

André Cardoso 50%  
andremacardoso@ua.pt  
108269

Tiago Figueiredo 50%  
tiago.a.figueiredo@ua.pt  
107263

09 de Janeiro de 2023

## Conteúdo

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Funções Usadas</b>	<b>3</b>
2.1	Hash Table . . . . .	3
2.1.1	Constructor . . . . .	3
2.1.2	Destructor . . . . .	3
2.1.3	add . . . . .	3
2.1.4	get . . . . .	4
2.1.5	add_edge . . . . .	4
2.1.6	BFS . . . . .	4
2.1.7	DFS . . . . .	5
2.1.8	list_connected_components . . . . .	5
2.1.9	find . . . . .	5
2.1.10	g_union . . . . .	6
2.1.11	hash and unhash . . . . .	6
2.2	Estatísticas da Hash Table . . . . .	6
2.2.1	get_load_factor e get_collisions . . . . .	6
2.2.2	get_distribution . . . . .	6
2.3	Estatísticas do Grafo . . . . .	6
2.3.1	get_connected_components . . . . .	6
2.3.2	get_diameter . . . . .	6
2.3.3	get_diameter_node . . . . .	7
2.4	Outras Funções . . . . .	7
2.4.1	longest . . . . .	7
2.4.2	connected . . . . .	7
2.4.3	path_finder . . . . .	7

2.4.4	connected_components . . . . .	7
<b>3</b>	<b>Resultados</b>	<b>7</b>
3.1	Resultados gerais . . . . .	7
3.2	Testes de Memory Leaks . . . . .	9
3.3	Resultados ao fim de 14 dias . . . . .	9
<b>4</b>	<b>Referências</b>	<b>11</b>
<b>5</b>	<b>Apêndice</b>	<b>12</b>
5.1	word_ladder.cpp . . . . .	12
5.2	makefile . . . . .	24
5.3	valgrind-out.txt . . . . .	24
5.4	longest.txt . . . . .	30
5.5	Ficheiros em Python . . . . .	45

## 1 Introdução

Uma word ladder é uma sequência de palavras em que cada palavra difere em uma e só uma letra da palavra anterior. Por exemplo, na língua Portuguesa é possível ir da palavra tudo para a palavra nada em quatro passos. *tudo* → *todo* → *nodo* → *nado* → *nada*. Como tal para resolver o problema proposto de criar um algoritmo em *C/C++* que permita encontrar foi feita uma implementação de uma *Hash Table* em *C++*, usada depois para permitir a implementação de grafos e do *union find*, tornando possível tal algoritmo.

## 2 Funções Usadas

Esta secção contém uma lista, com a respetiva descrição de todas as funções usadas para a criação do algoritmo.

### 2.1 Hash Table

Sendo a linguagem de programação escolhida para a resolução de este problema *C++*, a *Hash table* foi implementada através de duas classes, uma que contém os parâmetros de cada nó da *Hash Table*, e uma que contém a implementação da *Hash Table*.

#### 2.1.1 Constructor

O construtor inicializa todas as variáveis da classe quando um objeto do tipo da classe é criado. Começa por definir o tamanho da hash table como sendo 65536, cria um array de ponteiros chamado *words* com o tamanho da tabela. Inicializa também a variável *entries* a 0, a variável *connected\_components* também a 0 e a variável *load\_factor* como sendo 0,75. Define, por fim, todos os elementos do array *words* como sendo ponteiros nulos, o que indica que todos os elementos da tabela estão vazios.

#### 2.1.2 Destructor

O destrutor dá um loop pelo array *words* eliminando cada elemento não nulo, acabando por eliminar o array, no final.

#### 2.1.3 add

Esta função é usada para adicionar uma nova palavra à hash table. A função recebe um único argumento, uma string chamada *word* que representa a palavra a ser adicionada à tabela.

A função começa chamando a função *hash()* para obter o índice em que a palavra deve ser inserida. Em seguida, verifica se a palavra já está presente na tabela verificando se o elemento no índice calculado não é nulo e se a

palavra armazenada nesse elemento é a mesma da palavra a ser adicionada. Se a palavra já estiver presente, a função retorna sem adicionar a palavra à tabela. A função, caso a palavra não corresponder a nenhuma já existente, verifica se o número de entradas na tabela mais 1 é maior ou igual ao tamanho da tabela multiplicado pelo fator de carga. Se isso for verdade, significa que a tabela está a ficar cheia e é chamada a função `resize()` para aumentar o tamanho da tabela. Se o elemento no índice calculado for nulo, a função chama a função `create()` para criar um novo nó com a palavra e a inserir no índice caso contrário, se a palavra armazenada nesse elemento não for a mesma da palavra a ser adicionada, a função entra em um loop `while`. O loop executa até encontrar um espaço vazio na tabela podendo também parar se a palavra já se encontrar na tabela. Para encontrar o próximo espaço disponível, ele usa *linear probing*. Linear probing é uma técnica de resolução de colisões onde o próximo espaço é encontrado incrementando o índice um por um até ser encontrada uma posição vazia. Se um espaço vazio for encontrado, a função chama a função `create()`, finalmente criando um novo nó com a palavra e inserindo-a no índice.

#### 2.1.4 `get`

Esta função é utilizada para obter um nó específico dado uma palavra. Ela começa chamando a função *hash* para obter o índice onde a palavra deve ser encontrada. Se o elemento no índice calculado for nulo, a função retorna um *nullptr*, indicando que a palavra não está presente. Se o elemento não é nulo, ele verifica se a palavra armazenada é a mesma da palavra que se deseja procurar. Se for, a função retorna o endereço desse elemento, caso contrário a função entra num loop `while` usando linear probing, como na função de cima, para encontrar a palavra ou um espaço vazio. A função retorna o endereço do elemento que contém a palavra, caso este seja encontrado ou *nullptr*.

#### 2.1.5 `add_edge`

Esta função é utilizada para adicionar uma aresta entre dois nós na estrutura de dados que representa um grafo. Ela recebe dois argumentos, "from" e "to", que são ponteiros para os nós entre os quais a aresta será adicionada. A função adiciona o nó "to" à lista de adjacência do nó "from" e o nó "from" à lista de adjacência do nó "to", estabelecendo assim a conexão entre eles. A função incrementa o contador de arestas em ambos os nós, indicando que eles têm uma aresta adicional, chamando por fim a função `g_union` que se encontra descrita abaixo.

#### 2.1.6 `BFS`

A função `BFS` é uma implementação de breadth-first search. Ela começa por percorrer todos os nós e marcando-os como não visitados e sem pais

(senão incorre o risco de não funcionar caso outra função que modifique os campos `visited` e `parent` já tenha sido executada). Em seguida, a função adiciona o nó "from" a uma fila e marca-o como visitado. A função então entra em um loop enquanto a fila não estiver vazia. Dentro do loop, a função pega o primeiro elemento da fila e verifica se é o nó "to". Se for, a função retorna a profundidade atual. Caso contrário, a função percorre todos os nós adjacentes ao nó atual que ainda não foram visitados, marca-os como visitados e adiciona-os à fila. A profundidade é incrementada a cada iteração do loop. Se a profundidade atual é maior que o valor máximo de profundidade especificado e o valor máximo de profundidade é diferente de 0, a função retorna -1. Se o loop termina e o nó "to" ainda não foi encontrado, a função retorna -1.

### 2.1.7 DFS

A função DFS é uma implementação de depth-first search. Ela funciona de maneira semelhante à função BFS, mas usa uma pilha em vez de uma fila. A função marca todos os nós como não visitados e sem pais e adiciona o nó "from" à pilha. A função então entra em um loop enquanto a pilha não estiver vazia. Dentro do loop, a função pega o topo da pilha e verifica se é o nó "to". Se for, a função segue o caminho de volta ao nó "from" contando a profundidade e retorna-a. Caso contrário, a função percorre todos os nós adjacentes ao nó atual que ainda não foram visitados, marca-os como visitados e adiciona-os à pilha. Se o loop termina e o nó "to" ainda não foi encontrado, a função retorna -1.

### 2.1.8 `list_connected_components`

A função `list_connected_components` é usada para listar todos os nós que fazem parte do mesmo componente conectado de um nó específico dado uma palavra. Ela chama a função `get` para obter o nó correspondente à palavra fornecida, se ele não for encontrado, a função imprime "Palavra não encontrada" e retorna. Caso contrário, ela chama outra função chamada "find", passando o nó encontrado como argumento, essa função retorna um representante de um conjunto na estrutura de dados union-find. Em seguida, a função percorre todos os nós na tabela hash e adiciona a um vetor de componentes todos os nós cujo representante é o mesmo do nó encontrado anteriormente. Finalmente, a função imprime "Pertencente ao mesmo componente conectado como [palavra fornecida]:" e imprime todas as palavras armazenadas nos nós do vetor de componentes.

### 2.1.9 find

A função *find* é uma função recursiva que encontra o *representative* de vértice do grafo, verificando se o vértice atual é o requerido até o encontrar.

### **2.1.10 g\_union**

### **2.1.11 hash and unhash**

A função de hash transforma um inteiro ou através do *operator overloading* disponível do *C++*, uma função com o mesmo nome existe para em vez de um inteiro receber uma string, e através do método *64-bit FNV-1a hash* encodificar a string ou inteiro fornecido de modo a poder ser usado como index da hash table. A função unhash reverte este processo, como seria previsível

## **2.2 Estatísticas da Hash Table**

### **2.2.1 get\_load\_factor e get\_collisions**

As funções `get_load_factor` e `get_collisions` são usadas para obter estatísticas sobre a hash table. A primeira função retorna o fator de carga atual, que é o número de entradas na tabela dividido pelo tamanho da tabela. A segunda função retorna o número de colisões na tabela.

### **2.2.2 get\_distribution**

A função `get_distribution` é usada para obter a distribuição de entradas na tabela hash. Ela percorre toda a tabela e adiciona "true" a um vetor de bools, que permite com alguns compiladores guardar o valor em um bit, se uma entrada está presente na posição atual ou "false" se não estiver. A função retorna esse vetor.

## **2.3 Estatísticas do Grafo**

### **2.3.1 get\_connected\_components**

A função `get_connected_components` conta o número de componentes conectados no grafo. Ele percorre a tabela hash e verifica, para cada nó, se é um representante de seu conjunto de união (usando a propriedade `representative`). Se for, incrementa o contador de componentes. O número de componentes conectados é retornado no final.

### **2.3.2 get\_diameter**

A função `get_diameter` calcula o diâmetro do componente conexo. O diâmetro é o caminho maior caminho mais curto entre dois nós do mesmo componente conexo. Começa por percorrer a hash table e chama a função DFS em cada nó, passando o nó atual como início e o nó especificado como parâmetro como destino. A distância retornada pela DFS é comparada com a distância máxima encontrada até agora. Se for maior, atualiza a distância máxima e

salva o nó inicial. No final, se print for verdadeiro, imprime o diâmetro e o caminho encontrado.

### **2.3.3 get\_diameter\_node**

A função `get_diameter_node` é semelhante à anterior, mas apenas retorna o nó inicial do caminho com o maior diâmetro.

## **2.4 Outras Funções**

### **2.4.1 longest**

A função "longest" encontra o diâmetro do componente conexo da palavra dada. Ela faz isso chamando o método `get_diameter()` da tabela de hash correspondente ao tamanho da palavra dada.

### **2.4.2 connected**

A função "connected" verifica se as duas palavras dadas são conectadas (se elas diferem apenas por uma letra). Ela faz isso comparando cada letra das palavras uma a uma e verificando se elas são diferentes.

### **2.4.3 path\_finder**

A função "path\_finder" é usada para encontrar o caminho mais curto entre duas palavras no grafo. Ele usa a função BFS para encontrar a distância mais curta entre as duas palavras e imprime o caminho encontrado.

### **2.4.4 connected\_components**

A função "connected\_components" encontra todas as palavras que estão conectadas (que diferem apenas por uma letra) à palavra dada. Ela faz isso chamando o método `list_connected_components()` da tabela de hash correspondente ao tamanho da palavra dada.

## **3 Resultados**

### **3.1 Resultados gerais**

Após a execução do programa, foi verificado que este cumpria os testes impostos como condição para verificar o bom funcionamento do algoritmo e da hash table.

Alguns destes testes, incluem processar algumas palavras de cada comprimento para verificar que o bom funcionamento do algoritmo e da hash table enquanto este tamanho aumenta, a verificação de uma *word ladder* conhecida, de *etano* a *sitie*, e a confirmação de que o maior tamanho para chegar à

primeira dessas duas palavras é 1135. Como descrito nas secções abaixo, o programa foi capaz de executar sem a criação de *memory leaks*, e de com a capacidade de correr num servidor durante um longo período de tempo. Durante este período, foi capaz de encontrar várias *word ladders*, sendo que o seu tamanho se encontra no gráfico abaixo.

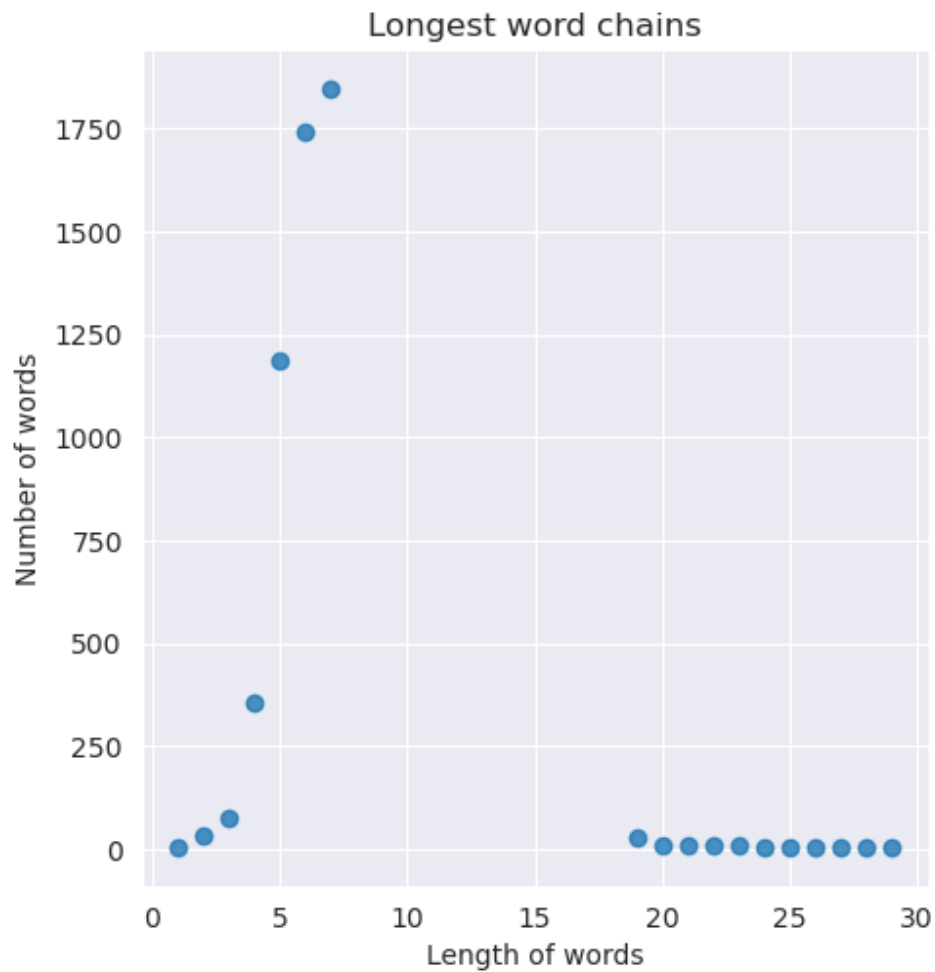


Figura 1: Comprimento das word ladders

Por fim, os dois gráficos seguintes demonstram o de posições ocupadas na hash table, comparados com o número de colisões existentes ao tentar inserir um novo elemento, incluindo a demonstração de que estes valores voltam, como seria esperado, a diminuir, após o aumento da hash table, usando a função *resize*.



