Фреймворк PyTorch для разработки искусственных нейронных сетей

Урок 8. Generative Adversarial Networks

Практическое задание

Домашнее задание к уроку 8

Переписать загрузку данных с python функций на Dataset и Dataloader и применить сеть с attention

Выполнил ***Соковнин ИЛ***

B [1]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F

import numpy as np
import pandas as pd
```

B [2]:
```
epochs = 30
num_samples = 10000
data_path = './data/fra-eng/fra.txt'
```

B [3]:
```
class eng_fra_dataset(torch.utils.data.Dataset):
    def __init__(self, file_name=data_path, num_samples=num_samples, nrows=num_samples):

        input_vocab = set()
        output_vocab = set()

        print(f'Загружаем файл: {file_name}')
        df = pd.read_csv(data_path, sep = '\t', header = None, nrows=num_samples)
        df.rename(columns = {0 : 'eng', 1: 'fra'}, inplace = True)
        # df = df[['eng', 'fra']]

        input_texts = list(df['eng'])
        target_texts = list(df['fra'])
        texts = df[['eng','fra']].values.tolist()

        for input_text in input_texts:
            for word in input_text.split():
                input_vocab.add(word.strip())

        for target_text in target_texts:
            for word in target_text.split():
                output_vocab.add(word.strip())

        output_vocab = set(list(output_vocab))
        input_vocab2index = {word: i+2 for i, word in enumerate(input_vocab)}
        output_vocab2index = {word: i+2 for i, word in enumerate(output_vocab)}

        self.texts = texts
        self.input_texts = input_texts
        self.target_texts = target_texts
        self.input_vocab2index = input_vocab2index
        self.output_vocab2index = output_vocab2index

        print('Файл {file_name} загружен')


    def __len__(self):
        return len(self.input_vocab)

    def __getitem__(self, idx):
        return self.texts[idx]

    def print_ds(self):
        for i in self.input_vocab2index:
            if self.input_vocab2index[i] < 12:
                print(i, self.input_vocab2index[i])
```

B [4]:
```
def indexesFromSentence(sentence, vocab):
    return [vocab.get(word.strip(), 0) for word in sentence.split(' ')]

def tensorFromSentence(sentence, vocab):
    indexes = indexesFromSentence(sentence, vocab)
    indexes.append(1)
    return torch.tensor(indexes, dtype=torch.long).view(-1, 1)


def tensorsFromSent(input_sentences, output_sentences, input_vocab2index, output_vocab2index):
    input_tensor = tensorFromSentence(input_sentences, input_vocab2index)
    target_tensor = tensorFromSentence(output_sentences, output_vocab2index)
    return (input_tensor, target_tensor)
```

B [5]:
```
class EncoderRNN(nn.Module):
    def __init__(self, input_size, hidden_size):
        super(EncoderRNN, self).__init__()
        self.hidden_size = hidden_size

        self.embedding = nn.Embedding(input_size, hidden_size)
        self.gru = nn.GRU(hidden_size, hidden_size)

    def forward(self, input, hidden):
        embedded = self.embedding(input).view(1, 1, -1)
        output = embedded
        output, hidden = self.gru(output, hidden)
        return output, hidden

    def initHidden(self):
        return torch.zeros(1, 1, self.hidden_size)


class AttnDecoderRNN(nn.Module):
    def __init__(self, hidden_size, output_size, dropout_p=0.1, max_length=10):
        super(AttnDecoderRNN, self).__init__()
        self.hidden_size = hidden_size
        self.output_size = output_size
        self.dropout_p = dropout_p
        self.max_length = max_length

        self.embedding = nn.Embedding(self.output_size, self.hidden_size)
        self.attn = nn.Linear(self.hidden_size * 2, self.max_length)
        self.attn_combine = nn.Linear(self.hidden_size * 2, self.hidden_size)
        self.dropout = nn.Dropout(self.dropout_p)
        self.gru = nn.GRU(self.hidden_size, self.hidden_size)
        self.out = nn.Linear(self.hidden_size, self.output_size)

    def forward(self, input, hidden, encoder_outputs):
        embedded = self.embedding(input).view(1, 1, -1)
        embedded = self.dropout(embedded)

        attn_weights = F.softmax(
            self.attn(torch.cat((embedded[0], hidden[0]), 1)), dim=1)
        attn_applied = torch.bmm(attn_weights.unsqueeze(0),
                                 encoder_outputs.unsqueeze(0))

        output = torch.cat((embedded[0], attn_applied[0]), 1)
        output = self.attn_combine(output).unsqueeze(0)

