情報通信実験2(情報理論) 課題 1

注意事項

- 例えば、学籍番号 17_55555 の学生が問題 1-2 を解答する場合は、17_55555-01-02 というプリフィックスの名前のファイルで提出すること。例：17_55555-01-02.c
- プログラム言語は何を利用してもよいが、実行できない場合には採点できないので、コンパイル方法と実行方法をプログラムの先頭に以下のようにコメントとして明記すること。

  //␣プログラミング言語：C
  //␣コンパイル方法：gcc␣-std=c11␣test.c␣-lm␣-o␣test.out
  //␣実行方法：ターミナル上で test.out を実行

  ␣は半角の空白を表します。プログラムが書かれているファイルがテキストでない場合には、readme.txt に上記の内容を書いて、プログラムと共に提出すること。
- 採点の間違いを避けるためにも、プログラムにはできるだけコメントを残すこと。
- プログラム内で定義した関数については必ず、それがどのような動作をするのか（何を入力として何を返すのか）を自分なりのコメントとして明記すること。コメントがない場合には減点対象となる場合があるので注意すること。
- 学生間で相談してもよいが、他の学生のファイルを写したりコピーしたりしてはいけない。自分なりのコメントを書いて回答すること。
- ☆ の数は難易度を表している。☆ が多いほど難しいので問題に取り組むための目安にするとよい。

  [難易度 ☆]：易しい（作業）
  [難易度 ☆☆]：普通（できる）
  [難易度 ☆☆☆]：少し難しい（できるはず）
  [難易度 ☆☆☆☆]：難しい（できてほしい）
  [難易度 ☆☆☆☆☆]：かなり難しい（できる人はいる）
  [難易度 ☆☆☆☆☆☆]：激むず（できたらすごい）

基本問題

1-1 [難易度 ☆] 講義資料で挙げた以外の可逆圧縮の規格を 3 つ挙げて主な用途を述べよ。

1-2 [難易度 ☆] 講義資料で挙げた以外の非可逆圧縮の規格を 3 つ挙げて主な用途を述べよ。

1-3 [難易度 ☆] 講義資料の ASCII コードの符号語長を答えよ。

1-4 [難易度 ☆] 情報源アルファベット $\{a, b, c, d, e\}$ に対する語頭符号の例を 2 つ挙げよ。

1-5 [難易度 ☆☆] 【inst_enc(a).c, inst_enc(b).c, inst_enc(c).c】

表 1〜3 のそれぞれの語頭符号に対して、任意の長さ (n としよう) の記号列 x_1^n を符号化して符号語列 c_1^n を表示するプログラムを作成して提出せよ。プログラムはそれぞれの語頭符号ごとに作成してよい。また、記号列の長さは高々 30 と仮定してよい。ただし、次の要件を満たすこと。

(a) `symbols>␣` と表示して、標準入力から記号列を入力する。

(b) `codewords:␣`と表示して、続いて符号語列を表示させる。

次の入力に対する実行結果を答えよ。

- 語頭符号 1 :

  `symbols>␣bababaaabaaaaaaabaabbbbbbabaaa`

  `codewords:␣?`

- 語頭符号 2 :

  `symbols>␣caaaaaacaacaaacbaacbbbccabbccb`

  `codewords:␣?`

- 語頭符号 3 :

  `symbols>␣abcabadcbddbadacbbaabbaccbacdd`

  `codewords:␣?`

表 1　語頭符号 1

記号	符号語
a	0
b	1

表 2　語頭符号 2

記号	符号語
a	1
b	01
c	001

表 3　語頭符号 3

記号	符号語
a	0
b	10
c	110
d	111

語頭符号 1 の動作例：

```
symbols> ababab
codewords: 010101
```

語頭符号 2 の動作例：

```
symbols> ababab
codewords: 010101
```

語頭符号 3 の動作例：

```
symbols> ababab
codewords: 010101
```

1-6 [難易度 ☆☆] 【inst_dec(a).c, inst_dec(b).c, inst_dec(c).c】

- 表 1〜3 の語頭符号に対して、任意の長さの符号語列を復号化して記号列を表示するプログラムを作成して提出せよ。プログラムはそれぞれの語頭符号ごとに作成してよい。また、記号列の長さは高々 30 と仮定してよい。ただし、次の要件を満たすこと。
  (a) 標準出力において `codewords:␣`と表示して、標準入力から符号語列を受け取ること。
  (b) 標準出力において `symbols:␣` と表示して、記号列を表示させること。
- 次の入力に対する実行結果を答えよ。
  - 語頭符号 1：