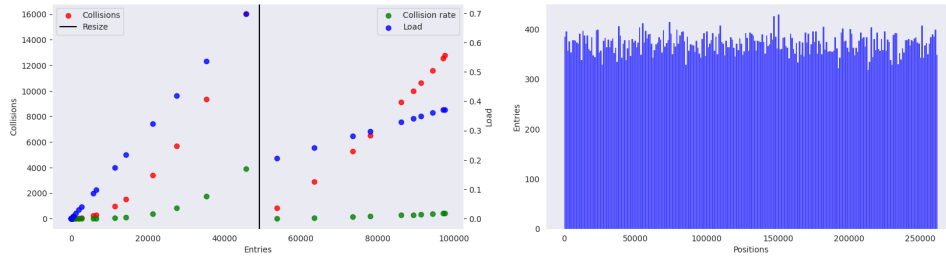


Figura 2: Comprimento das word ladders

### 3.2 Testes de Memory Leaks

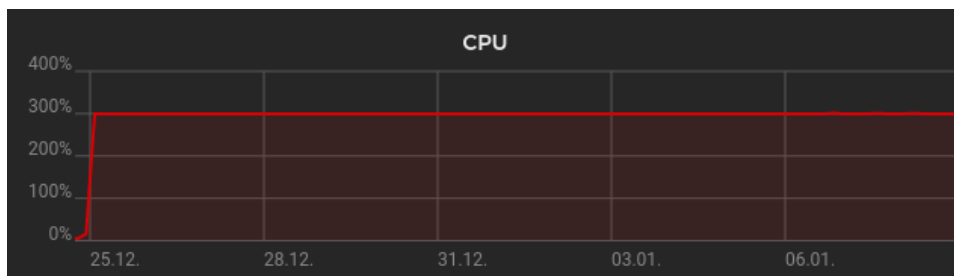
Para determinar se o programa tinha *memory leaks*, foi usado o valgrind, com as seguintes opções:

```
-leak-check=full
-show-leak-kinds=all
-track-origins=yes
-verbose
-log-file=valgrind-out.txt
./word_ladder
```

Tendo sido obtido o ficheiro de log encontrado no apêndice *valgrind-out.txt*, comprovando que não existem *memory leaks*.

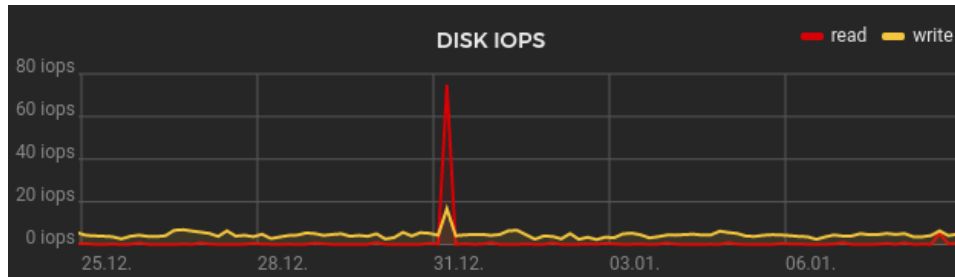
### 3.3 Resultados ao fim de 14 dias

Devido ao facto de que a lógica principal do programa foi acabada com alguma antecedência, o algoritmo correu com o objetivo de encontrar a maior *ladder* possível, para o dicionário fornecido, para cada tamanho de palavra. Infelizmente devido à quantidade de palavras de tamanho superior a oito ou nove letras e inferior a vinte letras, algumas das *ladders* para os comprimentos de palavra nesse intervalo não foram encontrados, tendo a maior sido uma *word ladder* com tamanho 1844 para palavras de tamanho sete.

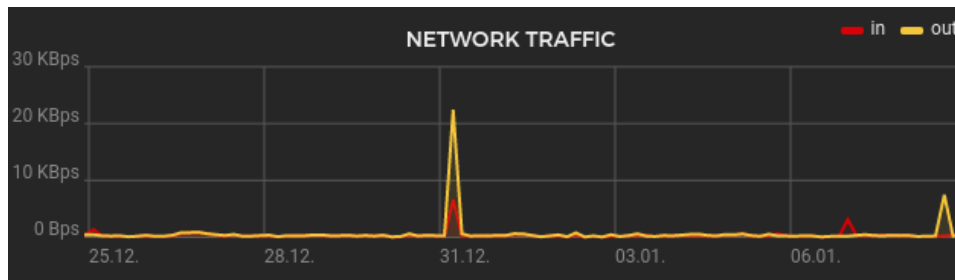


Devido ao uso de todos os threads disponíveis, três núcleos e 6 threads, de um processador *AMD<sup>TM</sup> Epyc Rome*, o uso do processador manteve-se a

300% (cada núcleo a 100%), durante a inteira duração.



O programa também demonstrou um uso residual do disco durante a maioria da sua duração, sendo o pico visível no gráfico, resultante de um pico no uso de rede enquanto este atualizava, visto que havia mais tarefas a correr no mesmo servidor.



Assim sendo, as maiores word ladders encontradas por este programa encontram-se na secção 5.4 do apêndice, sendo que seriam demasiado extensas para se encontrarem nesta descrição.

## 4 Referências

- [Val] *How do I use valgrind to find memory leaks?* 2011. URL: <https://stackoverflow.com/questions/5134891/how-do-i-use-valgrind-to-find-memory-leaks>.
- [Ref20] C Reference. *C Reference*. 2020. URL: <https://en.cppreference.com/w/>. (accessed: 22.12.2022).
- [Sil22] Tomás Oliveira e Silva. *Lecture Notes*. 2022. URL: [elearning.ua.pt](http://elearning.ua.pt). (accessed: 22.12.2022).
- [Fow] *Fowler-Noll-Vo hash function*. URL: [https://en.wikipedia.org/wiki/FowlerNollVo\\_hash\\_function](https://en.wikipedia.org/wiki/FowlerNollVo_hash_function).
- [Wel] Christopher Wellons. *skeeto/hash-prospector*. URL: <https://github.com/skeeto/hash-prospector#three-round-functions>.

## 5 Apêndice

### 5.1 word\_ladder.cpp

```
1  #include <algorithm>
2  #include <fstream>
3  #include <iostream>
4  #include <string>
5  #include <cmath>
6  #include <vector>
7  #include <queue>
8  #include <stack>
9  #include <thread>
10
11 using namespace std;
12 #define _max_word_size_ 32
13
14 class node {
15 public:
16     node(const string &word) : word(word) {
17         parent = nullptr;
18         visited = false;
19         representative = this;
20         vertices = 1;
21         edges = 0;
22     }
23
24     string word;
25     // search relevant data
26     node *parent;
27     bool visited;
28     // graph data structure
29     vector<node *> adjacency_list;
30     // union-find data structure
31     node *representative;
32     int vertices;
33     int edges;
34 };
35
36 class hashTable {
37 public:
38     unsigned int size;
39     node **words;
40     unsigned int entries;
41     int connected_components;
42     double load_factor;
43
44     hashTable() {
45         // Makes the dict only need to be resized once.
```

```

46         size = 65536;
47         words = new node *[size];
48         entries = 0;
49         connected_components = 0;
50         load_factor = 0.75;
51         for (unsigned int i = 0; i < size; i++) {
52             words[i] = nullptr;
53         }
54     }
55
56     ~hashTable() {
57         for (unsigned int i = 0; i < size; i++) {
58             if (words[i] != nullptr) {
59                 delete words[i];
60             }
61         }
62         delete[] words;
63     }
64
65     void add(const string &word) {
66         unsigned int index = hash(word);
67         if (words[index] != nullptr && words[index]->word == word) {
68             return;
69         }
70         if (entries + 1 >= size * load_factor) {
71             resize();
72         }
73         if (words[index] == nullptr) {
74             create(index, word);
75         } else if (words[index]->word != word) {
76             while (words[index] != nullptr && words[index]->word !=
77                 word) {
78                 // Linear probing is the fastest way.
79                 // Probably because it uses the cache more
80                 // efficiently.
81                 // And that matters the most when the table is huge
82                 // and we have memory to spare.
83                 index = (index + 1) % size;
84             }
85             create(index, word);
86         }
87     }
88
89     node *get(const string &word) {
90         unsigned int index = hash(word);
91         if (words[index] == nullptr) {
92             return nullptr;
93         }
94         if (words[index]->word == word) {

```

```

92         return words[index];
93     }
94     while (words[index] != nullptr && words[index]->word !=
95            word) {
96         index = (index + 1) % size;
97     }
98     return words[index];
99 }
100
101 // graph functions
102 void add_edge(node *from, node *to) {
103     from->adjacency_list.push_back(to);
104     to->adjacency_list.push_back(from);
105     from->edges++;
106     to->edges++;
107     g_union(from, to);
108 }
109
110 int BFS(node *from, node *to, int maximum_depth = 0) {
111     for (unsigned int i = 0; i < size; i++) {
112         if (words[i] != nullptr) {
113             words[i]->visited = false;
114             words[i]->parent = nullptr;
115         }
116     }
117     queue < node * > q;
118     from->visited = true;
119     q.push(from);
120     int depth = 0;
121     while (!q.empty()) {
122         int q_size = q.size();
123         for (int i = 0; i < q_size; i++) {
124             node *current = q.front();
125             q.pop();
126             if (current == to) {
127                 return depth;
128             }
129             for (size_t j = 0; j <
130                  current->adjacency_list.size(); j++) {
131                 node *adjacent = current->adjacency_list[j];
132                 if (!adjacent->visited) {
133                     adjacent->visited = true;
134                     adjacent->parent = current;
135                     q.push(adjacent);
136                 }
137             }
138             depth++;
139             if (depth > maximum_depth && maximum_depth != 0) {

```

```

139         return -1;
140     }
141 }
142 return -1;
143 }
144
145 int DFS(node *from, node *to) {
146     for (unsigned int i = 0; i < size; i++) {
147         if (words[i] != nullptr) {
148             words[i]->visited = false;
149             words[i]->parent = nullptr;
150         }
151     }
152     stack < node * > q;
153     from->visited = true;
154     q.push(from);
155     int depth = 0;
156     while (!q.empty()) {
157         int q_size = q.size();
158         for (int i = 0; i < q_size; i++) {
159             node *current = q.top();
160             q.pop();
161             if (current == to) {
162                 // god why
163                 while (current->parent != nullptr && current !=
164                     from) {
165                     current = current->parent;
166                     depth++;
167                 }
168                 return depth;
169             }
170             for (size_t j = 0; j <
171                 current->adjacency_list.size(); j++) {
172                 node *adjacent = current->adjacency_list[j];
173                 if (!adjacent->visited) {
174                     adjacent->visited = true;
175                     adjacent->parent = current;
176                     q.push(adjacent);
177                 }
178             }
179         }
180     }
181     return -1;
182 }
183
184 void list_connected_components(const string &word) {
185     vector < node * > components;
186     node *vertex = get(word);
187     if (vertex == nullptr) {

```

```

186         cout << "Word not found" << endl;
187         return;
188     }
189     node *representative = find(vertex);
190     for (unsigned int i = 0; i < size; i++) {
191         if (words[i] != nullptr && find(words[i]) ==
            representative) {
192             components.push_back(words[i]);
193         }
194     }
195     cout << "Belonging to same connected component as " << word
        << "are:" << endl;
196     for (size_t i = 0; i < components.size(); i++) {
197         cout << components[i]->word << "\n";
198     }
199 }
200
201 // hash table statistics
202 double get_load_factor() {
203     return (double) entries / size;
204 }
205
206 int get_collisions() {
207     unsigned int collisions = 0;
208     for (unsigned int i = 0; i < size; i++) {
209         if (words[i] != nullptr) {
210             if (hash(words[i]->word) != i) {
211                 collisions++;
212             }
213         }
214     }
215     return collisions;
216 }
217
218 vector<bool> get_distribution() {
219     vector<bool> distribution;
220     for (unsigned int i = 0; i < size; i++) {
221         if (words[i] != nullptr) {
222             distribution.push_back(true);
223         } else {
224             distribution.push_back(false);
225         }
226     }
227     return distribution;
228 }
229
230 // graph statistics
231 int get_connected_components() {
232     int components = 0;

```



```

233     for (unsigned int i = 0; i < size; i++) {
234         if (words[i] != nullptr) {
235             if (words[i]->representative == words[i]) {
236                 components++;
237             }
238         }
239     }
240     return components;
241 }
242
243 int get_diameter(node *n, bool print = true) {
244     int diameter = 0;
245     node *max = nullptr;
246     for (unsigned int i = 0; i < size; i++) {
247         if (words[i] != nullptr) {
248             if (words[i]->adjacency_list.size() == 0) {
249                 continue;
250             }
251             int distance = DFS(words[i], n);
252             if (distance > diameter) {
253                 diameter = distance;
254                 max = words[i];
255             }
256         }
257     }
258     // DFS data is wiped out every run.
259     DFS(n, max);
260     node *res = max;
261     if (res == nullptr) {
262         return 0;
263     }
264     if (print) {
265         cout << "Diameter: " << diameter << endl;
266         cout << "Path: ";
267         if (res == nullptr) {
268             cout << "No connected words." << endl;
269         }
270         while (res->parent != nullptr) {
271             cout << res->word << " -> ";
272             res = res->parent;
273         }
274         cout << res->word << endl;
275     }
276
277     return diameter;
278 }
279
280 node *get_diameter_node(node *n) {
281     int diameter = 0;