        #output = F.relu(output)
        output, hidden = self.gru(output, hidden)

        output = F.log_softmax(self.out(output[0]), dim=1)
        return output, hidden, attn_weights

    def initHidden(self):
        return torch.zeros(1, 1, self.hidden_size)
```

B [6]:
```
def train(input_tensor, target_tensor, encoder, decoder, encoder_optimizer, decoder_optimizer, criterion, max_length=10)
    encoder_hidden = encoder.initHidden()

    encoder_optimizer.zero_grad()
    decoder_optimizer.zero_grad()

    input_length = input_tensor.size(0)
    target_length = target_tensor.size(0)

    encoder_outputs = torch.zeros(max_length, encoder.hidden_size)

    loss = 0

    for ei in range(input_length):
        encoder_output, encoder_hidden = encoder(
            input_tensor[ei], encoder_hidden)
        encoder_outputs[ei] = encoder_output[0, 0]

    decoder_input = torch.tensor([[0]])

    decoder_hidden = encoder_hidden

    for di in range(target_length):
        decoder_output, decoder_hidden, decoder_attention = decoder(
            decoder_input, decoder_hidden, encoder_outputs)
        topv, topi = decoder_output.topk(1)
        decoder_input = topi.squeeze().detach()  # detach from history as input

        loss += criterion(decoder_output, target_tensor[di])
        if decoder_input.item() == 1:
            break

    loss.backward()

    encoder_optimizer.step()
    decoder_optimizer.step()

    return loss.item() / target_length
```

B [7]:
```
ds = eng_fra_dataset()
# ds.print_ds()
ds[1]
```

```
Загружаем файл: ./data/fra-eng/fra.txt
Файл {file_name} загружен
```

Out[7]:
```
['Hi.', 'Salut !']
```

B [8]:
```
encoder = EncoderRNN(len(ds.input_vocab2index)+2, 30)
attn_decoder1 = AttnDecoderRNN(30, len(ds.output_vocab2index)+2, dropout_p=0.1)

encoder_optimizer = torch.optim.SGD(encoder.parameters(), lr=0.01)
decoder_optimizer = torch.optim.SGD(attn_decoder1.parameters(), lr=0.01)
training_pairs = np.random.randint(0, len(ds.input_texts), size=10000)
criterion = nn.NLLLoss()

print_loss_total = 0

for i in range(1000):
    input_tensor, target_tensor = tensorsFromSent(ds.input_texts[training_pairs[i]], ds.target_texts[training_pairs[i]],
                                                  ds.input_vocab2index, ds.output_vocab2index)

    loss = train(input_tensor, target_tensor, encoder,
                 attn_decoder1, encoder_optimizer, decoder_optimizer, criterion)
    print_loss_total += loss

    print_loss_avg = print_loss_total / 1
    print_loss_total = 0
    print('(%d %d%%) %.4f' % (i, i / 10 * 100, print_loss_avg))
```

```
(0 0%) 8.6540
(1 10%) 8.6432
(2 20%) 8.7114
(3 30%) 8.7780
(4 40%) 8.6563
(5 50%) 8.6907
(6 60%) 8.6709
(7 70%) 8.6171
(8 80%) 8.7057
(9 90%) 8.6936
(10 100%) 8.5683
(11 110%) 8.5469
(12 120%) 8.7139
(13 130%) 8.5683
(14 140%) 8.5551
(15 150%) 8.5791
(16 160%) 8.7090
(17 170%) 8.6746
(18 180%) 8.5785
(19 190%) 8.5545
(20 200%) 8.5442
(21 210%) 8.6220
(22 220%) 8.4640
(23 229%) 8.4238
(24 240%) 8.4129
(25 250%) 8.6544
(26 260%) 8.5125
(27 270%) 8.6281
(28 280%) 8.4944
(29 290%) 8.5364
(30 300%) 4.3646
(31 310%) 2.8673
(32 320%) 3.4719
(33 330%) 5.6825
(34 340%) 6.9881
(35 350%) 3.6989
(36 360%) 5.7402
(37 370%) 8.4526
(38 380%) 6.3972
(39 390%) 2.8499
(40 400%) 5.6791
(41 409%) 3.5371
(42 420%) 6.9641
(43 430%) 3.4441
(44 440%) 2.9019
(45 450%) 3.5218
(46 459%) 8.5855
(47 470%) 3.4459
(48 480%) 3.3896
(49 490%) 5.7799
(50 500%) 8.3430
(51 509%) 4.1402
(52 520%) 8.3600
(53 530%) 3.4874
(54 540%) 3.4993
(55 550%) 5.6532
(56 560%) 8.3228
(57 570%) 8.2857
(58 580%) 3.3908
(59 590%) 3.4761
(60 600%) 2.9122
(61 610%) 5.7576
(62 620%) 2.1255
(63 630%) 2.8752
(64 640%) 3.4388
```