    `codewords>␣11111101000100111111111101001`

    `symbols:␣<?>`
  - 語頭符号 2 ：

    `codewords>␣00111001101010011001100100111001`

    `symbols:␣<?>`
  - 語頭符号 3 ：

    `codewords>␣10111101101111111000110110 10`

    `symbols:␣<?>`
- 語頭符号 1 の動作例

  `codewords>␣010101`

  `symbols:␣ababab`
- 語頭符号 2 の動作例

  `codewords>␣101001101001`

  `symbols:␣abcabc`
- 語頭符号 3 の動作例

  `codewords>␣010110111010110111 1`

  `symbols:␣abcdabcd`

1-7 [難易度 ☆☆☆] 【kraft.c】

- クラフトの不等式を満たす符号語長列に対する語頭符号を表示するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  (a) 標準出力において `alphabet␣size:␣` と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `l_i:␣`と表示して、第 i 符号語の符号語長 l_i を標準入力から受け取ること。符号語長は短いものから順に入力されることを仮定してよい。
  (c) 標準出力において、各 $i = 1, 2, 3, \cdots$ について、`cw␣for␣l_i:␣`と表示して、第 i 符号語を表示すること。
- 次の入力に対する実行結果を答えよ。

  `alphabet␣size>␣6`

  `l_1>␣1`

  `l_2>␣2`

  `l_3>␣4`

  `l_4>␣8`

l_5>␣16

l_6>␣32

<?>

- ヒント：サンプルプログラム kraft_sample.c を参考にしてもよい。サンプルプログラムはそのままではコンパイルできないが、プログラムを穴埋めすると動作するようになっている。
- 動作例：

alphabet␣size>␣4

l_1>␣1

l_2>␣2

l_3>␣3

l_4>␣3

cw␣for␣l_1:␣0

cw␣for␣l_2:␣10

cw␣for␣l_3:␣110

cw␣for␣l_4:␣111

1-8 [難易度 ☆☆ (問 7 ができている場合)] [難易度 ☆☆☆ (問 7 ができていない場合)] 【shannon.c】

- 与えられた確率分布 $\{p_i\}_{i=1}^{M}$ に対する Shannon 符号を表示するプログラムを作成して提出せよ。また、与えられた確率分布に対するエントロピーと、Shannon 符号の平均符号語長も出力すること。プログラムは次の要件を満たすこと。
  (a) 標準出力において alphabet␣size:␣ と表示して、情報源アルファベットの要素数 M を標準入力から受け取ること。
  (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において p_i:␣ と表示して、第 i 情報源アルファベット記号の確率を標準入力から受け取ること。確率は大きいものから順に入力されることを仮定してよい。
  (c) 標準出力において、各 $i = 1, 2, 3, \cdots$ について、cw␣for␣p_i:␣ と表示して、確率 p_i で生起する入力記号に対応する符号語を表示すること。
  (d) 標準出力において、entropy:␣と表示して、エントロピーを表示すること。
  (e) 標準出力において、average␣length:␣と表示して、平均符号語長を表示すること。
- 次の入力に対する実行結果を答えよ。

alphabet␣size>␣8

p_1>␣0.261

p_2>␣0.241

p_3>␣0.152

p_4>␣0.131

p_5>␣0.115

p_6>␣0.064

p_7>␣0.034

p_8>␣0.002

<?>

- 動作例：

alphabet␣size>␣4

p_1>␣0.4

p_2>␣0.3

p_3>␣0.2

p_4>␣0.1

cw␣for␣p_1:␣00

cw␣for␣p_2:␣01

cw␣for␣p_3:␣100

cw␣for␣p_4:␣1010

entropy:␣1.846439

average␣length:␣2.400000

1-9 [難易度 ☆☆☆☆] 【huffman.c】

- 問題：与えられた確率分布に対するハフマン符号を表示するプログラムを作成して提出せよ。また、与えられた確率分布に対するエントロピーと、ハフマン符号の平均符号語長、与えられた記号列の符号化、与えられた符号語列の復号化の結果を出力すること。プログラムは次の要件を満たすこと。
  (a) 標準出力において alphabet␣size:␣ と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において symbol_i:␣ と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において p_i:␣ と表示して、情報源アルファベットの symbol_i の確率を標準入力から受け取ること。
  (c) 標準出力において、各 $i = 1, 2, 3, \cdots$ について、cw␣for␣symbol_i:␣ と表示して、記号 symbol_i のための符号語を表示すること。
  (d) 標準出力において、entropy:␣と表示して、エントロピーを表示すること。
  (e) 標準出力において、average␣length:␣と表示して、平均符号語長を表示すること。
  (f) 標準出力において symbols>␣ と表示して、情報源アルファベットの記号からなる記号列を標準入力から受け取ること。
  (g) 標準出力において、encoded:␣と表示して、symbols を符号化した結果を表示すること。
  (h) 標準出力において、codewords>␣と表示して、前項で出力された結果の符号語列を標準入力から受け取ること。
  (i) 標準出力において、decoded:␣と表示して、codewords の符号語列を復号した結果を表示すること。
- 次の入力に対する実行結果を答えよ。入力ファイル n_array_huffman.txt をリダイレクトして入力するとよい。

