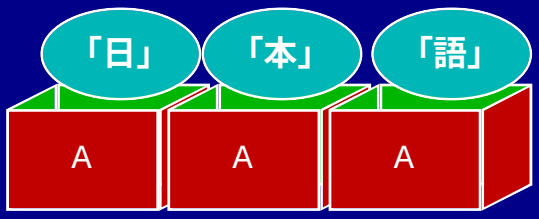
使用する可能性のある文字の集まり(「文字集合」)に番号を付け(「文字符号化」)整数型変数と同様に取扱う(「符号化文字集合」)

文字型変数



文字列

文字型変数の配列



文字集合

使用する文字を集めたもの

- 英文字 a b c d e f ...
- 数字 0 1 2 3 4 ...
- 記号 ! @ # $ % ^ ...
- ひらがな、カタカナ、漢字

どの文字を文字集合に含めるか?

文字集合

どの文字を文字集合に含めるか?

例

文字符号化法

文字集合の要素に整数を一対一に対応させる

- 固定長
  - 7ビット
  - 8ビット(16ビット・32ビット)
- 可変長
- 文字集合の切り替え

文字の表示(書体)

文字と一対一もしくは一対多で対応する図案を必要なだけ用意する

書体(フォント)がなければ表示はされない

半角?全角?

異字体(旧字体)

符号化文字集合

EBCDICコード

ASCIIコード

JISコード(JIS X 0201 (JIS8))

JISコード(JISX0208,iso2022jp)

シフトJISコード(MS-Kanji)

EUCコード(日本語EUCコード)

Unicode

EBCDICコード

IBM製の汎用機で使用された
文字コードと内部表現の相性が良い

過去のもの

ASCIIコード

基本文字コード
JIS7との違いに注意

コンソールアプリケーションではこれのみを使用する

JISコード(JIS X 0201 (JIS8))

カタカナを含む最古の日本語環境
漢字などの半分、英文字と同じ幅で表示された歴史から「半角カナ」と呼ばれた

使ってはいけない

JISコード(JISX0208,iso2022jp)

汎用的な日本語コード
文字集合拡張、多言語拡張も可能
電子メールで使用する

制御文字が必要

シフトJISコード(MS-Kanji)

MS-DOS/Windows Apple Macintosh などで使用された
制御文字無しで日本語使用が可能

拡張性なし

EUCコード(日本語EUCコード)

UNIXベースの計算機で使用された
JIS7の8ビット版
制御文字不要、拡張性あり

ASCII + 一つの言語のみ

Unicode

世界中の文字を一つの集合として扱うことを目的にした
制御文字不要
過去との互換性なし
多くの規格の乱立
将来性あり

ASCII符号化文字集合

「A」= 41H

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	制御文字															
1																
2	■	!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	□

ASCII符号化文字集合

英文字・数字・記号のみ(95文字)
7ビット(128文字)で表現可能

```
2 ■ ! " # $ % & ' ( ) * + , - . /
3 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
4 @ A B C D E F G H I J K L M N O
5 P Q R S T U V W X Y Z [ \ ] ^ _
6 ‘ a b c d e f g h i j k l m n o
7 p q r s t u v w x y z { | } ~ □
```

JIS C 6220

英数記号+カタカナ(約160文字)

```
  0 1 2 3 4 5 6 7 8 9 A B C D E F
0
1   制御文字
2 ■ ! " # $ % & ' ( ) * + , - . /
3 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
4 @ A B C D E F G H I J K L M N O
5 P Q R S T U V W X Y Z [ ¥ ] ^ _
6 ‘ a b c d e f g h i j k l m n o
7 p q r s t u v w x y z { | } ¯ □
```

JIS C 6220

```
  0 1 2 3 4 5 6 7 8 9 A B C D E F
0
1
2   ! " # $ % & ' ( ) * + , - . /
3 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
4 @ A B C D E F G H I J K L M N O
5 P Q R S T U V W X Y Z [ ¥ ] ^ _
6 ‘ a b c d e f g h i j k l m n o
7 p q r s t u v w x y z { | } ¯
8
9
A   カタカナ
B
C
D
E
F
```