```
(65 650%) 5.7602
(66 660%) 2.8140
(67 670%) 4.3067
(68 680%) 2.7792
(69 690%) 4.2024
(70 700%) 3.4899
(71 710%) 4.3297
(72 720%) 4.2798
(73 730%) 4.1941
(74 740%) 1.7677
(75 750%) 3.3655
(76 760%) 5.6881
(77 770%) 3.4872
(78 780%) 2.8212
(79 790%) 3.4471
(80 800%) 2.8373
(81 810%) 2.8392
(82 819%) 1.4405
(83 830%) 4.3065
(84 840%) 3.4888
(85 850%) 2.1211
(86 860%) 1.7083
(87 869%) 5.7449
(88 880%) 4.3728
(89 890%) 4.3166
(90 900%) 5.6929
(91 910%) 5.7073
(92 919%) 3.3514
(93 930%) 4.2191
(94 940%) 5.8076
(95 950%) 4.3092
(96 960%) 3.3367
(97 969%) 3.4265
(98 980%) 4.1652
(99 990%) 4.3131
(100 1000%) 3.3042
(101 1010%) 4.2407
(102 1019%) 4.3077
(103 1030%) 4.0787
(104 1040%) 3.4712
(105 1050%) 6.5220
(106 1060%) 5.4730
(107 1070%) 4.3562
(108 1080%) 3.3208
(109 1090%) 3.3962
(110 1100%) 2.6828
(111 1110%) 3.4143
(112 1120%) 5.2917
(113 1130%) 3.3450
(114 1140%) 4.3646
(115 1150%) 8.1083
(116 1160%) 3.2043
(117 1170%) 8.2521
(118 1180%) 3.2850
(119 1190%) 6.1555
(120 1200%) 4.1405
(121 1210%) 3.1853
(122 1220%) 8.2854
(123 1230%) 3.4270
(124 1240%) 5.8434
(125 1250%) 3.6965
(126 1260%) 5.1788
(127 1270%) 6.3482
(128 1280%) 5.2209
(129 1290%) 4.3987
(130 1300%) 2.7631
(131 1310%) 4.9685
(132 1320%) 6.4926
(133 1330%) 6.6403
(134 1340%) 2.7461
(135 1350%) 6.4476
(136 1360%) 6.1024
(137 1370%) 2.7978
(138 1380%) 6.4351
(139 1390%) 4.8719
(140 1400%) 6.3706
(141 1410%) 5.2169
(142 1420%) 5.2958
(143 1430%) 8.3757
(144 1440%) 3.7746
(145 1450%) 5.1228
(146 1460%) 6.4771
(147 1470%) 6.2795
(148 1480%) 8.0541
(149 1490%) 4.1945
(150 1500%) 3.2948
(151 1510%) 3.3078
(152 1520%) 3.4067
```

```
(153 1530%) 3.3969
(154 1540%) 4.3285
(155 1550%) 4.1467
(156 1560%) 2.3881
(157 1570%) 6.6833
(158 1580%) 4.2498
(159 1590%) 5.7011
(160 1600%) 6.3458
(161 1610%) 8.0959
(162 1620%) 6.1270
(163 1630%) 4.4043
(164 1639%) 5.9884
(165 1650%) 6.2670
(166 1660%) 5.1088
(167 1670%) 7.8307
(168 1680%) 4.3583
(169 1689%) 5.0198
(170 1700%) 2.5263
(171 1710%) 2.4644
(172 1720%) 3.7579
(173 1730%) 3.8781
(174 1739%) 5.1875
(175 1750%) 5.2423
(176 1760%) 6.4581
(177 1770%) 6.3770
(178 1780%) 4.4186
(179 1789%) 8.1706
(180 1800%) 5.1959
(181 1810%) 6.2181
(182 1820%) 5.3593
(183 1830%) 6.2772
(184 1839%) 3.4379
(185 1850%) 5.4718
(186 1860%) 5.8691
(187 1870%) 4.1194
(188 1880%) 5.9407
(189 1889%) 6.2854
(190 1900%) 4.9510
(191 1910%) 4.9567
(192 1920%) 7.9553
(193 1930%) 6.4501
(194 1939%) 5.0072
(195 1950%) 7.9465
(196 1960%) 4.2768
(197 1970%) 7.6960
(198 1980%) 5.4163
(199 1989%) 2.3627
(200 2000%) 3.3014
(201 2010%) 2.9093
(202 2020%) 4.2868
(203 2030%) 5.1238
(204 2039%) 3.4426
(205 2050%) 3.3029
(206 2060%) 3.6530
(207 2070%) 6.3795
(208 2080%) 8.2970
(209 2090%) 4.1029
(210 2100%) 3.0094
(211 2110%) 2.8075
(212 2120%) 5.7466
(213 2130%) 6.8209
(214 2140%) 4.0509
(215 2150%) 6.3899
(216 2160%) 6.6765
(217 2170%) 4.8609
(218 2180%) 5.7061
(219 2190%) 2.6665
(220 2200%) 4.2997
(221 2210%) 4.6779
(222 2220%) 7.7399
(223 2230%) 3.4904
(224 2240%) 5.9524
(225 2250%) 3.3732
(226 2260%) 6.5530
(227 2270%) 6.3897
(228 2280%) 4.3997
(229 2290%) 6.3131
(230 2300%) 6.1737
(231 2310%) 5.6638
(232 2320%) 6.5595
(233 2330%) 4.7846
(234 2340%) 6.2470
(235 2350%) 5.2470
(236 2360%) 5.0243
(237 2370%) 4.0766
(238 2380%) 4.9771
(239 2390%) 4.1620
(240 2400%) 4.2752
```