alphabet␣size>␣27

symbol_1>␣a

p_1>␣0.1571

symbol_2>␣b

p_2>␣0.1524

symbol_3>␣c
p_3>␣0.1386
symbol_4>␣d
p_4>␣0.127
symbol_5>␣e
p_5>␣0.1040
symbol_6>␣f
p_6>␣0.0912
symbol_7>␣g
p_7>␣0.0657
symbol_8>␣h
p_8>␣0.0564
symbol_9>␣i
p_9>␣0.0345
symbol_10>␣j
p_10>␣0.0301
symbol_11>␣k
p_11>␣0.0148
symbol_12>␣l
p_12>␣0.0138
symbol_13>␣m
p_13>␣0.0054
symbol_14>␣n
p_14>␣0.005
symbol_15>␣o
p_15>␣0.0018
symbol_16>␣p
p_16>␣0.0012
symbol_17>␣q
p_17>␣0.0005
symbol_18>␣r
p_18>␣0.0002
symbol_19>␣s
p_19>␣0.0001
symbol_20>␣t
p_20>␣0.00008
symbol_21>␣u
p_21>␣0.00007
symbol_22>␣v
p_22>␣0.00003

```
symbol_23>␣w
p_23>␣0.00001
symbol_24>␣x
p_24>␣0.000004
symbol_25>␣y
p_25>␣0.000003
symbol_26>␣z
p_26>␣0.000002
symbol_27>␣_
p_27>␣0.000001
symbols>␣symbols_are_symbols
encoded:␣<?>
codewords>␣<?>
decoded:␣<?>
```

平均符号語長を Shannon 符号と比較してみるとよい。

- ヒント：サンプルプログラム huffman_sample.c を参考にしてもよい。サンプルプログラムはそのままではコンパイルできないが、プログラムを穴埋めすると動作するようになっている。
- 動作例：

```
alphabet␣size>␣4
symbol_1>␣a
p_1>␣0.4
symbol_2>␣b
p_2>␣0.3
symbol_3>␣c
p_3>␣0.2
symbol_4>␣d
p_4>␣0.1
cw␣for␣a:␣1
cw␣for␣b:␣01
cw␣for␣c:␣000
cw␣for␣d:␣001
entropy:␣1.846439
average␣length:␣1.900000
symbols>␣abcdabcd
encoded:␣101000001101000001
codewords>␣101000001101000001
decoded:␣abcdabcd
```

発展問題

1-10 [難易度 ☆☆☆] 【not_inst_enc.c】

- 表 4 の符号（語頭符号ではない）に対して、任意の長さの符号語列を復号化して記号列を表示するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  （a）標準出力において codewords>␣と表示して、標準入力から符号語列を受け取ること。
  （b）標準出力において symbols:␣ と表示して、記号列を表示させること。
- 次の入力 (ファイル not_inst_enc.txt) に対する実行結果を答えよ。
  codewords>␣10001101011101101000110101000110101110110100011010

表 4　符号

記号	符号語
a	10
b	00
c	11
d	110

- 動作例：
  codewords>␣1100110100011
  symbols:␣cbdabc

1-11 [難易度 ☆☆☆] 【given_inst_enc.cpp】

- 問題 5 のように、符号ごとにプログラムを作成すると、何か別の符号を使いたい場合には、その符号のためのプログラムを新たに作成することになり手間がかかる。ここでは、符号ごとにプログラムを作ることはせず、任意に与えられた語頭符号に対して、任意の長さの記号列を符号化して符号語列を表示するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  （a）標準出力において alphabet␣size>␣ と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  （b）各 $i = 1, 2, 3, \cdots$ について、標準出力において symbol_i>␣ と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において codeword_i>␣ と表示して、情報源アルファベットの symbol_i の符号語を標準入力から受け取ること。
  （c）標準出力において symbols>␣ と表示して、標準入力から記号列を受け取ること。
  （d）標準出力において codewords:␣と表示して、符号語列を表示させること。
- 次の入力に対する実行結果を答えよ。
  alphabet␣size>␣4
  symbol_0>␣a
  codeword_0>␣0011
  symbol_1>␣b
  codeword_1>␣01110
  symbol_2>␣c

codeword_2>␣1

symbol_3>␣d

codeword_3>␣0010

symbols>␣abcabadcbddbadacbbaabbaccbacdd

- 動作例：

alphabet␣size>␣3

symbol_1>␣a

codeword_1>␣1

symbol_2>␣b

codeword_2>␣01

symbol_3>␣c

codeword_3>␣00

symbols>␣abcab

codewords:␣10100101

1-12 [難易度 ☆☆☆☆] 【given_inst_dec.cpp】

- 問題 6 のように、符号ごとにプログラムを作成すると、何か別の符号を使いたい場合には、その符号のためのプログラムを新たに作成することになり手間がかかる。ここでは、符号ごとにプログラムを作ることはせず、任意に与えられた語頭符号に対して、任意の長さの符号語列を復号化して記号列を表示するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  (a) 標準出力において alphabet␣size>␣ と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において symbol_i:␣ と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において codeword_i:␣ と表示して、情報源アルファベットの symbol_i の符号語を標準入力から受け取ること。
  (c) 標準出力において codewords>␣ と表示して、標準入力から符号語列を受け取ること。
  (d) 標準出力において symbols:␣ と表示して、記号列を表示させること。
- 次の入力に対する実行結果を答えよ。