```

```

282     node *max = nullptr;
283     for (unsigned int i = 0; i < size; i++) {
284         if (words[i] != nullptr) {
285             int distance = DFS(words[i], n);
286             if (distance > diameter) {
287                 diameter = distance;
288                 max = words[i];
289             }
290         }
291     }
292     return max;
293 }
294
295 private:
296     void create(int index, const string &word) {
297         entries++;
298         connected_components++;
299         words[index] = new node(word);
300     }
301
302     void resize() {
303         // High resize coefficient to reduce resizes, which are
304             expensive.
305         int coeff = 4;
306         size *= coeff;
307         node **new_words = new node *[size];
308         for (unsigned int i = 0; i < size; i++) {
309             new_words[i] = nullptr;
310         }
311         for (unsigned int i = 0; i < size / coeff; i++) {
312             if (words[i] != nullptr) {
313                 int index = hash(words[i]->word);
314                 if (new_words[index] == nullptr) {
315                     new_words[index] = words[i];
316                 } else {
317                     while (new_words[index] != nullptr) {
318                         index = (index + 1) % size;
319                     }
320                 }
321             }
322         }
323         delete[] words;
324         words = new_words;
325     }
326
327     node *find(node *vertex) {
328         if (vertex->representative != vertex) {
329             vertex->representative = find(vertex->representative);
330         }
331     }

```

```

330     return vertex->representative;
331 }
332
333 void g_union(node *from, node *to) {
334     node *from_rep = find(from);
335     node *to_rep = find(to);
336     if (from_rep != to_rep) {
337         to_rep->representative = from_rep;
338         connected_components--;
339     }
340 }
341
342 void print_adjacency_list(node *n) {
343     cout << n->word << " -> ";
344     for (size_t i = 0; i < n->adjacency_list.size(); i++) {
345         cout << n->adjacency_list[i]->word << " ";
346     }
347     cout << endl;
348 }
349
350 #define FNV_OFFSET 14695981039346656037UL
351 #define FNV_PRIME 1099511628211UL
352
353 // Return 64-bit FNV-1a hash for key (NUL-terminated).
354 unsigned int hash(const string &word) {
355     uint64_t hash = FNV_OFFSET;
356     const char *key = word.c_str();
357     for (const char *p = key; *p; p++) {
358         hash ^= (uint64_t)(unsigned char)(*p);
359         hash *= FNV_PRIME;
360     }
361     // Ensure hash is adjusted to the size of the table.
362     return (size_t)(hash & (uint64_t)(size - 1));
363 }
364
365 //
366     https://github.com/skeeto/hash-prospector#three-round-functions
367 // Kept for reference.
368 unsigned int hash(int x) {
369     x ^= x >> 17;
370     x *= 0xed5ad4bb;
371     x ^= x >> 11;
372     x *= 0xac4c1b51;
373     x ^= x >> 15;
374     x *= 0x31848bab;
375     x ^= x >> 14;
376     return x;
377 }

```

```

378     unsigned int unhash(int x) {
379         x ^= x >> 14 ^ x >> 28;
380         x *= 0x32b21703;
381         x ^= x >> 15 ^ x >> 30;
382         x *= 0x469e0db1;
383         x ^= x >> 11 ^ x >> 22;
384         x *= 0x79a85073;
385         x ^= x >> 17;
386         return x;
387     }
388 };
389
390 void longest(hashTable **dicts, const string &word) {
391     hashTable *dict = dicts[word.size() - 1];
392     node *n = dict->get(word);
393     cout << "Longest path to " << word << " is " << endl
394           << dict->get_diameter(n)
395           << " words long." << endl;
396     return;
397 }
398
399 bool connected(const string &a, const string &b) {
400     if (a.size() != b.size())
401         return false;
402     bool result = false;
403     for (size_t i = 0; i < a.size(); i++) {
404         if (a[i] != b[i]) {
405             // Only one difference is allowed
406             if (result)
407                 return false;
408             result = true;
409         }
410     }
411     return result;
412 }
413
414 void path_finder(hashTable **dicts, const string &start, const
415                 string &end) {
416     if (start.size() != end.size()) {
417         cout << "Cannot compare different sizes." << endl;
418         return;
419     }
420     hashTable *dict = dicts[start.size() - 1];
421     cout << "Trying to go from " << start << " to " << end << endl;
422     node *from = dict->get(end);
423     node *to = dict->get(start);
424     if (from == nullptr || to == nullptr) {
425         cout << "No path found." << endl;
426         return;
427     }

```

```

426     }
427     int travelled = dict->BFS(from, to);
428     cout << "Travelled " << travelled << " nodes. " << endl;
429     node *res = to;
430     while (res->parent != nullptr) {
431         cout << res->word << " -> ";
432         res = res->parent;
433     }
434     cout << res->word << endl;
435 }
436
437 void connected_components(hashTable **dicts, const string &word) {
438     hashTable *dict = dicts[word.size() - 1];
439     dict->list_connected_components(word);
440 }
441
442 void end(hashTable **dicts) {
443     #if defined(_stats_) || defined(_detail_) || defined(_full_)
444         ofstream file;
445         file.open("stats.txt");
446     #endif
447     for (size_t i = 0; i < _max_word_size_; i++) {
448         #if defined(_stats_) || defined(_detail_) || defined(_full_)
449             file << endl;
450             file << "Hash Table for " << i + 1 << " letter words" <<
451                 endl;
452             file << "Size: " << dicts[i]->size << endl;
453             file << "Load factor: " << dicts[i]->get_load_factor() <<
454                 endl;
455             file << "Collisions: " << dicts[i]->get_collisions() << endl;
456             #if defined(_detail_) || defined(_full_)
457                 vector<bool> distribution = dicts[i]->get_distribution();
458                 file << "Distribution: " << endl;
459                 for (size_t j = 0; j < distribution.size(); j++)
460                 {
461                     if (distribution[j])
462                         file << j << " ";
463                 }
464                 file << endl;
465             #endif
466             #endif
467             delete dicts[i];
468         }
469     }
470     #if defined(_stats_) || defined(_detail_) || defined(_full_)
471         file.close();
472     #endif
473 }
474
475 void graph_builder(hashTable *dict) {

```

```

473     int sizes = 0;
474     // TODO: Optimize this, O(n^1.5) ish isn't good
475     for (size_t i = 0; i < dict->size; i++) {
476         node *from = dict->words[i];
477         if (from == nullptr)
478             continue;
479         if (sizes == 0)
480             sizes = from->word.size();
481         for (size_t j = i + 1; j < dict->size; j++) {
482             node *to = dict->words[j];
483             if (to == nullptr)
484                 continue;
485             if (connected(from->word, to->word)) {
486                 dict->add_edge(from, to);
487             }
488         }
489     }
490     if (sizes != 0)
491         cout << "Processed " << sizes + 1 << " letter words" << endl;
492 }
493
494 void longest_path(hashTable *dict) {
495     int largest = 0;
496     vector < node * > reprs;
497     node *max = nullptr;
498     for (unsigned int i = 0; i < dict->size; i++) {
499         if (dict->words[i] != nullptr) {
500             if (find(reprs.begin(), reprs.end(),
501                     dict->words[i]->representative) == reprs.end()) {
502                 reprs.push_back(dict->words[i]->representative);
503                 int depth = dict->get_diameter(dict->words[i], false);
504                 if (depth > largest) {
505                     largest = depth;
506                     max = dict->words[i];
507                 }
508             }
509         }
510     }
511     node *origin = dict->get_diameter_node(max);
512     if (origin == nullptr || max == nullptr) {
513         cout << "No path found." << endl;
514         return;
515     }
516     dict->DFS(origin, max);
517     node *res = max;
518     ofstream file;
519     file.open("longest.txt", ios::app);
520     file << "Longest path for " << max->word.size() << " letter
521           words" << endl;

```

```

520     file << "Size: " << largest << endl;
521     while (res->parent != nullptr) {
522         file << res->word << " -> ";
523         res = res->parent;
524     }
525     file << res->word << endl;
526 }
527
528 int main() {
529     setlocale(LC_ALL, ".UTF8");
530     hashTable *dicts[_max_word_size_];
531     thread threads[_max_word_size_];
532     for (size_t i = 0; i < _max_word_size_; i++) {
533         dicts[i] = new hashTable;
534     }
535     ifstream in("wordlist-big-latest.txt");
536     if (!in) {
537         printf("Error: could not open words file\n");
538     }
539     string word;
540     while (in >> word) {
541         int size = word.size();
542         dicts[size - 1]->add(word);
543     }
544     for (int sizes = 0; sizes < _max_word_size_; sizes++) {
545         hashTable *dict = dicts[sizes];
546         threads[sizes] = thread(graph_builder, dict);
547     }
548     for (int sizes = 0; sizes < _max_word_size_; sizes++) {
549         threads[sizes].join();
550     }
551     path_finder(dicts, "etano", "sitie");
552 #ifdef _full_
553     ofstream file;
554     file.open("longest.txt", ios::trunc);
555     file.close();
556     for (int sizes = 0; sizes < _max_word_size_; sizes++)
557     {
558         hashTable *dict = dicts[sizes];
559         threads[sizes] = thread(longest_path, dict);
560     }
561     for (int sizes = 0; sizes < _max_word_size_; sizes++)
562     {
563         threads[sizes].join();
564     }
565 #endif
566     // connected_components(dicts, "belo");
567     longest(dicts, "etano");
568 }

```

```

569     // etano and sitia are opposite extremities of (one of the)
        main connected component, as they show up in lots of
        diameters
570 end(dict);
571 }

```

## 5.2 makefile

```

1  #
2  # makefile to compile the A.02 assignment (word ladder)
3  #
4
5  clean:
6      rm -rf a.out word_ladder *.exe
7
8  word_ladder: word_ladder.cpp
9      g++ -Wall -Wextra -O3 word_ladder.cpp -o word_ladder -lm -march=native
10
11 stats: word_ladder.cpp
12     g++ -Wall -Wextra -O3 word_ladder.cpp -o word_ladder -lm -march=native
        -D_stats_
13
14 detail: word_ladder.cpp
15     g++ -Wall -Wextra -O3 word_ladder.cpp -o word_ladder -lm -march=native
        -D_detail_
16
17 full: word_ladder.cpp
18     g++ -Wall -Wextra -O3 word_ladder.cpp -o word_ladder -lm -march=native
        -D_full_
19
20 debug: word_ladder.cpp
21     g++ -Wall -Wextra -O0 -ggdb3 word_ladder.cpp -o word_ladder -lm
        -march=native -D_full_

```

## 5.3 valgrind-out.txt

```

1      ==39494== Memcheck, a memory error detector
2  ==39494== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward
        et al.
3  ==39494== Using Valgrind-3.19.0-8d3c8034b8-20220411 and LibVEX;
        rerun with -h for copyright info
4  ==39494== Command: ./word_ladder
5  ==39494== Parent PID: 1689
6  ==39494==
7  --39494--
8  --39494-- Valgrind options:
9  --39494--   --leak-check=full
10 --39494--   --show-leak-kinds=all
11 --39494--   --verbose
12 --39494--   --log-file=valgrind-out.txt
13 --39494-- Contents of /proc/version:

```



```

14 --39494-- Linux version 6.1.1-zen1-1-zen (linux-zen@archlinux)
    (gcc (GCC) 12.2.0, GNU ld (GNU Binutils) 2.39.0) #1 ZEN SMP
    PREEMPT_DYNAMIC Wed, 21 Dec 2022 22:27:59 +0000
15 --39494--
16 --39494-- Arch and hwcaps: AMD64, LittleEndian,
    amd64-cx16-lzcnt-rdtscp-sse3-ssse3-avx-avx2-bmi-f16c-rdrand-rdseed
17 --39494-- Page sizes: currently 4096, max supported 4096
18 --39494-- Valgrind library directory: /usr/lib/valgrind
19 --39494-- Reading syms from
    /home/mycsina/Projects/Uni/AED/A02/A02/word_ladder
20 --39494-- Reading syms from /usr/lib/ld-linux-x86-64.so.2
21 ==39494== Downloading debug info for
    /usr/lib/ld-linux-x86-64.so.2...
22 --39494-- Considering
    /home/mycsina/.cache/debuginfod_client/22bd7a2c03d8cfc05ef7092bfae5932223189bc1/debuginfod
    ..
23 --39494-- .. CRC is valid
24 ==39494== Successfully downloaded debug file for
    /usr/lib/ld-linux-x86-64.so.2
25 --39494-- Reading syms from /usr/lib/valgrind/memcheck-amd64-linux
26 ==39494== Downloading debug info for
    /usr/lib/valgrind/memcheck-amd64-linux...
27 ==39494== Server query failed: No such file or directory
28 --39494-- object doesn't have a symbol table
29 --39494-- object doesn't have a dynamic symbol table
30 --39494-- Scheduler: using generic scheduler lock implementation.
31 --39494-- Reading suppressions file: /usr/lib/valgrind/default.supp
32 ==39494== embedded gdbserver: reading from
    /tmp/vgdb-pipe-from-vgdb-to-39494-by-mycsina-on-???
33 ==39494== embedded gdbserver: writing to
    /tmp/vgdb-pipe-to-vgdb-from-39494-by-mycsina-on-???
34 ==39494== embedded gdbserver: shared mem
    /tmp/vgdb-pipe-shared-mem-vgdb-39494-by-mycsina-on-???
35 ==39494==
36 ==39494== TO CONTROL THIS PROCESS USING vgdb (which you probably
37 ==39494== don't want to do, unless you know exactly what you're
    doing,
38 ==39494== or are doing some strange experiment):
39 ==39494== /usr/lib/valgrind/../../bin/vgdb --pid=39494
    ...command...
40 ==39494==
41 ==39494== TO DEBUG THIS PROCESS USING GDB: start GDB like this
42 ==39494== /path/to/gdb ./word_ladder
43 ==39494== and then give GDB the following command
44 ==39494== target remote | /usr/lib/valgrind/../../bin/vgdb
    --pid=39494
45 ==39494== --pid is optional if only one valgrind process is running
46 ==39494==
47 --39494-- REDIR: 0x40248f0 (ld-linux-x86-64.so.2:strlen) redirected