JIS X 0201(JIS8)

英数記号+カタカナ(約160文字)
8ビットを使用
7ビットの部分はASCIIと共通
(例外)「\」と「¥」、「~」と「¯」

JIS X 0208 (iso-2022-jp)

英数記号
ひらがな、カタカナ、漢字
14ビット(7ビット+7ビット)
JIS X 0208の文字集合の各バイトを0x21〜0x7Eに割り当てた7ビット符号方式
9216文字(6×16×6×16)使用可能
制御文字で切り替えて使用する

JIS X 0208 (iso-2022-jp)

制御文字で切り替えて使用する

3文字のシーケンス	切り替えられる文字コード
[ESC] (B	ASCII
[ESC] (J	JISローマ字
[ESC] $ @	JIS-1978
[ESC] $ B	JIS-1983

シフトJIS (MS kanji)

JIS X 0201で未定義領域になっている部分を使ってJIS X 0208を表現する
エスケープシーケンスなしで1バイト文字と2バイト文字を共存できる
16ビット(0x81〜0x9F、0xE0〜0xEF) + (0x40〜0x7E、0x80〜0xFC)

日本語EUCコード

ASCIIに相当する部分は7ビット
0xA0〜0xFFの範囲の文字は、2バイト(16ビット)でJIS X 0208
JISコードの8ビット目を1にしたもの

文字符号化の問題点

文字集合の切り替え

- 制御文字を使う(iso-2022-jpなど)
  - 文字数とバイト数が対応しない
  - 表現が一意的でない
  - 拡張が容易
- 固定長符号化(シフトJISなど)
  - 表現できる文字数が少ない
  - 多言語化、拡張が不可能

Unicode

単一の文字集合

- 16ビット/32ビットを使い文字コードの切り替えなしに各国の文字を扱う
- UTF-8
- UTF-7
- UCS-2

Unicode

UTF-8

- ASCII部分は従来どおり1バイト、他の文字を2/3バイトで符号化(Javaの文字コード)

UTF-7

- ASCII部分以外の文字を、Base64の変形版を使って7ビットで符号化

UCS-2

- ISO/IEC 10646 (Unicode文字集合)を2バイト(16ビット)で符号化

JISコード(iso-2022-jp)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3020		亜	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥
3030	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
3040	粟	袷	安	庵	按	暗	案	闇	鞍	杏	以	伊	位	依	偉	囲
3050	夷	委	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	緯	胃
3060	萎	衣	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸
3070	稲	茨	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭	
3120		院	陰	隠	韻	吋	右	宇	烏	羽	迂	雨	卯	鵜	窺	丑
3130	碓	臼	渦	嘘	唄	欝	蔚	鰻	姥	厩	浦	瓜	閏	噂	云	運
3140	雲	荏	餌	叡	営	嬰	影	映	曳	栄	永	泳	洩	瑛	盈	穎

シフトJISコード

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8890																亜
88A0	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥	旭	葦
88B0	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或	粟	袷
88C0	安	庵	按	暗	案	闇	鞍	杏	以	伊	位	依	偉	囲	夷	委
88D0	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	緯	胃	萎	衣
88E0	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸	稲	茨
88F0	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭			
8940	院	陰	隠	韻	吋	右	宇	烏	羽	迂	雨	卯	鵜	窺	丑	碓

Unicode (UCS-2)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4E60			𠂉													
4E70		乱		乳											乾	亂
4E80	亀		亂			亅	了		予	争	事	事	二	亍	于	
4E90		云	互		五	井	亖		亘	亙		些	亜		亞	亟
4EA0	亠	亡	亢		交	亥	亦		亨			享	京	亭	亮	
4EB0	京			亳			亶			亹	人	亻				
4EC0	什	仁	仂		仄		仆	仇			今	介		仍	从	仏
4ED0	仐				仔	仕	他	仗	付	仙				仝	仞	仟
4EE0	仟	仡		代	令	以								仞	仮	
4EF0	仰		仲				件	价				任	任	份		仿

文字化け?

文字集合は正しいか?

- JISX0208にない文字は送れない
- 機種異存文字、「正しい森鴎外」

文字符号化は正しいか?