alphabet␣size>␣4

symbol_0>␣a

codeword_0>␣0011

symbol_1>␣b

codeword_1>␣01110

symbol_2>␣c

codeword_2>␣1

symbol_3>␣d

codeword_3>␣0010

codewords>␣00110111000110011001100100111000100010100110111010011011101000

10100110011110011001110111000110010011100111 0

- 動作例：

alphabet␣size>␣3

symbol_1>␣a

codeword_1>␣1

symbol_2>␣b

codeword_2>␣01

symbol_3>␣c

codeword_3>␣00

codewords>␣10100101

symbols:␣abcab

1-13 [難易度 ☆☆☆☆☆] 【n_array_huffman.cpp】

- ハフマン符号の出力記号はビットに限る必要はなく、一般にどんな値でも出力記号として扱うことができる。与えられた確率分布に対して、0, 1, 2 を出力記号とする 3 元のハフマン符号を表示するプログラムを作成して提出せよ。プログラムは次の要件を満たすこと。
  (a) 標準出力において alphabet␣size:␣ と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において symbol_i:␣ と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において p_i:␣ と表示して、情報源アルファベットの symbol_i の確率を標準入力から受け取ること。
  (c) 標準出力において、各 $i = 1, 2, 3, \cdots$ について、cw␣for␣symbol_i:␣ と表示して、記号 symbol_i のための符号語を表示すること。
- 次の入力に対する実行結果を答えよ。入力ファイル alphabet_prob.txt をリダイレクトしてリダイレクトを使って入力するとよい。

alphabet␣size>␣27

symbol_1>␣a

p_1>␣0.1571

symbol_2>␣b

p_2>␣0.1524

symbol_3>␣c

p_3>␣0.1386

symbol_4>␣d

p_4>␣0.127

symbol_5>␣e

p_5>␣0.1040

symbol_6>␣f

p_6>␣0.0912

symbol_7>␣g

p_7>␣0.0657

symbol_8>␣h

p_8>␣0.0564

```
symbol_9>␣i
p_9>␣0.0345
symbol_10>␣j
p_10>␣0.0301
symbol_11>␣k
p_11>␣0.0148
symbol_12>␣l
p_12>␣0.0138
symbol_13>␣m
p_13>␣0.0054
symbol_14>␣n
p_14>␣0.005
symbol_15>␣o
p_15>␣0.0018
symbol_16>␣p
p_16>␣0.0012
symbol_17>␣q
p_17>␣0.0005
symbol_18>␣r
p_18>␣0.0002
symbol_19>␣s
p_19>␣0.0001
symbol_20>␣t
p_20>␣0.00008
symbol_21>␣u
p_21>␣0.00007
symbol_22>␣v
p_22>␣0.00003
symbol_23>␣w
p_23>␣0.00001
symbol_24>␣x
p_24>␣0.000004
symbol_25>␣y
p_25>␣0.000003
symbol_26>␣z
p_26>␣0.000002
symbol_27>␣_
p_27>␣0.000001
<?>
```

- ヒント：「ハフマン符号　ダミー」あるいは「Huffman code dummy」とグーグルしてみよう。

- 動作例：

  alphabet␣size>␣4

  symbol_1>␣a

  p_1>␣0.5

  symbol_2>␣b

  p_2>␣0.3

  symbol_3>␣c

  p_3>␣0.1

  symbol_4>␣d

  p_4>␣0.1

  cw␣for␣a:␣0

  cw␣for␣b:␣1

  cw␣for␣c:␣20

  cw␣for␣d:␣21

1-14 [難易度 ☆☆☆☆☆☆] 【lamp.cpp】

- 問題（ランプの魔人）: 次のゲームを考えよう。
  - 回答者は非負整数 $x \in \{1, 2, \cdots, n\}$ を確率 p_x で選択する。
  - 質問者は n と p_i for $i = 1, 2, \ldots, n$ を知っている。
  - 質問者は、回答者に “はい” か “いいえ” で答えら得る質問を行い、できるだけ少ない質問回数で、回答者が選択した x を推定する。

  できるだけ少ない平均質問回数で良い質問方法を答えよ。また、その方法を実行するプログラムを作成して提出せよ。プログラムは次の要件を満たすこと。

  (a) 標準出力において num␣of␣things>␣と表示して、n を標準入力から受け取ること。

  (b) 各 $i = 1, 2, 3, \cdots, n$ について、標準出力において p_i:␣ と表示して、i 番目のモノの確率を標準入力から受け取ること。