```

```

    to 0x580bd382 (???)
48 --39494-- REDIR: 0x40230a0 (ld-linux-x86-64.so.2:index) redirected
    to 0x580bd39c (???)
49 --39494-- Reading syms from
    /usr/lib/valgrind/vgpreload_core-amd64-linux.so
50 ==39494== Downloading debug info for
    /usr/lib/valgrind/vgpreload_core-amd64-linux.so...
51 ==39494== Server query failed: No such file or directory
52 --39494-- object doesn't have a symbol table
53 --39494-- Reading syms from
    /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so
54 ==39494== Downloading debug info for
    /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so...
55 ==39494== Server query failed: No such file or directory
56 --39494-- object doesn't have a symbol table
57 ==39494== WARNING: new redirection conflicts with existing --
    ignoring it
58 --39494-- old: 0x040248f0 (strlen ) R-> (0000.0)
    0x580bd382 ???
59 --39494-- new: 0x040248f0 (strlen ) R-> (2007.0)
    0x04847e20 strlen
60 --39494-- REDIR: 0x40232d0 (ld-linux-x86-64.so.2:strcmp) redirected
    to 0x4848d40 (strcmp)
61 --39494-- REDIR: 0x40224f0 (ld-linux-x86-64.so.2:mempcpy)
    redirected to 0x484c810 (mempcpy)
62 --39494-- Reading syms from /usr/lib/libstdc++.so.6.0.30
63 --39494-- Reading syms from /usr/lib/libm.so.6
64 ==39494== Downloading debug info for /usr/lib/libm.so.6...
65 --39494-- Considering
    /home/mycsina/.cache/debuginfod_client/2c8ff1d29b255da5b7371efd5caf57444d622838/debuginf
    ..
66 --39494-- .. CRC is valid
67 ==39494== Successfully downloaded debug file for /usr/lib/libm.so.6
68 --39494-- Reading syms from /usr/lib/libgcc_s.so.1
69 --39494-- Reading syms from /usr/lib/libc.so.6
70 ==39494== Downloading debug info for /usr/lib/libc.so.6...
71 --39494-- Considering
    /home/mycsina/.cache/debuginfod_client/1e94beb079e278ac4f2c8bce1f53091548ea1584/debuginf
    ..
72 --39494-- .. CRC is valid
73 ==39494== Successfully downloaded debug file for /usr/lib/libc.so.6
74 ==39494== WARNING: new redirection conflicts with existing --
    ignoring it
75 --39494-- old: 0x04c41270 (memalign ) R-> (1011.0)
    0x04847070 memalign
76 --39494-- new: 0x04c41270 (memalign ) R-> (1017.0)
    0x04847040 aligned_alloc
77 ==39494== WARNING: new redirection conflicts with existing --
    ignoring it

```

```

78 --39494--      old: 0x04c41270 (memalign          ) R-> (1011.0)
      0x04847070 memalign
79 --39494--      new: 0x04c41270 (memalign          ) R-> (1017.0)
      0x04847010 aligned_alloc
80 --39494-- REDIR: 0x4c47110 (libc.so.6:strncasecmp) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
81 --39494-- REDIR: 0x4c454d0 (libc.so.6:strchrnul) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
82 --39494-- REDIR: 0x4c445f0 (libc.so.6:memrchr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
83 --39494-- REDIR: 0x4c43c70 (libc.so.6:memcpy@@GLIBC_2.14)
      redirected to 0x48361c0 (_vgnU_ifunc_wrapper)
84 --39494-- REDIR: 0x4c58f70 (libc.so.6:wcslen) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
85 --39494-- REDIR: 0x4c5a720 (libc.so.6:wcsnlen) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
86 --39494-- REDIR: 0x4c47420 (libc.so.6:strnlen) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
87 --39494-- REDIR: 0x4c474b0 (libc.so.6:strpbrk) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
88 --39494-- REDIR: 0x4c45560 (libc.so.6:strcmp) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
89 --39494-- REDIR: 0x4c44680 (libc.so.6:memset) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
90 --39494-- REDIR: 0x4c58d80 (libc.so.6:wcschr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
91 --39494-- REDIR: 0x4c45450 (libc.so.6:index) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
92 --39494-- REDIR: 0x4c474e0 (libc.so.6:rindex) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
93 --39494-- REDIR: 0x4c58e10 (libc.so.6:wscmp) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
94 --39494-- REDIR: 0x4c448d0 (libc.so.6:stpncpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
95 --39494-- REDIR: 0x4c593d0 (libc.so.6:wmemchr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
96 --39494-- REDIR: 0x4c472c0 (libc.so.6:strncmp) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
97 --39494-- REDIR: 0x4c44940 (libc.so.6:strcasecmp) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
98 --39494-- REDIR: 0x4c467d0 (libc.so.6:strcspn) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
99 --39494-- REDIR: 0x4c58ea0 (libc.so.6:wscpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
100 --39494-- REDIR: 0x4c453d0 (libc.so.6:strcat) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
101 --39494-- REDIR: 0x4c471b0 (libc.so.6:strncasecmp_l) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
102 --39494-- REDIR: 0x4c43b70 (libc.so.6:bcmp) redirected to 0x48361c0

```

```

(_vgnU_ifunc_wrapper)
103 --39494-- REDIR: 0x4c46750 (libc.so.6:strcpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
104 --39494-- REDIR: 0x4c449e0 (libc.so.6:strcasecmp_l) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
105 --39494-- REDIR: 0x4c47080 (libc.so.6:strlen) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
106 --39494-- REDIR: 0x4c47360 (libc.so.6:strncpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
107 --39494-- REDIR: 0x4c44850 (libc.so.6:stpcpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
108 --39494-- REDIR: 0x4c443b0 (libc.so.6:memmove) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
109 ==39494== Preferring higher priority redirection:
110 --39494--   old: 0x04cfd840 (__memcpy_avx_unalign) R-> (2018.0)
      0x0484a040 __memcpy_avx_unaligned_erms
111 --39494--   new: 0x04cfd840 (__memcpy_avx_unalign) R-> (2018.1)
      0x0484b910 memmove
112 --39494-- REDIR: 0x4c43ae0 (libc.so.6:memchr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
113 --39494-- REDIR: 0x4c476a0 (libc.so.6:strspn) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
114 --39494-- REDIR: 0x4c444d0 (libc.so.6:mempcpy) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
115 --39494-- REDIR: 0x4c44780 (libc.so.6:rawmemchr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
116 --39494-- REDIR: 0x4c47e50 (libc.so.6:strstr) redirected to
      0x48361c0 (_vgnU_ifunc_wrapper)
117 --39494-- REDIR: 0x4d03930 (libc.so.6:__strchr_avx2) redirected to
      0x4847800 (rindex)
118 --39494-- REDIR: 0x4c40590 (libc.so.6:malloc) redirected to
      0x4841810 (malloc)
119 --39494-- REDIR: 0x4d00fe0 (libc.so.6:__strlen_avx2) redirected to
      0x4847d00 (strlen)
120 --39494-- REDIR: 0x4cfd0e0 (libc.so.6:__memcpy_avx2_movbe)
      redirected to 0x484b0c0 (bcmp)
121 --39494-- REDIR: 0x4d006f0 (libc.so.6:__strcmp_avx2) redirected to
      0x4848c40 (strcmp)
122 --39494-- REDIR: 0x4d026b0 (libc.so.6:__strncmp_avx2) redirected to
      0x4848450 (strncmp)
123 --39494-- REDIR: 0x4d002c0 (libc.so.6:__strchr_avx2) redirected to
      0x48479e0 (index)
124 --39494-- REDIR: 0x4cfce40 (libc.so.6:__memchr_avx2) redirected to
      0x4848dc0 (memchr)
125 --39494-- REDIR: 0x4d00500 (libc.so.6:__strchrnul_avx2) redirected
      to 0x484c300 (strchrnul)
126 --39494-- REDIR: 0x4cfd800 (libc.so.6:__mempcpy_avx_unaligned_erms)
      redirected to 0x484c410 (mempcpy)
127 --39494-- REDIR: 0x4cfd840 (libc.so.6:__memcpy_avx_unaligned_erms)

```

```

    redirected to 0x484b910 (memmove)
128 --39494-- REDIR: 0x4c40b30 (libc.so.6:free) redirected to 0x4844200
    (free)
129 --39494-- REDIR: 0x4c40d70 (libc.so.6:realloc) redirected to
    0x4846c40 (realloc)
130 --39494-- REDIR: 0x4cff240 (libc.so.6:__strcasecmp_l_avx2)
    redirected to 0x4848850 (strcasecmp_l)
131 --39494-- REDIR: 0x4cfe4f0 (libc.so.6:__stpcpy_avx2) redirected to
    0x484b1a0 (stpcpy)
132 --39494-- REDIR: 0x4911460 (libstdc++.so.6:operator new(unsigned
    long)) redirected to 0x4841f90 (operator new(unsigned long))
133 --39494-- REDIR: 0x49114c0 (libstdc++.so.6:operator new[](unsigned
    long)) redirected to 0x4843270 (operator new[](unsigned long))
134 --39494-- REDIR: 0x4c41340 (libc.so.6:calloc) redirected to
    0x48469c0 (calloc)
135 --39494-- REDIR: 0x490f6e0 (libstdc++.so.6:operator delete(void*,
    unsigned long)) redirected to 0x4844af0 (operator delete(void*,
    unsigned long))
136 --39494-- REDIR: 0x490f700 (libstdc++.so.6:operator
    delete[](void*)) redirected to 0x4845a10 (operator
    delete[](void*))
137 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
138 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
139 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
140 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
141 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
142 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
143 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
144 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
145 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
146 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
147 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
148 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
149 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
150 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
151 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
152 --39494-- memcheck GC: 1000 nodes, 0 survivors (0.0%)
153 ==39494==
154 ==39494== HEAP SUMMARY:
155 ==39494==   in use at exit: 0 bytes in 0 blocks
156 ==39494== total heap usage: 10,831 allocs, 10,831 frees,
    17,971,237 bytes allocated
157 ==39494==
158 ==39494== All heap blocks were freed -- no leaks are possible
159 ==39494==
160 ==39494== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0
    from 0)
161 Footer

```

## 5.4 longest.txt

Longest path for 1 letter words Diameter: 1 v -i x Longest path for 29 letter words Diameter: 1 constitucionalizar-vos-íamos -i constitucionalizar-nos-íamos Longest path for 28 letter words Diameter: 1 constitucionalizar-te-íamos -i constitucionalizar-me-íamos Longest path for 2 letter words Diameter: 30 PS -i PT -i BT -i BE -i CE -i CP -i GP -i GB -i kB -i km -i mm -i ma -i ia -i iv -i xv -i xx -i ex -i eh -i oh -i ou -i tu -i te -i de -i dr -i ar -i as -i vs -i vi -i li -i lo -i no Longest path for 27 letter words Diameter: 1 instrumentalizar-vos-íamos -i instrumentalizar-nos-íamos Longest path for 26 letter words Diameter: 2 corresponsabilizar-vos-ias -i corresponsabilizar-vos-iam -i corresponsabilizar-nos-iam Longest path for 3 letter words Diameter: 74 elo -i eco -i oco -i ovo -i ova -i ola -i ala -i ada -i ade -i ase -i use -i uso -i uno -i ano -i aro -i pro -i pio -i pia -i dia -i diz -i giz -i gim -i rim -i riu -i viu -i vii -i vai -i sai -i sal -i sul -i suo -i duo -i doo -i roo -i roa -i boa -i bom -i som -i sem -i nem -i num -i pum -i pus -i bus -i bis -i tis -i tos -i mos -i mor -i mar -i lar -i las -i ias -i iam -i Sam -i San -i Van -i Vaz -i paz -i pau -i tau -i teu -i meu -i mel -i fel -i fez -i foz -i foi -i fui -i rui -i rei -i dei -i der -i per -i pen Longest path for 25 letter words Diameter: 3 transcendentalizar-se-iam -i transcendentalizar-te-iam -i transcendentalizar-te-ias -i transcendentalizar-me-ias Longest path for 24 letter words Diameter: 3 contratestemunhar-me-ias -i contratestemunhar-te-ias -i contratestemunhar-te-iam -i contratestemunhar-se-iam Longest path for 4 letter words Diameter: 353 sras -i eras -i eram -i iram -i irem -i ires -i ores -i oves -i ovem -i ovam -i oval -i oral -i orai -i trai -i traz -i truz -i cruz -i crua -i frua -i flua -i alua -i alue -i alie -i avie -i avir -i agir -i agiu -i adiu -i adia -i afia -i afta -i apta -i apto -i alto -i alho -i olho -i olmo -i elmo -i ermo -i armo -i arfo -i arfe -i arpe -i arpa -i area -i arei -i grei -i geei -i geai -i ceai -i cear -i tear -i toar -i soar -i suar -i suor -i suou -i soou -i coou -i ceou -i geou -i grou -i arou -i aros -i atos -i ator -i atar -i atam -i adam -i adem -i ades -i ales -i alas -i asas -i asai -i usai -i usam -i usem -i unem -i unes -i unis -i unia -i unha -i unhe -i unge -i urge -i urre -i urra -i urda -i urdo -i urjo -i unjo -i anjo -i ando -i anda -i anca -i inca -i isca -i isco -i asco -i asso -i osso -i ouso -i ougo -i jugo -i jogo -i joga -i ioga -i ioda -i iode -i rode -i rodo -i lodo -i ledos -i vedo -i vejo -i veja -i reja -i ruja -i ruma -i duma -i duna -i puna -i pune -i pule -i pile -i pire -i tire -i tive -i uive -i uiuo -i vivo -i viva -i vila -i dila -i dilo -i digo -i ligo -i limo -i cimo -i cima -i cita -i cite -i dite -i dine -i fine -i fino -i nino -i nono -i novo -i nove -i nave -i cave -i cape -i tape -i taxe -i taxa -i tara -i para -i papa -i papo -i pado -i gado -i gajo -i gaja -i gafa -i rafa -i raia -i Gaia -i Gaza -i vaza -i vasa -i vaso -i raso -i ralo -i Palo -i Pato -i nato -i nata -i nada -i cada -i ceda -i cena -i lena -i lona -i loja -i boja -i boje -i bote -i pote -i pose -i cose -i come -i dome -i dobe -i dobo -i bobo -i boio -i bois -i dois -i does -i coes -i coei -i voei -i voem -i moem