```
(241 2410%) 6.1546
(242 2420%) 8.0127
(243 2430%) 7.3944
(244 2440%) 3.7852
(245 2450%) 5.5639
(246 2460%) 5.8070
(247 2470%) 4.5081
(248 2480%) 3.3177
(249 2490%) 6.1424
(250 2500%) 4.8769
(251 2510%) 5.6767
(252 2520%) 6.3313
(253 2530%) 5.8503
(254 2540%) 5.1131
(255 2550%) 6.2380
(256 2560%) 6.6334
(257 2570%) 4.9057
(258 2580%) 7.5600
(259 2590%) 6.3421
(260 2600%) 6.0600
(261 2610%) 5.9970
(262 2620%) 7.5013
(263 2630%) 2.9166
(264 2640%) 4.4993
(265 2650%) 6.3387
(266 2660%) 7.5894
(267 2670%) 4.9674
(268 2680%) 4.9916
(269 2690%) 7.0191
(270 2700%) 3.1235
(271 2710%) 6.2449
(272 2720%) 4.3063
(273 2730%) 3.6311
(274 2740%) 5.1241
(275 2750%) 4.2560
(276 2760%) 6.5261
(277 2770%) 6.2576
(278 2780%) 7.5537
(279 2790%) 6.7295
(280 2800%) 3.4686
(281 2810%) 4.4341
(282 2820%) 4.6484
(283 2830%) 4.1926
(284 2840%) 5.6641
(285 2850%) 7.4276
(286 2860%) 6.3403
(287 2870%) 4.8021
(288 2880%) 6.3185
(289 2890%) 6.1308
(290 2900%) 4.9397
(291 2910%) 6.1703
(292 2920%) 5.1116
(293 2930%) 5.1514
(294 2940%) 7.4930
(295 2950%) 6.1430
(296 2960%) 5.2488
(297 2970%) 6.5583
(298 2980%) 6.4585
(299 2990%) 2.5360
(300 3000%) 7.3432
(301 3010%) 5.3094
(302 3020%) 4.7731
(303 3030%) 3.8094
(304 3040%) 6.4413
(305 3050%) 4.8452
(306 3060%) 4.8443
(307 3070%) 6.7259
(308 3080%) 5.0113
(309 3090%) 4.2529
(310 3100%) 5.3686
(311 3110%) 3.5010
(312 3120%) 7.3298
(313 3130%) 3.0126
(314 3140%) 5.7961
(315 3150%) 4.0713
(316 3160%) 4.1895
(317 3170%) 4.2154
(318 3180%) 3.3621
(319 3190%) 3.7719
(320 3200%) 3.5461
(321 3210%) 3.8672
(322 3220%) 2.2592
(323 3229%) 4.4236
(324 3240%) 4.1651
(325 3250%) 4.9972
(326 3260%) 6.3210
(327 3270%) 5.6646
(328 3279%) 3.1966
```

```
(329 3290%) 1.8386
(330 3300%) 6.2228
(331 3310%) 2.9484
(332 3320%) 2.9042
(333 3329%) 6.2112
(334 3340%) 5.7372
(335 3350%) 2.9380
(336 3360%) 2.9884
(337 3370%) 6.4366
(338 3379%) 4.8631
(339 3390%) 2.9941
(340 3400%) 5.9566
(341 3410%) 6.0075
(342 3420%) 2.9243
(343 3429%) 4.9600
(344 3440%) 6.1358
(345 3450%) 4.8024
(346 3460%) 6.3665
(347 3470%) 4.5617
(348 3479%) 5.5599
(349 3490%) 5.3879
(350 3500%) 2.4614
(351 3510%) 5.3554
(352 3520%) 4.6011
(353 3529%) 6.9401
(354 3540%) 6.3419
(355 3550%) 5.3728
(356 3560%) 7.0425
(357 3570%) 2.2512
(358 3579%) 2.2133
(359 3590%) 4.6224
(360 3600%) 4.7654
(361 3610%) 3.4882
(362 3620%) 6.1286
(363 3629%) 3.6462
(364 3640%) 3.8383
(365 3650%) 3.8151
(366 3660%) 5.7941
(367 3670%) 2.7073
(368 3679%) 4.1310
(369 3690%) 5.8564
(370 3700%) 4.9188
(371 3710%) 3.4872
(372 3720%) 2.7547
(373 3729%) 4.3752
(374 3740%) 4.2608
(375 3750%) 2.6504
(376 3760%) 4.4365
(377 3770%) 4.5438
(378 3779%) 4.0117
(379 3790%) 3.2426
(380 3800%) 2.8025
(381 3810%) 2.5820
(382 3820%) 4.7871
(383 3829%) 2.3473
(384 3840%) 3.3660
(385 3850%) 4.7917
(386 3860%) 6.2630
(387 3870%) 2.2825
(388 3879%) 5.9082
(389 3890%) 6.0612
(390 3900%) 6.6363
(391 3910%) 6.3408
(392 3920%) 3.3637
(393 3929%) 2.9790
(394 3940%) 1.2784
(395 3950%) 4.6737
(396 3960%) 4.6312
(397 3970%) 4.9905
(398 3979%) 4.4991
(399 3990%) 2.3875
(400 4000%) 5.5885
(401 4010%) 5.1912
(402 4020%) 5.1969
(403 4029%) 3.3273
(404 4040%) 3.9910
(405 4050%) 2.1268
(406 4060%) 6.0192
(407 4070%) 5.8745
(408 4079%) 6.8655
(409 4090%) 2.8233
(410 4100%) 4.5174
(411 4110%) 4.6348
(412 4120%) 5.2043
(413 4130%) 5.9632
(414 4140%) 3.6224
(415 4150%) 3.3544
(416 4160%) 4.4811
```