  (c) 標準出力において question:␣と表示して、質問が表示されるようにすること。

  (d) 標準出力において answer:␣と表示して、 y または n を入力することで回答できるようにすること。

  (e) x がわかったら、標準出力において final␣answer:␣と出力して、続いて x を出力する。
- 次の入力に対する実行結果を答えよ。x は適当に選んでよい。

  num␣of␣things>␣15

  p_1>␣0.0001

  p_2>␣0.0009

  p_3>␣0.0056

  p_4>␣0.0222

  p_5>␣0.0611

  p_6>␣0.1222

  p_7>␣0.1833

  p_8>␣0.2092

  p_9>␣0.1833

p_10>␣0.1222

p_11>␣0.0611

p_12>␣0.0222

p_13>␣0.0056

p_14>␣0.0009

p_15>␣0.0001

- ヒント：確率が高い順に、あるモノであるかどうかを聞いていく方法は必ずしも最良ではない。
- 動作例：

num␣of␣things>␣4

p_1>␣0.4

p_2>␣0.3

p_3>␣0.2

p_4>␣0.1

question:␣x は 1 ですか？

answer>␣n

question:␣x は 2 ですか？

answer>␣y

final␣answer:␣2

情報通信実験2(情報理論) 課題 2

基本問題

2-1 [難易度 ☆☆☆☆] 【iid.c】

- 与えられた 1 記号の生起確率 $\{p_i\}_{i=1}^{M}$ に従う無記憶情報源について、記号列の生起確率を出力するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  (a) 標準出力において `alphabet␣size:␣`と表示して、情報源アルファベットの要素数 M を標準入力から受け取ること。
  (b) 標準出力において `length:␣`と表示して、記号列長 n を標準入力から受け取ること。
  (c) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i:␣`と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i:␣`と表示して、第 i 情報源アルファベット記号の確率 p_i を標準入力から受け取ること。
  (d) 長さ n の各記号列 $x_1 \cdots x_n$、例えば $n = 2$, $x_1 \cdots x_n$=`aa` について、標準出力において `P(aa): ` と表示して、記号列 $x_1 \cdots x_n$ の確率を表示させること。
- 次の入力に対する実行結果を答えよ。

```
alphabet size> 3
length> 4
symbol_1> a
p_1> 0.1
symbol_2> b
p_2> 0.2
symbol_3> c
p_3> 0.7
```

- 動作例：

```
alphabet size> 3
length> 2
symbol_1 a
p_1> 0.1
symbol_2> b
p_2> 0.2
symbol_3> c
p_3> 0.7
P(aa): 0.010000
P(ab): 0.020000
P(ac): 0.070000
P(ba): 0.020000
P(bb): 0.040000
```

```
P(bc): 0.140000
P(ca): 0.070000
P(cb): 0.140000
P(cc): 0.490000
```

2-2 [難易度 ☆☆☆☆] 【block_huffman.c】

- $e = 3$ 個の入力記号 $x_1x_2\cdots x_e$ をまとめてハフマン符号化するプログラムを作成して提出せよ。ただし、e 個の記号は確率分布 $\{p_i\}_{i=1}^{M}$ にしたがい統計的に独立して生起するとする。つまり、無記憶情報源から記号列が生起することを仮定する。プログラムは次の要件を満たすこと。
  - (a) 標準出力において `alphabet␣size>␣`と表示して、情報源アルファベットの要素数 M を標準入力から受け取ること。
  - (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i>␣`と表示して、第 i 情報源アルファベット記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i>␣`と表示して、第 i 情報源アルファベット記号の確率 p_i を標準入力から受け取ること。
  - (c) 標準出力において `symbols>␣` と表示して、情報源アルファベットの記号からなる長さ $e \times n$ の入力記号列 $x_1\cdots x_{en}$ を標準入力から受け取ること。
  - (d) 標準出力において、`entropy:␣`と表示して、情報源エントロピーを表示すること。
  - (e) 標準出力において、各記号列 **x** について、`cw for x:` と表示して、記号列 **x** のに対応する符号語を表示すること。
  - (f) 標準出力において、`average␣length:␣`と表示して、単位記号あたりの平均符号語長を表示すること。
  - (g) 標準出力において、`encoded:␣`と表示して、入力記号列を符号化した符号語列 $c_1\cdots c_n$ を表示すること。
  - (h) 標準出力において、`codewords>␣`と表示して、前項の出力の符号語列 $c_1\cdots c_n$ を標準入力から受け取ること。
  - (i) 標準出力において、`decoded:␣`と表示して、符号語列 $c_1\cdots c_n$ を復号した $y_1\cdots y_{en}$ を表示すること。
- 次の入力に対する実行結果を答えよ。

```
alphabet size> 4
symbol_1> a
p_1> 0.1
symbol_2> b
p_2> 0.3
symbol_3> c
p_3> 0.2
symbol_4> d
p_4> 0.4
entropy: <?>
average length: <?>
cw for ...: <?>
```

```
cw for ...: <?>
symbols> abcdaabbccdd
encoded: <?>
codewords> <?>
decoded: <?>
```