-i miem -i mies -i fies -i fiei -i piei -i piai -i piam -i liam -i lias -i luas -i  
 ruas -i ruis -i ruim -i ruem -i suem -i suei -i subi -i suba -i juba -i jura -i  
 lura -i lusa -i musa -i mesa -i mexa -i mexo -i nexo -i nego -i sego -i sega  
 -i saga -i sala -i sola -i solo -i tolo -i toro -i moro -i miro -i mira -i miga  
 -i figa -i fixa -i rixa -i roxa -i rosa -i Rosa -i Roma -i toma -i tema -i teme  
 -i leme -i lume -i lute -i luto -i luxo -i laxo -i lavo -i lava -i lama -i mama  
 -i mana -i mane -i mate -i mete -i mede -i fede -i fedi -i feri -i fera -i feia  
 -i feio -i geio -i gero -i zero -i zelo -i pelo -i peco -i pico -i pica -i pisa -i  
 sisa -i siso -i biso -i bise -i base -i bafe -i bafo -i bano -i bani -i bali -i dali  
 -i deli -i devi -i reví -i regi -i rege -i ruge -i muge -i mugí -i muni -i muno  
 -i huno -i humo -i sumo -i sujo -i pujo -i pojo -i pomo -i gomo -i goto -i  
 gota -i dota -i dopa -i copa -i coca -i coco -i foco -i fofo -i fofa -i mofa -i  
 mofe -i mole -i fole -i fale -i gale -i gele -i gela -i nela -i nula -i fula -i fulo  
 -i furo -i faro -i saro -i sare -i vare -i vaie -i vais -i cais -i caio -i cabo -i  
 cubo -i tubo -i tufo -i tifo -i rifo -i rife -i rime Longest path for 23 letter  
 words Diameter: 8 desconstitucionalizarei -i desconstitucionalizarem -i des-  
 constitucionalizaram -i desconstitucionalizavam -i desconstitucionalizavas -i  
 desconstitucionalizadas -i desconstitucionalizados -i desconstitucionalizamos  
 -i desconstitucionalizemos Longest path for 22 letter words Diameter: 6 des-  
 governamentalizavam -i desgovernamentalizaram -i desgovernamentalizaras  
 -i desgovernamentalizadas -i desgovernamentalizados -i desgovernamentali-  
 zamos -i desgovernamentalizemos Longest path for 5 letter words Diameter:  
 1187 enche -i inche -i incha -i incra -i intra -i entra -i extra -i exara -i  
 exata -i exato -i exalo -i exulo -i exumo -i eximo -i exibo -i exhibe -i exile  
 -i exila -i axila -i afila -i affa -i affe -i affine -i afins -i afias -i aftas -i  
 altas -i altar -i aliar -i adiar -i odiar -i opiar -i opias -i opies -i optes -i  
 optem -i ontem -i untem -i unhem -i unhei -i unhai -i unhar -i untar -i  
 untas -i unias -i uniam -i unjam -i urjam -i urram -i urras -i erras -i eiras  
 -i liras -i lidas -i lidam -i limam -i limai -i ligai -i ligar -i migar -i migam  
 -i mijam -i mujam -i sujam -i subam -i cubam -i cubas -i cujas -i fujas  
 -i fumas -i fumes -i gumes -i guies -i guiem -i geiem -i gemem -i remem  
 -i regem -i reger -i reter -i meter -i metem -i matem -i macem -i maces  
 -i mames -i mamei -i mimei -i rimei -i ripei -i ripem -i rixem -i rixam -i  
 riram -i piram -i pilam -i pilai -i pilei -i pisei -i pisem -i sisem -i sises  
 -i bises -i bases -i banes -i banis -i bania -i banda -i panda -i panca -i  
 manca -i mansa -i massa -i massa -i mosse -i misse -i misso -i misto -i  
 mosto -i rosto -i rasto -i fasto -i facto -i pacto -i parto -i parlo -i parla -i  
 parva -i larva -i larga -i largo -i cargo -i carpo -i zarpo -i zarpa -i farpa -i  
 farta -i farte -i falte -i falhe -i talhe -i telhe -i telha -i tolha -i solha -i  
 solva -i solve -i salve -i salvo -i salgo -i salga -i salta -i Malta -i Malva  
 -i valva -i valsa -i falsa -i falia -i faliu -i baliu -i balir -i bulir -i bulia -i  
 bulha -i bulhe -i bolhe -i bolho -i rolho -i rilho -i filho -i filha -i milha -i  
 minha -i vinha -i vinda -i finda -i findo -i finco -i cinco -i cisco -i risco -i  
 risca -i rasca -i basca -i basco -i tasco -i talco -i palco -i polco -i poluo -i

polua -i polia -i polis -i polos -i povos -i novos -i novas -i noras -i foras -i  
 foram -i focam -i tocam -i togam -i togar -i vogar -i vogas -i jogas -i jogai  
 -i rogai -i regai -i recaí -i pecai -i pejai -i pejar -i penar -i penal -i renal  
 -i retal -i fetal -i fatal -i faval -i favas -i facas -i lacas -i lamas -i damas -i  
 dadas -i dados -i danos -i panos -i pinos -i sinos -i sidos -i ridos -i ricos -i  
 bicos -i bicas -i bicai -i bisai -i bisar -i pisar -i posar -i posas -i cosas -i  
 cosam -i cosem -i comem -i comei -i somei -i somai -i domai -i domam -i  
 dotam -i datam -i batam -i bagam -i bagar -i pagar -i pagas -i gagas -i  
 gigas -i digas -i dilas -i delas -i devas -i debes -i dever -i devir -i revir -i  
 reviu -i remiu -i remia -i regia -i regra -i regre -i regue -i legue -i ligue -i  
 migue -i migre -i mitre -i nitre -i nitro -i litro -i livro -i livra -i lavra -i  
 lacra -i lacro -i ladro -i ladre -i padre -i podre -i pobre -i dobre -i dobra  
 -i doera -i moera -i moela -i goela -i grela -i grega -i grego -i prego -i  
 predo -i prado -i orado -i ovado -i ovada -i ovava -i orava -i crava -i ceava  
 -i reava -i reavo -i reato -i meato -i meado -i miado -i miada -i piada -i  
 piava -i fiava -i fiara -i tiara -i toara -i soara -i sofra -i sofre -i cofre -i  
 cofie -i cofio -i copio -i copia -i copra -i coira -i coiro -i doiro -i doido -i  
 doida -i doada -i coada -i corda -i coroa -i coroo -i corvo -i sorvo -i sirvo  
 -i sirva -i sirza -i cirza -i cirze -i cinze -i cinde -i cindi -i cingi -i tingi -i  
 tinge -i finge -i finte -i fonte -i monte -i monge -i longe -i longo -i gongo  
 -i gingo -i mingo -i mango -i manjo -i ranjo -i ranho -i ganho -i ganso -i  
 tanso -i tarso -i tarjo -i sarjo -i sarja -i surja -i surra -i burra -i birra -i  
 birro -i mirro -i murro -i curro -i curso -i cursa -i curta -i culta -i culpa -i  
 culpe -i cuspe -i custe -i suste -i susto -i surto -i surti -i sorti -i sorte -i  
 norte -i noite -i coite -i coice -i foice -i force -i forco -i forno -i torno -i  
 terno -i termo -i teimo -i teixo -i deixo -i dei-o -i dei-a -i deu-a -i dou-a  
 -i dou-o -i douto -i couto -i chuto -i chupo -i chupe -i chape -i chame -i  
 chamo -i clamo -i clama -i clima -i clicla -i cliclo -i clivo -i crivo -i privo -i  
 priva -i prova -i trova -i trovo -i trono -i trino -i trine -i urine -i urina -i  
 crina -i china -i chita -i chata -i chaga -i chega -i checa -i cueca -i sueca  
 -i sueva -i suava -i suada -i suado -i soado -i soldo -i solto -i volto -i vulto  
 -i multo -i muito -i quito -i quita -i quina -i quine -i guine -i guise -i guiso  
 -i griso -i iriso -i irisa -i frisa -i frita -i frota -i brota -i brote -i trote -i  
 troce -i trace -i trave -i grave -i gravo -i grano -i grana -i grada -i arada  
 -i asada -i asado -i atado -i ataco -i ataca -i atava -i amava -i amova -i  
 amola -i atola -i atolo -i afole -i aforo -i afora -i agora -i agira -i agita -i  
 apita -i apipa -i apipo -i apupo -i apuro -i aparo -i avaro -i avara -i alara  
 -i alapa -i alapo -i alago -i alego -i adego -i adega -i adeja -i areja -i arejo  
 -i brejo -i breio -i breia -i creia -i crema -i creme -i treme -i tremo -i trepo  
 -i trepa -i tripa -i gripa -i gripe -i grite -i grete -i frete -i freto -i ereto -i  
 ereta -i preta -i prata -i trata -i traja -i trajo -i trago -i drago -i draga -i  
 Braga -i Brama -i brama -i bramo -i aramo -i arame -i afame -i afama -i  
 acama -i acaba -i acabe -i acate -i acato -i acaso -i acuso -i acuse -i acure  
 -i ature -i atire -i ative -i ativo -i atino -i atina -i atida -i avida -i avido



-i amido -i amigo -i amiga -i amiba -i ameba -i amena -i ameno -i ameio  
 -i ateio -i ateie -i apeie -i apeia -i apela -i anela -i anele -i anile -i anise -i  
 aniso -i anoso -i anajo -i anoja -i anota -i azota -i azote -i adote -i adoto  
 -i adito -i edito -i edita -i emita -i emite -i elite -i elide -i elude -i ilude -i  
 iludo -i aludo -i aludi -i aluei -i aguei -i aguai -i aguar -i atuar -i atuas -i  
 atues -i atuem -i anuem -i andem -i andam -i ardam -i arpam -i arpem -i  
 armem -i armei -i armai -i armar -i arfar -i arfas -i areas -i arees -i artes  
 -i antes -i entes -i estes -i esses -i asses -i assas -i ansas -i ancas -i incas -i  
 iscas -i iscos -i ascos -i arcos -i arcou -i areou -i ateou -i atuou -i aluou -i  
 aluiu -i fluui -i fluir -i fruir -i fruis -i freis -i ireis -i irais -i trais -i toais  
 -i doais -i doeis -i coeis -i ceeis -i ceais -i geais -i geris -i geriu -i gerou  
 -i gemou -i gamou -i gabou -i babou -i bafou -i safou -i sacou -i secou -i  
 secos -i selos -i zelos -i zelou -i melou -i meloa -i melra -i melro -i medro  
 -i cedro -i cetra -i cetra -i letra -i leira -i leiga -i veiga -i verga -i veria -i  
 viria -i viril -i viral -i virar -i oirar -i obrar -i obrai -i obrei -i obrem -i  
 abrem -i abres -i abris -i abria -i arria -i ardia -i urdia -i urdir -i urgir -i  
 ungir -i ungis -i unges -i urges -i urdes -i irdes -i iodes -i iodas -i iodar -i  
 rodar -i radar -i fadar -i fadam -i nadam -i nadem -i nades -i fades -i fales  
 -i falem -i fazem -i jazem -i jazam -i jazas -i vazas -i vasas -i rasas -i raias  
 -i raios -i rabos -i nabos -i natos -i gatos -i galos -i talos -i talas -i balas  
 -i bafas -i bufas -i bufam -i bufem -i bufei -i rufei -i rufai -i tufai -i tufar  
 -i lufar -i lunar -i lanar -i danar -i dinar -i ninar -i ninam -i tinam -i tinas  
 -i tonas -i topas -i topes -i tomes -i domes -i dotes -i notes -i notei -i rotei  
 -i rotem -i lotem -i lutem -i luzem -i luzam -i luxam -i luxai -i luxei -i  
 lixei -i lixes -i lixos -i lixou -i fixou -i ficou -i nicou -i ninou -i minou -i  
 mimou -i mimos -i vimos -i vemos -i remos -i regos -i cegos -i cepos -i  
 copos -i cocos -i cacos -i cavos -i cavou -i lavou -i levou -i nevou -i negou  
 -i pegou -i pejou -i pojou -i posou -i pisou -i pilou -i pulou -i puxou -i  
 puxos -i puxes -i punes -i munes -i munas -i mulas -i pulas -i putas -i  
 lutas -i lutos -i lusos -i fusos -i furos -i furou -i murou -i mudou -i mudos  
 -i modos -i godos -i gomos -i somos -i somou -i domou -i dotou -i rotou -i  
 rolou -i ralou -i rapou -i tapou -i tarou -i torou -i toros -i tiros -i tifos  
 -i tufos -i tufou -i rufou -i rumou -i rumor -i tumor -i temor -i tenor -i  
 menor -i menos -i fenos -i fetos -i tetos -i tesos -i tesas -i temas -i gemas  
 -i geias -i feias -i frias -i irias -i iriam -i triam -i troam -i troas -i troes -i  
 tries -i triei -i criei -i chiei -i chies -i chias -i chiam -i caiam -i caiai -i vaiiai  
 -i vaiei -i raiei -i rasei -i rasem -i rafem -i rafes -i rifes -i rifas -i rimas  
 -i rimar -i remar -i rezar -i rezam -i reiam -i leiam -i levam -i lavam -i  
 lavem -i lavei -i cavei -i catei -i cates -i cites -i fites -i fitas -i fitar -i filar  
 -i folar -i bolar -i botar -i cotar -i corar -i corai -i gorai -i gorei -i gozei -i  
 gozem -i gizem -i gizes -i gizes -i gabes -i gabem -i babem -i bebem -i  
 bebam -i bebas -i Tebas -i Texas -i mexas -i mexam -i vexam -i vetam -i  
 vetas -i vedas -i vedar -i velar -i valar -i valam -i galam -i galai -i galei  
 -i gelei -i selei -i sedei -i cedei -i cedem -i fedem -i fodem -i modem -i