```
(417 4170%) 7.1022
(418 4180%) 2.6919
(419 4190%) 3.3021
(420 4200%) 6.3905
(421 4210%) 5.4008
(422 4220%) 6.6378
(423 4230%) 6.1315
(424 4240%) 6.6388
(425 4250%) 4.9089
(426 4260%) 6.4351
(427 4270%) 5.1482
(428 4280%) 3.0260
(429 4290%) 4.9495
(430 4300%) 3.7016
(431 4310%) 6.1299
(432 4320%) 4.0895
(433 4330%) 3.9902
(434 4340%) 5.9496
(435 4350%) 5.3130
(436 4360%) 4.6522
(437 4370%) 5.4032
(438 4380%) 4.4384
(439 4390%) 5.7474
(440 4400%) 4.3345
(441 4410%) 2.7817
(442 4420%) 4.0884
(443 4430%) 3.5009
(444 4440%) 2.8396
(445 4450%) 4.5937
(446 4460%) 3.1118
(447 4470%) 3.3859
(448 4480%) 2.5290
(449 4490%) 1.8920
(450 4500%) 5.7369
(451 4510%) 1.6092
(452 4520%) 6.4466
(453 4530%) 7.0454
(454 4540%) 5.4016
(455 4550%) 6.4004
(456 4560%) 5.4915
(457 4570%) 4.9020
(458 4580%) 2.3821
(459 4590%) 6.1554
(460 4600%) 6.0630
(461 4610%) 6.0515
(462 4620%) 2.9704

(463 4630%) 4.3882
(464 4640%) 2.5619
(465 4650%) 3.1845
(466 4660%) 3.5118
(467 4670%) 3.7234
(468 4680%) 4.5972
(469 4690%) 4.2440
(470 4700%) 5.6884
(471 4710%) 3.2573
(472 4720%) 4.5277
(473 4730%) 4.7703
(474 4740%) 3.5489
(475 4750%) 1.6232
(476 4760%) 4.6424
(477 4770%) 5.2152
(478 4780%) 2.1379
(479 4790%) 4.4938
(480 4800%) 3.1150
(481 4810%) 4.2398
(482 4820%) 3.1490
(483 4830%) 2.7097
(484 4840%) 2.2471
(485 4850%) 3.2650
(486 4860%) 2.9194
(487 4870%) 4.7739
(488 4880%) 2.2859
(489 4890%) 4.3408
(490 4900%) 4.5840
(491 4910%) 5.3370
(492 4920%) 4.5242
(493 4930%) 4.1507
(494 4940%) 5.4935
(495 4950%) 3.4002
(496 4960%) 3.6655
(497 4970%) 2.0248
(498 4980%) 3.8510
(499 4990%) 4.2003
(500 5000%) 1.5793
(501 5010%) 2.1432
(502 5020%) 1.8643
(503 5030%) 2.7311
```