- e を大きくするにしたがって、平均符号語長がエントロピーに近づいていくことを確認せよ。
- 動作例：

```
alphabet size> 2
symbol_1> a
p_1> 0.4
symbol_2> b
p_2> 0.6
cw for aaa: 0011
cw for aab: 110
cw for aba: 111
cw for abb: 011
cw for baa: 0010
cw for bab: 010
cw for bba: 000
cw for bbb: 10
entropy: 0.970951
average length: 0.981333
symbols> abaabbaaabbb
encoded: 111011001110
codewords> 111011001110
decoded: abaabbaaabbb
```

2-3 [難易度 ☆☆] 算術符号の復号では記号列長を知っていることを仮定している。これを実現するには例えばガンマ符号と呼ばれる符号を利用すればよい。ガンマ符号をどのように利用すればよいだろうか？ガンマ符号の説明とともに、その具体的な利用方法を説明せよ。

2-4 [難易度 ☆☆☆☆] 【arithmetic_code.c】

- 算術符号を実装するプログラムを作成して提出せよ。ただし、記号列は算術符号が実行できる程度の長さであることを仮定してよい。プログラムは次の要件を満たすこと。
  - (a) 標準出力において `alphabet␣size:␣`と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  - (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i>␣`と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i>␣`と表示して、情報源アルファベットの symbol_i の確率を標準入力から受け取ること。
  - (c) 標準出力において `symbols>␣` と表示して、情報源アルファベットの記号からなる記号列を標準入力から受け取ること。

(d) 標準出力において、`range:␣`と表示して、symbols を符号化した結果の区間の下限と上限を [区間の下限, 区間の上限) という形式で表示すること。

(e) 標準出力において、`encoded>␣`と表示して、symbols を符号化した結果を表示すること。

(f) 標準出力において、`codewords>␣`と表示して、符号語列を標準入力から受け取ること。

(g) 標準出力において、`length>␣`と表示して、符号語列の長さを標準入力から受け取ること。

(h) 標準出力において、`decoded:␣`と表示して、codewords の符号語列を復号した結果を表示すること。

- 次の入力に対する実行結果を答えよ。

```
alphabet size> 4
symbol_1> a
p_1> 0.6
symbol_2> b
p_2> 0.2
symbol_3> c
p_3> 0.1
symbol_4> d
p_4> 0.1
symbols> aadaadaabcccbdbcbacc
<?>
codewords> 00101011001100110011
length> 30
<?>
```

- 動作例：

```
alphabet size> 4
symbol_1> a
p_1> 0.2
symbol_2> b
p_2> 0.2
symbol_2> c
p_2> 0.4
symbol_2> _
p_2> 0.2
symbols> aa_bb_cccc
range: [0.0342175744, 0.0342192128)
encoded: 000010001100001010001
codewords> 0000100010111
length> 5
decoded: aa_bb
```

2-5 [難易度 ☆☆☆☆☆] 【lz78-internal.cpp】

- LZ78 符号を実装するプログラムを作成して提出せよ。ただし、記号は ASCII コードによって表さ

れるとする。中間符号語までを出力すればよいこととする。プログラムは次の要件を満たすこと。

(a) 標準出力において symbols>␣ と表示して、記号列を標準入力から受け取ること。

(b) 標準出力において、encoded:␣と表示して、symbols を符号化した結果を表示すること。

(c) 標準出力において、codewords␣length>␣と表示して、符号語列の長さを標準入力から受け取ること。

(d) 標準出力において、node:␣と表示して、中間符号語の第一要素である節点番号を、symbol:␣と表示して、中間符号語の第二要素である記号をそれぞれ上記の符号語列の長さだけ標準入力から受け取ること。

(e) 標準出力において、decoded:␣と表示して、中間符号語列を復号した結果を表示すること。

(f) 次の入力 (ファイル lz78-internal.txt) に対する実行結果を答えよ。

```
symbols> LZ77_and_LZ78_are_the_two_lossless_data_compression_algorithms
_published_in_papers_by_Abraham_Lempel_and_Jacob_Ziv_in_1977_and_1978.
_They_are_also_known_as_LZ1_and_LZ2_respectively._These_two_algorithms
_form_the_basis_for_many_variations_including_LZW,_LZSS,_LZMA_and_others.
_Besides_their_academic_influence,_these_algorithms_formed_the_basis_of
_several_ubiquitous_compression_schemes,_including_GIF_and_the_DEFLATE
_algorithm_used_in_PNG_and_ZIP.
codewords length> 10
node: 0
symbol: L
node: 0
symbol: Z
node: 0
symbol: 7
node: 3
symbol: _
node: 0
symbol: a
node: 0
symbol: n
node: 0
symbol: d
node: 0
symbol: _
node: 1
symbol: Z
node: 3
symbol: 8
```