provou -i prosou -i presou -i presos -i preses -i predes -i credes -i crepes  
 -i trepes -i trepei -i trepai -i tremai -i tremas -i cremas -i creias -i cheias  
 -i chegas -i chegai -i chagai -i chagar -i chamar -i chamam -i clamam -i  
 clamem -i clames -i claves -i clives -i crives -i crises -i irises -i irisei -i irisai  
 -i irisar -i frisar -i frisam -i fresam -i fresai -i presai -i predai -i predar -i  
 pregar -i pregas -i pragas -i dragas -i dragam -i tragam -i tragai -i trajai  
 -i trajei -i tracei -i tracem -i trazem -i trazes -i traves -i troves -i trovas -i  
 provas -i provar -i prover -i provem -i provim -i provia -i previa -i previu  
 -i premiu -i premir -i fremir -i fremis -i fremes -i fremem -i cremem -i  
 cremei -i premei -i primei -i privei -i privai -i privam -i crivam -i crivar -i  
 clivar -i clicar -i clicas -i plicas -i placas -i planas -i planos -i pianos -i  
 piados -i fiados -i fiamos -i fiamos -i viemos -i voemos -i coemos -i coamos  
 -i coados -i zoados -i zoadas -i voadas -i veadas -i meadas -i meados -i  
 meatos -i beatos -i boatos -i boatou -i bostou -i bastou -i pastou -i passou  
 -i pausou -i pousou -i poisou -i poisos -i coisos -i coisas -i coimas -i colmas  
 -i colmar -i calmar -i palmar -i palmam -i pasmam -i pasmai -i pasmei  
 -i pasmes -i palmes -i palpes -i palpos -i palpou -i palmou -i calmou -i  
 calmos -i caldos -i saldos -i saldou -i salgou -i galgou -i galeou -i goleou  
 -i golfou -i golfos -i golfas -i golfai -i golfei -i goleei -i galeei -i baleei -i  
 baseei -i faseei -i faseai -i gaseai -i gasear -i basear -i balear -i baldar -i  
 baldai -i saldai -i soldai -i soltai -i voltai -i voltam -i voltem -i volvem -i  
 volveu -i solveu -i sorveu -i sorves -i torves -i torces -i torcem -i torrem -i  
 borrem -i berrem -i serrem -i seriem -i series -i sedies -i sediei -i mediei -i  
 medrei -i medres -i madres -i padres -i podres -i pobres -i cobres -i coeres  
 -i roeres -i roerem -i moerem -i moeram -i moeras -i moelas -i goelas -i  
 goelam -i grelam -i gretam -i gretem -i gretes -i greles -i grelos -i grilos -i  
 grifos -i grafos -i grafou -i gradou -i aradou -i arador -i atador -i atados -i  
 atamos -i aramos -i iramos -i irados -i iradas -i iraras -i oraras -i oravas -i  
 aravas -i amavas -i amovas -i amoras -i adoras -i adoraí -i adotai -i anotai  
 -i anatai -i anafai -i anafar -i abafar -i abalar -i abalas -i abatas -i abatom  
 -i acatam -i acatem -i anatem -i anates -i anafes -i abafes -i abafem -i  
 abanem -i abanei -i abanai -i afanai -i afanam -i afagam -i apagam -i  
 apagai -i apegai -i apegar -i adegar -i adegas -i alegas -i alagas -i aladas -i  
 asadas -i usadas -i usavas -i usavam -i usaram -i usarem -i usarei -i asarei  
 -i afarei -i aforei -i afores -i aporem -i apoiem -i apoiam -i apeiam -i  
 apeias -i apenas -i acenas -i acenai -i acendi -i acenda -i agenda -i agendo  
 -i atendo -i atando -i amando -i amanho -i amanha -i amansa -i amassa -i  
 amossa -i amosso -i acosso -i acosse -i acesse -i atesse -i atasse -i arasse  
 -i irasse -i iraste -i traste -i triste -i trista -i crista -i crispa -i cripa -i  
 crespo -i crestos -i presto -i presta -i fresta -i fresca -i fresco -i frasco -i  
 franco -i tranco -i trinco -i brinco -i brinca -i brinda -i brinde -i blinde -i  
 alinde -i alinhe -i alinho -i azinho -i azinha -i avinha -i avinca -i avinco -i  
 avindo -i aviado -i adiado -i adiada -i aliada -i alhada -i achada -i achara  
 -i aclara -i aclama -i aclamo -i aclimo -i aclime -i acoime -i acoima -i

acoita -i amoita -i amonta -i aponta -i aponte -i aporte -i aborte -i aborde  
 -i abordo -i acordo -i acorda -i acorra -i acorre -i aforre -i aferre -i aferra  
 -i aferia -i adería -i adiria -i aviria -i avaria -i amaria -i amarra -i amarro  
 -i amargo -i alargo -i alarga -i alarma -i alarme -i alarde -i atarde -i aturde  
 -i aturdo -i atuado -i aguado -i aguada -i aguava -i agrava -i agrafa -i  
 agrafe -i agrade -i agride -i agrido -i abrido -i abrida -i abrira -i abeira  
 -i abeiro -i aberro -i aberto -i aberta -i alerta -i alerte -i aleite -i aleito  
 -i alento -i avento -i aventa -i atenta -i atente -i atinte -i atinge -i atingi  
 -i atinai -i atinar -i ativar -i ativer -i ativei -i avivei -i avives -i avivas -i  
 avisas -i anisas -i anisar -i animar -i animai -i amimai -i amimei -i amimes  
 -i animes -i animem -i anilem -i anilei -i anisei -i alisei -i alises -i alijes -i  
 alojés -i alojem -i alojam -i anojam -i anojas -i enojas -i enojai -i enojei -i  
 anojei -i amojei -i amolei -i amolai -i amolam -i atolam -i atolem -i atoles  
 -i atores -i atores -i aterei -i iterei -i iterai -i iteram -i ateram -i atiram  
 -i atiras -i aturas -i aturai -i amurai -i amarai -i amaram -i azaram -i  
 azarar -i aparar -i aparas -i aparos -i aparou -i amarou -i amurou -i acurou  
 -i acusou -i abusou -i abusos -i abuses -i abusem -i acusam  
 -i acusas -i acudas -i ajudas -i ajudam -i aludam -i iludam -i iludem -i  
 eludem -i eludes -i eludis -i aludis -i alueis -i amueis -i amuais -i anuais -i  
 andais -i ardaís -i ardeis -i areeis -i apeeis -i apeais -i ateais -i atraís -i  
 abrais -i obraís -i ourais -i jurais -i jureis -i fureis -i fumeis -i fumais -i  
 sumais -i sujais -i sujeis -i pujeis -i puxeis -i pixeis -i pileis -i fileis -i flais  
 -i ficais -i bicaís -i bisais -i cisais -i ciseis -i viseis -i viveis -i vivais -i vitais  
 -i vetais -i metais -i mexais -i mexeis -i meleis -i releis -i reveis -i deveis -i  
 devais -i demais -i temais -i temeis -i gemeis -i gameis -i gamais -i gabais  
 -i babais -i bagais -i pagais -i papais -i tapais -i tarais -i sarais -i sacais  
 -i socais -i vocais -i vogais -i togaís -i tonais -i toneis -i topeis -i dopeis  
 -i dobeis -i dobais -i dosais -i rosais -i rotais -i ratais -i latais -i lutais  
 -i luxais -i lixais -i rixais -i rimais -i rimeis -i mimeis -i mijeis -i mijais  
 -i migais -i sigais -i sinaís -i ninaís -i nineis -i naneis -i faneis -i fadeis  
 -i fedeis -i sedeis -i sedais -i selais -i pelais -i pulais -i punais -i munais  
 -i mungis -i munges -i monges -i montes -i pontes -i pondes -i pordes -i  
 bordes -i bordos -i bardos -i barcos -i marcos -i mancos -i mansos -i tansos  
 -i tangos -i tangou -i tangeu -i tangei -i rangei -i rangem -i rancem -i  
 dancem -i dances -i lances -i lancei -i lanhei -i lanhem -i lanham -i lanhar  
 -i banhar -i banhai -i banzai -i banzam -i baniam -i baliám -i buliam -i  
 bulias -i bulhas -i bulhar -i bilhar -i rilhar -i rilhas -i pilhas -i pilhes -i  
 pilhem -i rilhem -i rilhei -i rolhei -i molhei -i molhes -i folhes -i falhes -i  
 faltes -i faltei -i fartei -i fartai -i fartar -i faltar -i saltar -i saltam -i salgam  
 -i salgas -i salsas -i balsas -i bolsas -i bolsos -i bolsou -i bolhou -i rolhou  
 -i ralhou -i falhou -i faltou -i fartou -i fartos -i factos -i facho -i fechos -i  
 fechas -i fecham -i ficham -i fichem -i fiches -i biches -i boches -i coches -i  
 cochas -i tochas -i tolhas -i telhas -i telham -i tenham -i renham -i renhas  
 -i senhas -i sonhas -i sonham -i sondam -i rondam -i rondai -i rondei -i

mondei -i mordei -i mordem -i moldem -i toldem -i toldes -i soldes -i soltes  
-i soltos -i sultos -i sou-os -i dou-os -i deu-os -i deu-as -i deusas -i deuses  
-i reuses -i reusem -i reusam -i reusai -i ressei -i gessai -i gessas -i cessas  
-i cesses -i cessem -i dessem -i descem -i descei -i descai -i pescai -i pescar  
-i piscar -i biscar -i biscar -i riscar -i riscam -i discam -i dispam -i dispas  
-i distas -i mistas -i mistos -i vistos -i vastos -i rastos -i rostos -i gostos  
-i gestos -i leitos -i leitões -i leitor -i reitor -i reator -i reatou -i restou -i  
testou -i tentou -i sentou -i sentiu -i sentis -i mentis -i mentia -i meneia -i  
medeia -i medeio -i mede-o -i medi-o -i medico -i dedico -i dedica -i debica  
-i debita -i rebita -i recita -i recito -i receto -i recete -i remete -i remate  
-i remato -i remoto -i remota -i remove -i renova -i renove -i renome -i  
retome -i retomo -i retoco -i reboco -i reboca -i rebola -i recola -i recoza  
-i recozo -i recoso -i recose -i recase -i recave -i recavo -i recaio -i recais -i  
reiais -i leiais -i leais -i leseis -i peseis -i poseis -i poreis -i moreis -i mofeis  
-i mofais -i modais -i iodais -i iodeis -i rodeis -i rodeia -i rodeta -i rodete  
-i rolete -i colete -i colite -i colige -i coliga -i colina -i coluna -i comuna -i  
comuta -i comuto -i cometo -i coreto -i careto -i cateto -i catito -i catita -i  
capita -i capina -i cabina -i cabida -i cabido -i sabido -i subido -i sumido  
-i sumida -i sumira -i sugira -i sugara -i sugava -i sujava -i sujada -i sujado  
-i sugado -i ougado -i ousado -i ousada -i ousara -i ourara -i curara -i  
cubara -i cubana -i cubano -i cubado -i jubado -i jurado -i furado -i furada  
-i murada -i murava -i morava -i gorava -i gozava -i gozara -i gizara -i  
gizada -i gizado -i girado -i virado -i visado -i bisado -i bisada -i bicada -i  
bicava -i ficava -i focava -i locava -i locara -i tocara -i tonara -i tonava  
-i topava -i dopava -i dosava -i dosara -i dobara -i dobada -i lobada -i  
lobado -i locado -i cocado -i cocada -i cacada -i capada -i capara -i casara  
-i casava -i caiava -i vaiava -i valava -i falava -i fadava -i fadara -i fanara  
-i danara -i data -i ditara -i fitara -i fitada -i fitado -i ditado -i datado -i  
matado -i matada -i mamada -i mamava -i mimava -i minava -i ninava -i  
nanava -i panava -i pagava -i pagara -i vagara -i varara -i virara -i virava  
-i pirava -i pilava -i pelava -i pejava -i pojava -i pojara -i podara -i iodara  
-i iodada -i iodado -i podado -i posado -i posada -i pomada -i tomada -i  
tomado -i topado -i tapado -i tarado -i tarada -i talada -i talara -i ralara  
-i rafara -i rafava -i ratava -i rotava -i notava -i notara -i botara -i botada  
-i bolada -i bulada -i bufada -i bafada -i babada -i babado -i gabado -i  
galado -i ralado -i ramado -i ramudo -i rabudo -i rabujo -i rabuje -i rabule  
-i cabule -i cabula -i cabala -i cavala -i cavalo -i cavado -i cagado -i vagado  
-i vogado -i votado -i vetado -i velado -i velada -i velara -i gelara -i gerara  
-i gerira -i ferira -i ferida -i ferido -i fedido -i cedido -i cedida -i medida -i  
mexida -i mexido -i metido -i detido -i devido -i devida -i decida -i tecida  
-i temida -i temido -i remido -i relido -i relida -i retida -i retina -i retine -i  
refine -i refino -i refiro -i refira -i refila -i repila -i repisa -i repesa -i retesa  
-i retese -i reteve -i releve -i releve -i relega -i delega -i delego -i denego -i  
renego -i refego -i refugio -i refuto -i reduto -i reduzo -i reluzo -i reluzia -i

reluta -i relata -i delata -i delate -i debate -i debute -i debuto -i deuto -i depuro -i deparo -i reparo -i repare -i separe -i separa -i secara -i pecara -i pesara -i lesara -i legara -i legada -i regada -i regado -i regalo -i regulo -i regula -i regela -i revela -i revelo -i rebelo -i rabelo -i tabelo -i tabele -i tabefe -i tarefe -i tarefa -i tareca -i careca -i caneca -i caneco -i canelo -i capelo -i capela -i capeia -i mapeia -i mapeio -i mareio -i mareie -i rareie -i rareia -i raleia -i baleia -i baleie -i galeie -i goleie -i goteie -i loteie -i loteis -i coteis -i cociis -i caceis -i caleis -i calais -i valais -i vazais -i vazeis -i vareis -i vereis -i vereia -i sereia -i servia -i servis -i serzis -i cerzis -i cernis -i cernas -i cernam -i cercam -i cercai -i cerra -i ferrai -i ferrei -i feirei -i feires -i feixes -i deixes -i deixei -i deixai -i deitai -i deitam -i deitem -i dentem -i sentem -i sentei -i ventei -i vencei -i vencem -i vendem -i pendem -i pendam -i pensam -i pensas -i pencas -i percas -i perdas -i herdas -i hordas -i mordas -i mornas -i moinas -i moitas -i muitas -i quitas -i quitam -i quinam -i quinem -i quines -i guines -i guinei -i guisei -i guisem -i guisam -i guisas -i grisas -i brisas -i brigas -i brigai -i britai -i britei -i briter -i brotem -i trotem -i trotam -i trotai -i trocai -i trocar -i brocar -i brocas -i bromas -i brumas -i bruxas -i bruxos -i brutos -i brotos -i brotou -i britou -i fritou -i fritos -i fritas -i frutas -i trutas -i tratas -i tratar -i tramar -i grammar -i granar -i granas -i granes -i granem -i grafem -i grafei -i grafai -i gradai -i gradam -i gravam -i geavam -i geavas -i reavas -i relvas -i relvam -i relvem -i reavam -i reagem -i reages -i reates -i rentes -i lentes -i lenhes -i lenhos -i lenhou -i lanhou -i ganhou -i ganhos -i galhos -i galhas -i malhas -i manhas -i mandas -i mandam -i mangam -i mangar -i mancar -i mancai -i marcai -i marcam -i marram -i garram -i garras -i jarras -i jardas -i sardas -i sarjas -i sarjes -i sarjem -i sarnem -i sornem -i sornei -i tornei -i tornai -i tornar -i torvar -i turvar -i curvar -i curvas -i cursas -i curses -i curtes -i curtos -i certos -i cercos -i circos -i ciscos -i riscos -i riscou -i rosco -i foscou -i forcou -i forrou -i forros -i fornos -i tornos -i turnos -i turbos -i turbas -i turras -i zurras -i zurrai -i surrai -i surram -i supram -i suprem -i soprem -i sopres -i sopros -i soprou -i sobrou -i dobrou -i dourou -i lourou -i louvou -i louvor -i louvar -i louvas -i loucas -i moucas -i moucos -i mouros -i touros -i toiros -i loiros -i loires -i loirei -i doirei -i doarei -i zoarei -i zoarem -i coarem -i coaxem -i coaxe -i coaxai -i coaxas -i coaras -i toaras -i toaram -i voaram -i voavam -i doavam -i doavas -i soavas -i solvas -i solvam -i sorvam -i sirvam -i sirvas -i sirzas -i cirzas -i cirzes -i cirzem -i cinzem -i cindem -i cindes -i cinges -i finges -i fingem -i fintem -i finte -i fintai -i fincai -i vincai -i vincam -i vingam -i xingam -i xingas -i gingas -i gijas -i cinjas -i cinjam -i tinjam -i tanjam -i tanjas -i tantas -i santas -i sintas -i pintas -i pintar -i pintor -i pintou -i fintou -i findou -i fundou -i fundos -i fundis -i fundir -i fundar -i findar -i findas -i fendas -i tendas -i tendes -i tendeu -i rendeu -i rendou -i rondou -i mondou -i montou -i contou -i contos -i contas -i contar -i coutar -i coutam -i chutam -i chutai -i chutei -i chutes -i chutos -i cautos