```
(504 5040%) 4.2268
(505 5050%) 2.7305
(506 5060%) 3.2924
(507 5070%) 3.4501
(508 5080%) 2.6632
(509 5090%) 1.7728
(510 5100%) 2.4792
(511 5110%) 4.5720
(512 5120%) 2.8095
(513 5130%) 3.0489
(514 5140%) 6.3393
(515 5150%) 2.4366
(516 5160%) 2.4764
(517 5170%) 5.1370
(518 5180%) 2.4352
(519 5190%) 5.5652
(520 5200%) 4.5606
(521 5210%) 1.2337
(522 5220%) 1.3014
(523 5230%) 6.1669
(524 5240%) 3.3596
(525 5250%) 5.6415
(526 5260%) 1.6060
(527 5270%) 3.2634
(528 5280%) 2.7399
(529 5290%) 5.1398
(530 5300%) 2.7163
(531 5310%) 3.0911
(532 5320%) 6.0871
(533 5330%) 3.7637
(534 5340%) 5.8852
(535 5350%) 3.2227
(536 5360%) 4.0740
(537 5370%) 4.2617
(538 5380%) 3.2622
(539 5390%) 2.3433
(540 5400%) 3.5195
(541 5410%) 3.3033
(542 5420%) 6.1364
(543 5430%) 2.7124
(544 5440%) 5.1742
(545 5450%) 1.5899
(546 5460%) 1.7169
(547 5470%) 6.0026
(548 5480%) 2.8376
(549 5490%) 3.4707
(550 5500%) 4.2186
(551 5510%) 3.3628
(552 5520%) 1.8032
(553 5530%) 1.4022
(554 5540%) 1.5711
(555 5550%) 2.3210
(556 5560%) 2.4524
(557 5570%) 3.4528
(558 5580%) 1.3069
(559 5590%) 5.9102
(560 5600%) 1.9204
(561 5610%) 3.1208
(562 5620%) 5.6202
(563 5630%) 4.2761
(564 5640%) 2.7956
(565 5650%) 2.7352
(566 5660%) 6.7815
(567 5670%) 4.2670
(568 5680%) 2.3099
(569 5690%) 3.4949
(570 5700%) 4.5932
(571 5710%) 6.2870
(572 5720%) 2.3541
(573 5730%) 3.0744
(574 5740%) 5.6804
(575 5750%) 4.2349
(576 5760%) 5.3227
(577 5770%) 2.0502
(578 5780%) 5.7806
(579 5790%) 2.0841
(580 5800%) 6.0612
(581 5810%) 2.4590
(582 5820%) 1.0460
(583 5830%) 1.3983
(584 5840%) 1.9901
(585 5850%) 6.1954
(586 5860%) 4.3326
(587 5870%) 2.7201
(588 5880%) 4.2186
(589 5890%) 6.2596
(590 5900%) 4.6208
(591 5910%) 2.6723
```

```
(592 5920%) 3.0902
(593 5930%) 3.9246
(594 5940%) 3.9104
(595 5950%) 4.5507
(596 5960%) 1.4327
(597 5970%) 3.6279
(598 5980%) 5.5394
(599 5990%) 6.5413
(600 6000%) 1.9520
(601 6010%) 4.2267
(602 6020%) 2.8289
(603 6030%) 3.4020
(604 6040%) 3.7404
(605 6050%) 3.2151
(606 6060%) 3.7463
(607 6070%) 1.8858
(608 6080%) 3.8114
(609 6090%) 1.5263
(610 6100%) 2.0896
(611 6110%) 5.0682
(612 6120%) 1.4567
(613 6130%) 6.0593
(614 6140%) 4.5287
(615 6150%) 1.3939
(616 6160%) 2.1800
(617 6170%) 4.7549
(618 6180%) 3.3318
(619 6190%) 4.5872
(620 6200%) 4.2637
(621 6210%) 6.2524
(622 6220%) 4.5613
(623 6230%) 2.3501
(624 6240%) 1.0413
(625 6250%) 3.9661
(626 6260%) 1.3207
(627 6270%) 2.2535
(628 6280%) 4.2964
(629 6290%) 2.2436
(630 6300%) 3.6343
(631 6310%) 3.0994
(632 6320%) 1.9402
(633 6330%) 3.4909
(634 6340%) 3.2099
(635 6350%) 4.0970
(636 6360%) 4.1430
(637 6370%) 5.9743
(638 6380%) 3.7235
(639 6390%) 4.3740
(640 6400%) 4.1232
(641 6409%) 4.1537
(642 6420%) 4.1483
(643 6430%) 6.1538
(644 6440%) 1.8419
(645 6450%) 4.0008
(646 6459%) 1.6962
(647 6470%) 4.4074
(648 6480%) 4.1972
(649 6490%) 5.6707
(650 6500%) 1.8554
(651 6509%) 0.7198
(652 6520%) 2.0245
(653 6530%) 3.7353
(654 6540%) 1.7167
(655 6550%) 3.1159
(656 6559%) 2.4822
(657 6570%) 3.9311
(658 6580%) 5.3745
(659 6590%) 4.0397
(660 6600%) 2.5777
(661 6609%) 3.6315
(662 6620%) 1.1101
(663 6630%) 6.4561
(664 6640%) 2.7261
(665 6650%) 2.1559
(666 6659%) 1.3620
(667 6670%) 1.9939
(668 6680%) 3.3508
(669 6690%) 2.2311
(670 6700%) 0.9456
(671 6709%) 3.0271
(672 6720%) 0.8981
(673 6730%) 3.2267
(674 6740%) 1.6600
(675 6750%) 3.1209
(676 6759%) 4.6881
(677 6770%) 3.8797
(678 6780%) 4.4218
(679 6790%) 6.8815
```