- 動作例：

```
symbols: ababbababaa
```

```
encoded: (0, a) (0, b) (1, b) (2, a) (4, b) (1, a)
codewords length: 3
node: 0
symbol: a
node: 0
symbol: b
node: 1
symbol: b
decoded: abab
```

発展問題

2-6 [難易度 ☆☆☆☆] 【markov.c】

- 与えられた 1 記号の生起確率と、1 記号が生起した後に 1 記号が生起する確率（条件付き確率）に従うマルコフ情報源について、記号列の生起確率を出力するプログラムを作成して提出せよ。ただし、次の要件を満たすこと。
  （a）標準出力において `alphabet␣size>␣`と表示して、情報源アルファベットの要素数 M を標準入力から受け取ること。
  （b）標準出力において `length>␣`と表示して、記号列長を標準入力から受け取ること。
  （c）各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i>␣`と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i>␣`と表示して、第 i シンボルの生起確率 p_i を標準入力から受け取ること。
  （d）各 $i = 1, 2, 3, \cdots, M$ と $k = 1, 2, 3, \cdots, M$ について、標準出力において `cp_ik:␣`と表示して、第 i シンボルが生起した後に第 k シンボルが生起する確率（条件付き確率）$p_{k|i}$ を標準入力から受け取ること。
  （e）各記号列 $\mathbf{x}$ について、標準出力において $\mathrm{P}(\mathbf{x})$: と表示して、記号列 $\mathbf{x}$ の確率を表示させること。
- 次の入力に対する実行結果を答えよ。

```
alphabet size> 2
length> 5
symbol_1> a
p_1> 0.2
symbol_2> b
p_2> 0.8
cp_11> 0.2
cp_12> 0.8
cp_21> 0.5
cp_22> 0.5
<?>
```

- 動作例：

```
alphabet size> 2
length> 2
symbol_1> a
p_1> 0.2
symbol_2> b
p_2> 0.8
cp_11> 0.2
cp_12> 0.8
```

```
cp_21> 0
cp_22> 1
P(aa): 0.040000
P(ab): 0.160000
P(ba): 0.000000
P(bb): 0.800000
```

2-7 [難易度 ☆☆☆☆☆] 【block_huffman.c】

- 記号列をまとめてハフマン符号化するプログラムを作成して提出せよ。ただし、記号列は統計的に独立して生起するとする。つまり、無記憶情報源から記号列が生起することを仮定する。プログラムは次の要件を満たすこと。
  - (a) 標準出力において `alphabet␣size>␣`と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  - (b) 標準出力において `length>␣`と表示して、記号列長を標準入力から受け取ること。
  - (c) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i>␣`と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i>␣`と表示して、情報源アルファベットの symbol_i の確率を標準入力から受け取ること。
  - (d) 標準出力において、各記号列 $\mathbf{x}$ について、cw for $\mathbf{x}$: と表示して、記号列 $\mathbf{x}$ のための符号語を表示すること。
  - (e) 標準出力において、`entropy:␣`と表示して、エントロピーを表示すること。
  - (f) 標準出力において、`average␣length:␣`と表示して、単位記号あたりの平均符号語長を表示すること。
  - (g) 標準出力において `symbols>␣` と表示して、情報源アルファベットの記号からなる記号列を標準入力から受け取ること。
  - (h) 標準出力において、`encoded:␣`と表示して、symbols を符号化した結果を表示すること。
  - (i) 標準出力において、`codewords>␣`と表示して、前項の出力の符号語列を標準入力から受け取ること。
  - (j) 標準出力において、`decoded:␣`と表示して、codewords の符号語列を復号した結果を表示すること。
- 次の入力に対する実行結果を答えよ。

```
alphabet size> 2
length> 4
symbol_1> a
p_1> 0.8
symbol_2> b
p_2> 0.2
<?>
symbols> aaababaaaaaaaaaabaaabbaa
encoded: <?>
codewords> <?>
```

```
decoded: <?>
```

平均符号語長を 1 記号の場合のハフマン符号と比較してみるとよい。エントロピーに近づいているだろうか？

また、記号列長を長くしても実行できるかを確認せよ。どこまでの長さであれば実行できるだろうか？

- 動作例：

```
alphabet size> 2
length> 4
symbol_1> a
p_1> 0.4
symbol_2> b
p_2> 0.6
cw for aaaa: 01101
cw for aaab: 00110
cw for aaba: 1101
cw for aabb: 0111
cw for abaa: 01100
cw for abab: 1000
cw for abba: 1010
cw for abbb: 0010
cw for baaa: 00111
cw for baab: 1100
cw for baba: 1001
cw for babb: 0000
cw for bbaa: 1011
cw for bbab: 0001
cw for bbba: 111
cw for bbbb: 010
entropy: 0.970951
average length: 0.981200
symbols> abbbabbbbbbb
encoded: 00100010010
codewords> 00100010010
decoded: abbbabbbbbbb
```