-i lautos -i lautas -i lactas -i lactes -i lactei -i laceei -i lajeei -i lajeai -i  
 ladeai -i ladear -i ladrar -i lavrar -i lavrai -i lacrai -i lucrai -i lucram -i  
 lucrem -i lucrez -i luarez -i guares -i guaris -i guaria -i gearia -i cearia  
 -i coaria -i soaria -i sorria -i sorrir -i sortir -i surtir -i surdir -i surdis -i  
 surgis -i surges -i surgem -i surtem -i sustem -i sustei -i sustai -i sustar -i  
 custar -i custas -i justas -i juntas -i juntas -i juntei -i jantei -i jantem -i  
 cantem -i cansem -i canses -i causes -i causei -i pausei -i pausem -i pousem  
 -i poupem -i poupei -i poupai -i poupar -i pousar -i pausar -i passar -i  
 pastar -i pastas -i partas -i portas -i portam -i portem -i cortem -i coroom  
 -i coroei -i correi -i jorrei -i jorres -i morres -i mirres -i migres -i migrem  
 -i miarem -i miarei -i fiarei -i fiares -i piares -i piores -i piorei -i piorai -i  
 pioram -i piaram -i fiaram -i fiavam -i miavam -i miavas -i miaras -i mitras  
 -i mitrar -i mirrar -i birrar -i barrar -i barbar -i barbas -i barbes -i barres  
 -i barris -i carris -i carros -i cargos -i cargas -i carpas -i carpam -i cardam  
 -i tardam -i tardem -i fardem -i fardes -i dardes -i derdes -i verdes -i vertes  
 -i vertas -i ventas -i ventar -i tentar -i testar -i testas -i tostar -i tostes  
 -i tostei -i teste -i restei -i reptei -i raptai -i raptas -i raptar -i  
 captar -i capear -i mapear -i mapeai -i tapeai -i tapeei -i tatei -i tatuei  
 -i tatues -i tatuas -i tatuar -i tatear -i ratear -i rarear -i rareai -i rarei -i  
 marei -i marrei -i varrei -i varrem -i variem -i vadiem -i radiem -i radiei  
 -i radiar -i radiar -i vadiar -i vadias -i vazias -i razias -i raziam -i faziam -i  
 fariam -i feriam -i geriam -i gemiam -i gemias -i temias -i terias -i termas  
 -i bermas -i berras -i serras -i serrar -i seriar -i sediar -i mediar -i mediam  
 -i pediam -i podiam -i poliam -i poluam -i poluas -i polpas -i pompas -i  
 rompas -i rampas -i raspas -i rascas -i roscas -i roscar -i foscas -i foscai -i  
 forcai -i forcam -i formam -i firmam -i firmar -i filmar -i filmas -i filias  
 -i folias -i folgas -i folgar -i folhar -i folhai -i falhai -i ralhai -i ralham -i  
 valham -i valsam -i valsar -i valvar -i valvas -i calvas -i calvai -i calvei -i  
 calvem -i calhem -i colhem -i colher -i tolher -i talher -i talhei -i tachei -i  
 tachim -i sachem -i saciem -i saciei -i saciai -i saciar -i sachar -i sachas -i  
 sacras -i safras -i sofras -i sofram -i sobram -i sobrar -i dobrar -i dobrar  
 -i doiras -i doiram -i coiram Longest path for 20 letter words Diameter:  
 10 emporcar-lhes-íamos -i emborcar-lhes-íamos -i emborrar-lhes-íamos -i  
 embarrar-lhes-íamos -i esbarrar-lhes-íamos -i escarrar-lhes-íamos -i escarpar-  
 lhes-íamos -i escalpar-lhes-íamos -i escalfar-lhes-íamos -i esfalfar-lhes-íamos  
 Longest path for 7 letter words Diameter: 1844 infetar -i infetai -i injetai  
 -i injetei -i injetem -i infetem -i inferem -i inserem -i inseres -i ingeres -i  
 ingeris -i ingerir -i inserir -i inseriu -i inferiu -i inferia -i inferna -i inverna  
 -i inverno -i inverto -i inverte -i invente -i intente -i intende -i intendi -i  
 entendi -i entenda -i enteada -i enteava -i entrava -i entrave -i entraxe -i  
 entrapo -i entrado -i estrada -i estrada -i estiada -i espiada -i espiara -i  
 espirra -i espirro -i esbirro -i embirro -i emborro -i emborco -i embarco -i  
 embargo -i embarga -i embarra -i esbarra -i escarra -i escarre -i escarne -i  
 encarne -i encarte -i encarto -i encardo -i encardi -i encara -i encara -i

encapar -i encapas -i encapes -i escapes -i escales -i escalem -i estalem -i  
 estalam -i estavam -i estivam -i estivem -i estirem -i estirei -i estarei -i  
 estares -i estafes -i estufes -i estufas -i estufai -i estofai -i estocai -i estacai  
 -i estacas -i esticas -i esticar -i estucar -i estudar -i estudam -i estudem  
 -i estudei -i escudei -i escutei -i escutem -i escutam -i escutar -i escusar  
 -i escusas -i escudas -i escadas -i escamas -i escamar -i escavar -i escavai  
 -i escavei -i escovei -i encovei -i encovem -i encovam -i encovas -i escovas  
 -i escoras -i escorai -i esporai -i esporei -i exporei -i exporem -i expirem  
 -i expires -i expiras -i espiras -i aspiras -i aspiram -i assiram -i assaram  
 -i assacam -i assacar -i assapar -i assapas -i assadas -i assedas -i assedai  
 -i assedei -i assetei -i assetem -i asserem -i asseres -i asseris -i asseria -i  
 assaria -i ossaria -i ousaria -i ougaria -i sugaria -i sujaria -i pujaria -i  
 pojaria -i podaria -i rodaria -i rolaria -i bolaria -i botaria -i lotaria -i  
 locaria -i focaria -i ficaria -i bicaria -i bisaria -i pisaria -i piraria -i miraria  
 -i muraria -i furaria -i fumaria -i rumaria -i rimaria -i rixaria -i lixaria  
 -i ligaria -i legaria -i pegaria -i pagaria -i bagaria -i babaria -i gabaria  
 -i galaria -i calaria -i cacaria -i sacaria -i sararia -i tararia -i taparia -i  
 raparia -i rafaria -i refaria -i referia -i reveria -i reversa -i reverso -i revento  
 -i reverti -i revesti -i reveste -i de veste -i devaste -i degaste -i segaste  
 -i secaste -i sacaste -i sacasse -i sarasse -i tarasse -i taraste -i toraste -i  
 goraste -i gorasse -i corasse -i curasse -i durasse -i duraste -i muraste -i  
 miraste -i mirante -i girante -i garante -i galante -i galaste -i ralaste -i  
 rasaste -i rasasse -i rapasse -i papasse -i pagasse -i pagaste -i vagaste -i  
 vogaste -i jogaste -i jogasse -i togasse -i topasse -i dopasse -i dosasse -i  
 posasse -i pesasse -i lesasse -i levasse -i lavasse -i cavasse -i catasse -i  
 datasse -i danasse -i fanasse -i falasse -i falisse -i balisse -i baliste -i buliste  
 -i buli-te -i bulo-te -i bulo-me -i bula-me -i bula-se -i bulasse -i bolasse  
 -i rolasse -i rotasse -i lotasse -i locasse -i locaste -i cocaste -i cotaste -i  
 botaste -i bojaste -i pojaste -i pojante -i pujante -i puxante -i puxaste -i  
 pulaste -i pelaste -i melaste -i melasse -i gelasse -i gemasse -i gemesse -i  
 temesse -i temeste -i temente -i demente -i decente -i recente -i rebente -i  
 rebenta -i sebenta -i sebento -i sedento -i sedente -i sede-te -i pede-te -i  
 pedi-te -i pedinte -i pedante -i pecante -i picante -i picaste -i bicaste -i  
 bicasse -i nicasse -i ninasse -i ninaste -i finaste -i fixaste -i fixasse -i rixasse  
 -i rifasse -i rufasse -i rufaste -i rumaste -i rimaste -i ripaste -i repaste -i  
 repasta -i reposta -i recosta -i recosto -i retosto -i retorto -i reporto -i  
 reparto -i reparti -i reparai -i deparai -i deparar -i depurar -i depuras -i  
 deputas -i debutas -i debuxas -i debuxai -i debuxei -i debuxem -i debutem  
 -i debatem -i debateu -i rebateu -i rebatei -i rebitei -i rebitai -i rebitar -i  
 debitar -i debicar -i dedicar -i dedicas -i medicas -i medi-as -i mede-as -i  
 sede-as -i sede-os -i vede-os -i vedemos -i fedemos -i fademos -i faremos -i  
 paremos -i pare-os -i pare-as -i pareias -i tareias -i tarefas -i carecas -i  
 canecas -i canetas -i manetas -i maletas -i valetas -i varetas -i varejas -i  
 varejam -i farejam -i farejem -i farejei -i marejei -i marejes -i mareies -i



rareies -i rareiem -i pareiem -i parecem -i parecer -i perecer -i pereces -i  
 perenes -i serenes -i serenos -i seremos -i leremos -i lesemos -i lesamos -i  
 levamos -i levados -i levadas -i lavadas -i cavadas -i cavalas -i cavalos -i  
 cavamos -i casamos -i cisamos -i visamos -i visemos -i viremos -i riremos -i  
 rimemos -i limemos -i lixemos -i luxemos -i lutemos -i lotemos -i rotemos -i  
 rodemos -i rodamos -i fodamos -i focamos -i tocamos -i tonamos -i tonados  
 -i tomados -i somados -i somamos -i soviemos -i movemos -i  
 mofemos -i mofamos -i mofados -i mofadas -i moradas -i moraras -i goraras  
 -i gorares -i gorarem -i gerarem -i gemarem -i gemares -i gamares -i gabares  
 -i gabarei -i galarei -i talarei -i talaes -i taxares -i taxarem -i taxaram  
 -i taxavam -i tapavam -i papavam -i panavam -i sanavam -i sacavam -i  
 secavam -i recavam -i recaiam -i recriam -i recrias -i recries -i receies -i  
 recetes -i recetei -i recebei -i recebem -i recebam -i recetam -i remetam -i  
 remetas -i remedas -i remedar -i remedir -i remedia -i remetia -i repetia  
 -i repetiu -i repeliu -i repelis -i repeles -i remeles -i remelem -i revelem  
 -i revezem -i revezei -i revezai -i revezar -i revelar -i rebelar -i debelar -i  
 degelar -i degelam -i degolam -i degolai -i desolai -i desovai -i desovei -i  
 demovei -i demoves -i demores -i demoras -i devoras -i devoram -i decoram  
 -i decorai -i decorri -i recorri -i recobri -i recobra -i recubra -i recuara -i  
 recuada -i reguada -i regrada -i regrado -i regravado -i retravo -i retrava -i  
 retraia -i retrais -i regrais -i regreis -i regueis -i legueis -i ligueis -i migueis  
 -i migreis -i mirreis -i birreis -i barreis -i varreis -i varieis -i carieis -i cariais  
 -i cardais -i pardais -i pareais -i mareais -i marcais -i mascas -i pascais  
 -i pescais -i descais -i descaia -i descara -i discara -i distara -i distava -i  
 listava -i listada -i lestada -i cestada -i custada -i sustada -i sustida -i  
 sustido -i surtido -i surgido -i surgida -i surgira -i surtira -i sortira -i sorrira  
 -i sorrida -i sorrido -i corrido -i corrijo -i corrija -i correja -i correta -i  
 correto -i correio -i correis -i forreis -i forceis -i forcais -i foscais -i fiscais -i  
 fincais -i findais -i fendais -i vendais -i venhais -i lenhais -i lenheis -i lenteis  
 -i tenteis -i tendeis -i rendeis -i rondeis -i sondeis -i sondaes -i soldais -i  
 saldaes -i salgais -i galgaes -i galeais -i gaseais -i baseais -i baseeis -i basteis  
 -i pasteis -i passeis -i pauseis -i pausais -i pousais -i possais -i postais -i  
 portais -i porteis -i porteio -i ponteio -i penteio -i penteie -i panteie -i  
 panteia -i pandeia -i candeia -i candeio -i bandeio -i baldeio -i baldeis -i  
 caldeis -i calmeis -i calmais -i colmais -i colhais -i rolhais -i rilhais -i pilhais  
 -i pinhais -i pingais -i mingais -i mangais -i manjais -i tanjais -i tarjais -i  
 sarjais -i sarnais -i sornais -i tornais -i torvais -i torveis -i turveis -i curveis  
 -i curseis -i cursais -i curtais -i furtais -i fartaes -i faltaes -i falhais -i talhais  
 -i tachais -i sachais -i sacheis -i racheis -i ralheis -i malheis -i molheis -i  
 moldeis -i mordeis -i mordais -i bordais -i borrais -i berrais -i serraes -i  
 surrais -i supraes -i sopraes -i sopraes -i soareis -i toareis -i trareis -i traceis  
 -i troceis -i troteis -i broteis -i brotais -i bromais -i aromais -i aramais -i  
 afamais -i afagais -i afogais -i afolais -i afoleis -i afileis -i affeis -i affais -i  
 afirais -i adirais -i adireis -i aditeis -i adoteis -i azoteis -i azotais -i anotais