```
(680 6800%) 2.1220
(681 6809%) 4.9250
(682 6820%) 4.5081
(683 6830%) 4.2446
(684 6840%) 3.1571
(685 6850%) 2.8989
(686 6859%) 1.7244
(687 6870%) 4.2176
(688 6880%) 1.9445
(689 6890%) 3.1798
(690 6900%) 4.2148
(691 6909%) 3.2200
(692 6920%) 2.3887
(693 6930%) 2.2243
(694 6940%) 3.1446
(695 6950%) 0.9657
(696 6959%) 4.0849
(697 6970%) 6.1134
(698 6980%) 2.2058
(699 6990%) 1.1061
(700 7000%) 6.7907
(701 7009%) 3.6868
(702 7020%) 3.7562
(703 7030%) 2.4400
(704 7040%) 4.7776
(705 7050%) 3.5131
(706 7059%) 3.6515
(707 7070%) 5.6338
(708 7080%) 1.1374
(709 7090%) 2.1899
(710 7100%) 2.8442
(711 7109%) 4.5871
(712 7120%) 3.8321
(713 7130%) 6.8139
(714 7140%) 3.8761
(715 7150%) 2.2034
(716 7159%) 3.3655
(717 7170%) 4.4994
(718 7180%) 2.1014
(719 7190%) 2.6824
(720 7200%) 6.3732
(721 7209%) 3.4790
(722 7220%) 2.7774
(723 7230%) 5.5364
(724 7240%) 1.1725
(725 7250%) 1.1051
(726 7259%) 1.8192
(727 7270%) 2.6738
(728 7280%) 2.9622
(729 7290%) 5.2896
(730 7300%) 1.2833
(731 7309%) 1.1724
(732 7320%) 1.2706
(733 7330%) 3.8615
(734 7340%) 4.3453
(735 7350%) 2.5249
(736 7359%) 3.3489
(737 7370%) 3.9453
(738 7380%) 1.7922
(739 7390%) 5.8228
(740 7400%) 2.6815
(741 7409%) 1.2792
(742 7420%) 2.1995
(743 7430%) 1.7845
(744 7440%) 3.8312
(745 7450%) 4.2562
(746 7459%) 2.4810
(747 7470%) 3.5921
(748 7480%) 4.2468
(749 7490%) 2.8617
(750 7500%) 2.0669
(751 7509%) 6.4175
(752 7520%) 0.8401
(753 7530%) 4.4079
(754 7540%) 3.5944
(755 7550%) 4.5841
(756 7559%) 2.9359
(757 7570%) 3.1534
(758 7580%) 2.5979
(759 7590%) 2.4932
(760 7600%) 1.9984
(761 7609%) 6.1752
(762 7620%) 3.3149
(763 7630%) 5.7867
(764 7640%) 1.1176
(765 7650%) 1.6334
(766 7659%) 5.7810
(767 7670%) 1.8960
```

```
(768 7680%) 2.3284
(769 7690%) 4.4677
(770 7700%) 2.8963
(771 7709%) 2.4750
(772 7720%) 3.9073
(773 7730%) 2.0620
(774 7740%) 1.0619
(775 7750%) 3.8097
(776 7759%) 1.3092
(777 7770%) 3.3448
(778 7780%) 4.5576
(779 7790%) 6.5549
(780 7800%) 1.0476
(781 7809%) 1.7388
(782 7820%) 3.2340
(783 7830%) 4.2763
(784 7840%) 4.2132
(785 7850%) 1.2507
(786 7859%) 2.9833
(787 7870%) 7.1884
(788 7880%) 5.4576
(789 7890%) 0.9589
(790 7900%) 4.4913
(791 7909%) 5.3599
(792 7920%) 4.0107
(793 7930%) 4.7250
(794 7940%) 2.9034
(795 7950%) 1.1081
(796 7959%) 6.0847
(797 7970%) 2.5734
(798 7980%) 6.2019
(799 7990%) 3.5667
(800 8000%) 4.9820
(801 8009%) 2.0571
(802 8020%) 3.0828
(803 8030%) 3.5585
(804 8040%) 5.0399
(805 8050%) 6.4270
(806 8059%) 4.5826
(807 8070%) 2.4345
(808 8080%) 4.3550
(809 8090%) 2.1160
(810 8100%) 6.1394
(811 8109%) 4.1133
(812 8120%) 2.1501
(813 8130%) 4.5464
(814 8140%) 2.2458
(815 8150%) 3.1960
(816 8159%) 3.1854
(817 8170%) 3.3364
(818 8180%) 3.7407
(819 8190%) 3.0604
(820 8200%) 4.2654
(821 8210%) 5.4341
(822 8220%) 4.4157
(823 8230%) 4.4647
(824 8240%) 2.6290
(825 8250%) 2.8269
(826 8260%) 3.1470
(827 8270%) 4.8573
(828 8280%) 4.6543
(829 8290%) 3.0168
(830 8300%) 2.2960
(831 8310%) 5.9380
(832 8320%) 4.7555
(833 8330%) 3.0140
(834 8340%) 0.9124
(835 8350%) 6.8300
(836 8360%) 2.3461
(837 8370%) 2.5810
(838 8380%) 4.0752
(839 8390%) 3.3699
(840 8400%) 3.2453
(841 8410%) 1.4774
(842 8420%) 2.7417
(843 8430%) 1.1580
(844 8440%) 4.9844
(845 8450%) 5.6506
(846 8460%) 5.1236
(847 8470%) 3.9320
(848 8480%) 4.3702
(849 8490%) 4.4216
(850 8500%) 6.2106
(851 8510%) 2.9564
(852 8520%) 3.9072
(853 8530%) 4.9957
(854 8540%) 3.7464
(855 8550%) 2.2657
```