- 電子メールは7ビット
- 日本語はiso-2022-jpのみ

フォントはあるか?

配列の宣言

配列を使用するためには変数宣言と同様に配列の宣言を行う

- 配列の名前
- 配列の要素の型
- 配列の添字の範囲

配列の宣言と参照(Pascal)

宣言文

var **配列名**: array [**添字範囲**] of **要素の型**
添字範囲　[**最小添字**..**最大添字**]
二次元配列　[**最小**..**最大**, **最小**..**最大**]

要素の参照

配列名[**添字**(**整数値**)]
配列名[**添字**(**整数値**), **添え字**(**整数値**)]

配列の宣言と参照(Pascal)

例

var a: array [1..100] of integer;
a[1], a[2], a[3], …, a[100]

var x: array [0..31, 0..31] of real;
x[0,0], x[0,1], x[0, 2], …, x[0,31],
x[1,0], x[1,1], x[1, 2], …, x[1,31],
…,
x[31,0], x[31,1], x[31,2], …, x[31,31]

配列の宣言と参照(FORTRAN)

宣言文

要素の型 配列名
DIMENSION **配列名**(**添字範囲**)
添字範囲　(**最小添字** : **最大添字**)
二次元配列　(**最小** : **最大**,　**最小** : **最大**)

要素の参照

配列名(**添字**(**整数値**))
配列名(**添字**(**整数値**), **添え字**(**整数値**))

配列の宣言と参照(FORTRAN)

例

INTERGER A
DIMENSION A(100)
a[1], a[2], a[3], …, a[100]
DIMENSION X (0:31, 0:31)
X(0,0), X(0,1), X(0, 2), …,X(0,31),
X(1,0), X(1,1), X(1, 2), …, X(1,31),
…,
X(31,0), X(31,1),X(31,2), …,X(31,31)

配列の宣言と参照(c言語)

宣言文

要素の型 配列名[**要素の数**]
添字範囲　[0] **から**[**要素の数**–1]
二次元配列　[**要素の数**][**要素の数**]

要素の参照

配列名[**添字**(**整数値**)]
配列名[**添字**(**整数値**)][**添え字**(**整数値**)]

配列の宣言と参照(c言語)

例

int a[100];
a[0], a[1], a[2], …, a[99]

double x[32][32];
x[0][0], x[0][1], x[0][2], …, x[0][31],
x[1][0], x[1][1], x[1][2], …, x[1][31],
…,
x[31][0], x[31][1], x[31][2], …, x[31][31]

[例題]

点数を読み込んでその平均値を出力するプログラム
ただし、データの個数は既にわかっているものとする

例題(Pascal)

```
program average(input, output);
const n = 10;
var x: array [1..n] of integer;
    i, sum: interger;
    a: real;
begin
  for i:= 1 to n do read(x[i]);
  sum:=0;
  for i:= 1 to n do sum:=sum+x[i];
  a:= sum/n;
  writeln(a)
end.
```

例題(c言語)

```c
#include <stdio.h>
#define N 10
int main(void){
  int x[N];
  int i, sum;
  double a;
  for(i=0;i<N;i++)scanf("%d",&x[i]);
  sum=0;
  for(i=0;i<N;i++) sum+=x[i];
  a=(double) sum/ (double) N;
  printf("%lf\n", a);
  return 0;
}
```

例題(FORTRAN)

```
      PROGRAM AVERAG
      PARAMETER(N=10)
      INTEGER I, X, SUM
      DOUBLE PRECISION A
      DIMENSION X(1:N)
      READ(5, *) (X(I), I=1, N)
      SUM=0
      DO 10 I=1, N
   10 SUM=SUM+X(I)
      A=DFLOAT(SUM)/DFLOAT(N)
      WRITE(6,*) A
      STOP
      END
```

[レポート課題]

学生番号と点数を読み込んで全体の平均点と最高点及びその学生の学生番号を出力するプログラムを作成せよ

ただし、データの個数は既にわかっているものとする

[レポート課題]

締切

12月17日(火曜日)午後6時

返却

1月11日(土曜日)講義時間中

提出場所

いつものところ

注意事項

2度は書かない