2-8 [難易度 ☆☆☆☆☆☆]

- 問題：算術符号では、区間によって記号列 $\mathbf{x}$ を識別するために、区間の中点を表す $m(\mathbf{x})$ を用いている。これは、$m(\mathbf{x})$ の値がわかれば、その値を含む区間から記号列 $\mathbf{x}$ を一意に特定することができるからである。ところが実際には、$m(\mathbf{x})$ の小数点を $l(\mathbf{x})$ 桁で打ち切った値を用いている。$m(\mathbf{x})$ の代わりにこの値を用いたとしても、記号列 $\mathbf{x}$ が一意に特定できるのはなぜかを説明せよ。
- ヒント：$l(\mathbf{x})$ 桁で打ち切った値が他の記号列が表す区間に侵入しなければ、一意に区間を特定で

きる。

2-9 [難易度 ☆☆☆☆☆☆]

- 問題： 参考資料を元に Jones 符号を実装するプログラムを作成して提出せよ。プログラムは次の要件を満たすこと。
  - (a) 標準出力において `alphabet␣size>␣`と表示して、情報源アルファベットの要素数を標準入力から受け取ること。
  - (b) 各 $i = 1, 2, 3, \cdots$ について、標準出力において `symbol_i>␣`と表示して、情報源アルファベットの各記号を標準入力から受け取ること。ただし、アルファベットの要素は ASCII コードで用いられる記号を仮定してよい。また、標準出力において `p_i>␣`と表示して、情報源アルファベットの symbol_i の確率を標準入力から受け取ること。
  - (c) 標準出力において `symbols>␣` と表示して、情報源アルファベットの記号からなる記号列を標準入力から受け取ること。
  - (d) 標準出力において、`encoded:␣`と表示して、symbols を符号化した結果を表示すること。
  - (e) 標準出力において、`codewords>␣`と表示して、前項で出力した符号語列を標準入力から受け取ること。
  - (f) 標準出力において、`decoded:␣`と表示して、codewords の符号語列を復号した結果を表示すること。
- 次の入力に対する実行結果を答えよ。

```
alphabet size> 4
symbol_1> a
p_1> 0.6
symbol_2> b
p_2> 0.2
symbol_3> c
p_3> 0.1
symbol_4> d
p_4> 0.1
symbols> aadaadaabcccbdbcbacc
<?>
codewords> <?>
<?>
```

- 動作例：

```
alphabet size> 4
symbol_1> a
p_1> 0.2
symbol_2> b
p_2> 0.2
symbol_2> c
p_2> 0.4
symbol_2> _
```

```
p_2> 0.2
symbols> aa_bb_cccc
encoded: 00001000110001010101
codewords> 00001000110001010101
decoded: aa_bb_cccc
```

2-10 [難易度 ☆☆☆☆☆☆] 【lz78.cpp】

- 問題 : LZ78 符号を実装するプログラムを作成して提出せよ。ただし、記号は ASCII コードによって表されるとする。プログラムは次の要件を満たすこと。
  (a) 標準出力において `symbols>␣` と表示して、記号列を標準入力から受け取ること。
  (b) 標準出力において、`encoded:␣`と表示して、symbols を符号化した結果を表示すること。
  (c) 標準出力において、`codewords>␣`と表示して、符号語列を標準入力から受け取ること。
  (d) 標準出力において、`decoded:␣`と表示して、codewords の符号語列を復号した結果を表示すること。
- 次の入力 (ファイル `lz78-internal.txt`) に対する実行結果を答えよ。

```
symbols> LZ77_and_LZ78_are_the_two_lossless_data_compression_algorithms
_published_in_papers_by_Abraham_Lempel_and_Jacob_Ziv_in_1977_and_1978.
_They_are_also_known_as_LZ1_and_LZ2_respectively._These_two_algorithms
_form_the_basis_for_many_variations_including_LZW,_LZSS,_LZMA_and_others.
_Besides_their_academic_influence,_these_algorithms_formed_the_basis_of
_several_ubiquitous_compression_schemes,_including_GIF_and_the_DEFLATE
_algorithm_used_in_PNG_and_ZIP.
<?>
codewords> 0101010000110100000011001010001111001000010111110000110000 1
000011100100110101111100000110001000000110111100000011101000010010111 11
101101101000001101101111011101100101101101101001000000110001100110011 0
110000000011011000010001011111000000110010000000011010011000101110100 1
011001101111000000110111000110011100101010001100011010100110010000011 0
111001000000011100110000000101110
<?>
```

- 動作例 :

```
symbols> ababbababaa
encoded: 01100001001100010010110001010011000011000110001000101100001
codewords> 01100001001100010010110001010011000011000110001000101100001
decoded: ababbababaa
```

- 注意 : ASCII コードを 7 ビットと考えれば、動作例の元の記号列を保存するには 77 ビット必要となる。一方で、符号語は 59 ビットに圧縮されているため、より少ないビット列で記号列が保存できることになる。