```
(856 8560%) 3.2035
(857 8570%) 4.4796
(858 8580%) 4.3141
(859 8590%) 2.1664
(860 8600%) 6.4033
(861 8610%) 3.3228
(862 8620%) 4.2456
(863 8630%) 4.5317
(864 8640%) 3.7598
(865 8650%) 2.7789
(866 8660%) 5.7930
(867 8670%) 2.9396
(868 8680%) 1.9242
(869 8690%) 4.5286
(870 8700%) 1.5416
(871 8710%) 4.6725
(872 8720%) 4.4981
(873 8730%) 2.9960
(874 8740%) 2.6844
(875 8750%) 2.8118
(876 8760%) 3.8506
(877 8770%) 2.5259
(878 8780%) 2.7148
(879 8790%) 3.6405
(880 8800%) 1.5530
(881 8810%) 3.1391
(882 8820%) 1.9213
(883 8830%) 4.0698
(884 8840%) 2.2137
(885 8850%) 4.4922
(886 8860%) 4.2562
(887 8870%) 1.1027
(888 8880%) 2.8725
(889 8890%) 2.9921
(890 8900%) 2.1795
(891 8910%) 5.2772
(892 8920%) 1.5078
(893 8930%) 2.1069
(894 8940%) 1.7157

(895 8950%) 2.3273
(896 8960%) 2.0332
(897 8970%) 2.3997
(898 8980%) 4.1392
(899 8990%) 1.6401
(900 9000%) 3.3084
(901 9010%) 2.1375
(902 9020%) 1.0310
(903 9030%) 3.6959
(904 9040%) 2.8701
(905 9050%) 2.0204
(906 9060%) 3.8856
(907 9070%) 4.8772
(908 9080%) 3.2336
(909 9090%) 1.0065
(910 9100%) 2.8203
(911 9110%) 6.7024
(912 9120%) 4.3834
(913 9130%) 5.5559
(914 9140%) 3.9844
(915 9150%) 4.4662
(916 9160%) 1.2627
(917 9170%) 1.8317
(918 9180%) 2.9188
(919 9190%) 2.0300
(920 9200%) 2.3838
(921 9210%) 4.1008
(922 9220%) 4.0528
(923 9230%) 4.3405
(924 9240%) 4.5046
(925 9250%) 2.3615
(926 9260%) 3.2252
(927 9270%) 2.0133
(928 9280%) 2.0388
(929 9290%) 2.2498
(930 9300%) 2.5454
(931 9310%) 2.5528
(932 9320%) 2.5090
(933 9330%) 2.5109
(934 9340%) 2.2407
(935 9350%) 3.1549
(936 9360%) 3.5790
(937 9370%) 1.5973
(938 9380%) 3.2621
(939 9390%) 2.3939
(940 9400%) 4.7931
(941 9410%) 3.3666
(942 9420%) 4.1133
(943 9430%) 4.9498
```

```
(944 9440%) 5.9166
(945 9450%) 2.9350
(946 9460%) 2.1426
(947 9470%) 1.9516
(948 9480%) 1.3464
(949 9490%) 2.4034
(950 9500%) 2.9451
(951 9510%) 1.1675
(952 9520%) 4.6673
(953 9530%) 3.7889
(954 9540%) 1.8215
(955 9550%) 1.8975
(956 9560%) 4.2725
(957 9570%) 4.0650
(958 9580%) 4.5495
(959 9590%) 4.4778
(960 9600%) 2.0526
(961 9610%) 3.6464
(962 9620%) 4.0264
(963 9630%) 1.9329
(964 9640%) 4.5144
(965 9650%) 2.4502
(966 9660%) 1.1273
(967 9670%) 4.0193
(968 9680%) 3.5556
(969 9690%) 2.6650
(970 9700%) 2.0316
(971 9710%) 2.5513
(972 9720%) 2.1210
(973 9730%) 3.6899
(974 9740%) 3.6209
(975 9750%) 1.3453
(976 9760%) 1.7917
(977 9770%) 4.0436
(978 9780%) 1.3313
(979 9790%) 1.9874
(980 9800%) 6.0942
(981 9810%) 2.5031
(982 9820%) 1.2699
(983 9830%) 1.0183
(984 9840%) 3.8106
(985 9850%) 5.8766
(986 9860%) 4.2649
(987 9870%) 6.1361
(988 9880%) 4.6726
(989 9890%) 4.0461
(990 9900%) 2.5812
(991 9910%) 3.7053
(992 9920%) 3.0813
(993 9930%) 2.7247
(994 9940%) 3.5682
(995 9950%) 0.8832
(996 9960%) 0.9582
(997 9970%) 2.8367
(998 9980%) 3.9911
(999 9990%) 5.7930
```