-i anatais -i abatais -i abanais -i abonais -i atonais -i atoneis -i atineis -i  
 ativeis -i ativais -i avivais -i avisais -i aviseis -i aliseis -i alijeis -i alijais -i  
 alojais -i amojais -i amorais -i amurais -i aturais -i aturdis -i aturdas -i  
 aturdam -i atardam -i atardem -i atardes -i amardes -i amarres -i amarrem  
 -i amarram -i amariam -i arariam -i ararias -i afarias -i afasias -i afastas  
 -i afastam -i abastam -i abastem -i abastes -i asastes -i asasses -i atasses  
 -i atesses -i atessem -i acessem -i acessei -i acessai -i acessar -i avessar -i  
 avessas -i avessos -i acessos -i acessou -i acossou -i acostou -i apostou -i  
 apontou -i atontou -i atentou -i atendeu -i atendem -i agendem  
 -i agendas -i agentes -i alentes -i aleites -i aceites -i aceitem -i azeitem -i  
 azeitei -i azeitai -i aceitai -i aceitar -i ajeitar -i ajeitas -i afeitas -i aflitas  
 -i aflitos -i afoitos -i afoites -i amoites -i amontes -i amantes -i amanses -i  
 amansas -i amanhas -i apanhas -i apanhes -i apanhei -i apinhei -i apilhei -i  
 afilhei -i afilhem -i afiliem -i afiliam -i aflias -i afilhas -i apilhas -i apilham  
 -i anilham -i anilhai -i aninhai -i alinhei -i alinhas -i atinhas -i atinjas -i  
 atinjam -i atintam -i atintai -i atontai -i amontai -i amountam -i apontam  
 -i apontem -i aponte -i aporte -i apportes -i apartes -i apartas -i apartar  
 -i acartar -i acartam -i acertam -i apertam -i apertai -i alertai -i alentai -i  
 alentam -i aventam -i aventem -i adentem -i adentei -i adensei -i adenses -i  
 adeuses -i adeusem -i adeusam -i adeusar -i adensar -i adentar -i adentas -i  
 atentas -i atestas -i arestas -i crestas -i cristas -i tristas -i tristes -i trastes  
 -i toastes -i coastes -i ceastes -i geastes -i geasses -i grasses -i irasses -i  
 irassem -i arassem -i amassem -i amassei -i amossei -i apossei -i apossem  
 -i apossam -i apossar -i apostar -i acostar -i acostas -i acoitas -i acoitam -i  
 acoimam -i acoimem -i acoimei -i aclimei -i aclamei -i aclames -i aclares -i  
 achares -i achates -i achatem -i achatam -i achavam -i achavas -i achadas -i  
 alhadas -i aluadas -i aluados -i amuados -i amuamos -i atuamos -i atuemos  
 -i aluemos -i alieiros -i adieiros -i adirmos -i avirmos -i aviamos -i afiamos  
 -i afiados -i afiadas -i aviadas -i aviaras -i aviaram -i adiaras -i odiaram -i  
 odiavam -i opiavam -i optavam -i optavas -i optaras -i opiaras -i opiases -i  
 odiases -i adiares -i afiares -i afiares -i afiares -i afiares -i aluarei -i aluirei  
 -i aloirei -i agoirei -i agoirai -i agourai -i alourai -i alouram -i alousam -i  
 alousas -i aloucas -i aloucar -i amoucar -i amourar -i amouras -i amoiras  
 -i amoiram -i amoirem -i amourem -i agourem -i agoures -i aloures -i  
 algures -i augures -i augurem -i augirem -i augirei -i tugirei -i tugires -i  
 fugires -i fugiras -i rugiras -i rugiram -i mugiram -i muniram -i munirem -i  
 punirem -i punires -i puniras -i punidas -i punidos -i punimos -i punamos  
 -i munamos -i mudamos -i mudados -i murados -i jurados -i juramos -i  
 ouramos -i obramos -i obrados -i obradas -i obravas -i obravam -i ouravam  
 -i duravam -i duraram -i muraram -i miraram -i migaram -i migarem -i  
 minarem -i ninarem -i ninaram -i finaram -i fitaram -i fitaras -i fitavas  
 -i citavas -i cisavas -i sisavas -i sisavam -i bisavam -i bicavam -i ficavam  
 -i fixavam -i lixavam -i lixavas -i lidavas -i lidaras -i lideras -i lideres -i  
 liderei -i lidarei -i lidarem -i lixarem -i rixarem -i rixarei -i rifarei -i rifares

-i rufares -i bufares -i bulares -i bularei -i pularei -i pujarei -i pujarem -i  
 puxarem -i puxaram -i pularam -i pularas -i pulavas -i puxavas -i puxadas  
 -i luxadas -i luxaras -i lufaras -i lufavas -i lufavam -i tufavam -i tufaram  
 -i tufarem -i tufarei -i lufarei -i lutarei -i lutarem -i lutaram -i lotaram  
 -i notaram -i notavam -i dotavam -i dopavam -i dopavas -i domavas -i  
 somavas -i somavam -i sovavam -i sovaram -i sovaras -i socaras -i sacaras  
 -i sacares -i sacarem -i sararem -i sararei -i vararei -i vazarei -i vazares -i  
 vazaras -i vararas -i vararam -i pararam -i pariram -i parirem -i parires  
 -i parares -i panares -i danares -i dinares -i finares -i finirei -i ficarei -i  
 ficarem -i filarem -i falarem -i ralarem -i rolarem -i rolarei -i colarei -i  
 cocarei -i locarei -i locares -i tocarei -i togares -i vogares -i vogaras -i  
 vogavas -i jogavas -i jogavam -i jogaram -i rogaram -i rodaram -i rodavam  
 -i iodavam -i iodavas -i iodadas -i podadas -i podidas -i polidas -i solidas -i  
 solidai -i solidei -i solidem -i colidem -i colides -i colidis -i colidir -i coligir  
 -i coligar -i coligas -i colinas -i colunas -i comunas -i comutas -i comutar -i  
 cometar -i cometam -i comeram -i cozeram -i cozerem -i cozerei -i comerei  
 -i comecei -i comecem -i comedem -i comedes -i cometes -i coletes -i roletes  
 -i roletas -i rodetas -i rodela -i modelas -i moderas -i foderas -i federas -i  
 federes -i federem -i foderem -i poderem -i poderes -i podares -i polares  
 -i pilares -i picares -i bicares -i bisares -i visares -i visarem -i pisarem -i  
 pisarei -i pesarei -i pesares -i pesaras -i posaras -i posavas -i posavam -i  
 pesavam -i lesavam -i lesavas -i legavas -i segavas -i segaras -i segaram -i  
 separam -i separem -i sedarem -i vedarem -i velarem -i velarei -i vexarei -i  
 vexares -i vetares -i vetaras -i vedaras -i vedadas -i vexadas -i vexados -i  
 velados -i melados -i melamos -i melemos -i pelemos -i pulemos -i pusemos  
 -i posemos -i cosemos -i cozemos -i gozemos -i gizemos -i fizemos -i fitemos  
 -i ditemos -i ditamos -i dilamos -i pilamos -i picamos -i nicamos -i nicados  
 -i bicados -i bicadas -i picadas -i pisadas -i visadas -i viradas -i viravas  
 -i viravam -i giravam -i gizavam -i gizavas -i gizadas -i gizados -i girados  
 -i girador -i gerador -i gelador -i pelador -i pesador -i pesados -i posados  
 -i dosados -i dopados -i dopamos -i dobamos -i dobemos -i domemos -i  
 tomemos -i topemos -i tapemos -i taxemos -i taxamos -i taramos -i tiramos  
 -i miramos -i migamos -i ligamos -i ligados -i ligadas -i migadas -i minadas  
 -i manadas -i manaras -i manaram -i mamaram -i mamarem -i mamarei -i  
 matarei -i matares -i ratares -i rasares -i rasaras -i raparas -i raparam -i  
 raiaram -i raiavam -i caiavam -i caiavas -i cagavas -i bagavas -i bagavam -i  
 babavam -i babaram -i bafaram -i bafarem -i bafarei -i bagarei -i pagarei  
 -i pagarem -i pegarem -i cegarem -i cegares -i cagares -i caiares -i caiarem  
 -i casarem -i casarei -i casalei -i cabalei -i cabulei -i cabules -i rabules  
 -i rabulas -i rebulas -i rebulis -i rebolis -i reboliu -i rebolou -i rebelou  
 -i regelou -i regalou -i regatou -i relatou -i delatou -i delator -i delatar  
 -i desatar -i desatas -i desates -i delates -i relates -i regates -i regatem  
 -i regatam -i regatai -i recatai -i recatas -i recitas -i repitas -i repisas -i  
 repisar -i repesar -i repesai -i repesei -i repisei -i repisem -i revisem -i

revises -i revezes -i reteses -i reteres -i meteres -i meterei -i deterer -i  
 deterem -i deferem -i diferem -i diferes -i dizeres -i fizeres -i fazeres -i  
 lazeres -i laceres -i lacerei -i macerei -i macerai -i macerar -i lacerar -i  
 laceram -i laceiam -i laceias -i ladeias -i cadeias -i cadelas -i capelas -i  
 lapelas -i lamelas -i gamelas -i gazelas -i gazeias -i gazeies -i gaseies -i  
 baseies -i baseiem -i baseiam -i gaseiam -i galeiam -i raleiam -i raleias  
 -i baleias -i balelas -i bacelas -i bacelar -i bacilar -i vacilar -i vacilai -i  
 vacilei -i vaciles -i vacines -i vacinas -i varinas -i marinas -i marinar -i  
 matinar -i motinar -i motinam -i motinem -i motinei -i matinei -i matizei  
 -i matizes -i batizes -i balizes -i balizei -i balizai -i balizam -i baliram -i  
 baniram -i ganiram -i ganiras -i ganidas -i ganidos -i ganimos -i banimos  
 -i balimos -i balidos -i bulidos -i bulados -i pulados -i pujados -i sujados -i  
 sujamos -i sejamos -i rejamos -i regamos -i regemos -i revemos -i devemos  
 -i devimos -i devidos -i detidos -i detidas -i metidas -i mexidas -i mexidos  
 -i medidos -i cedidos -i cedidas -i fedidas -i feridas -i geridas -i gemidas  
 -i gemidos -i remidos -i relidos -i relidas -i religas -i religai -i relegai -i  
 relegar -i relevar -i relevas -i releras -i regeras -i regerar -i regirar -i remirar  
 -i remiras -i refiras -i defiras -i definas -i defines -i definem -i refinem -i  
 refilem -i refilem -i refilai -i refinai -i refinar -i resinar -i resinas -i retinas  
 -i retinam -i retiram -i retirem -i regirem -i regerem -i regerei -i regarei -i  
 rezarei -i rezarem -i rezaram -i rezaras -i rezadas -i rezados -i rezador -i  
 regador -i negador -i negados -i segados -i senados -i senador -i secador -i  
 sacador -i sacados -i safados -i safadas -i safavas -i saravas -i taravas -i  
 taradas -i tapadas -i topadas -i copadas -i cocadas -i locadas -i locavas -i  
 lotavas -i rotavas -i rotaras -i dotaras -i dotares -i dotarem -i dobarem -i  
 dobarei -i domarei -i tomarei -i toparei -i toparem -i toparam -i toraram  
 -i toravam -i coravam -i colavam -i bolavam -i bojavam -i bojaram -i  
 bojaras -i bolaras -i colaras -i calaras -i galaras -i gelaras -i gelaram -i  
 gelavam -i zelavam -i zelavas -i zeladas -i peladas -i pegadas -i pagadas -i  
 vagadas -i vaiadas -i vaiados -i varados -i varador -i valador -i ralador -i  
 rapador -i capador -i capados -i calados -i galados -i gamados -i gamadas  
 -i gabadas -i babadas -i baladas -i raladas -i ratadas -i datadas -i datados  
 -i datador -i dotador -i notador -i notados -i votados -i votamos -i botamos  
 -i batamos -i babamos -i babemos -i cabemos -i caiemos -i cairmos -i  
 sairnos -i saiamos -i raíamos -i rafamos -i rifamos -i rixamos -i fixamos -i  
 fixados -i fixador -i rixador -i rimador -i rimados -i ripados -i ripadas -i  
 ripavas -i rifavas -i rifavam -i rifaram -i rimaram -i rimaras -i mimaras -i  
 mijaras -i mijares -i mirares -i murares -i durares -i durarei -i curarei -i  
 cubarei -i cubarem -i cubaram -i cubavam -i cubavas -i curavas -i curaras  
 -i furaras -i furadas -i fumadas -i fumavas -i rumavas -i remavas -i removas  
 -i remocas -i remocar -i rebocar -i rebocam -i rebolam -i rebolai -i recolai  
 -i recolhi -i recolhe -i refolhe -i refolho -i repolho -i reponho -i repondo -i  
 depondo -i dependo -i depende -i detende -i retende -i refende -i refenda -i  
 revenda -i revendo -i regendo -i regando -i rogando -i togando -i tocando

-i socando -i sacando -i safando -i rafando -i raiando -i vaiando -i valando  
-i falando -i fanando -i manando -i minando -i mimando -i limando -i  
lixando -i luxando -i lufando -i bufando -i bulando -i bolando -i bojando -i  
pojando -i pejando -i pecando -i picando -i pirando -i parando -i pagando  
-i cagando -i catando -i citando -i ditando -i dotando -i votando -i vetando  
-i vedando -i sedando -i selando -i gelando -i gerando -i gerindo -i ferindo  
-i feriado -i seriado -i seria-o -i seria-a -i seriara -i feriar -i feria -i  
ferrava -i feirava -i beirava -i beirara -i berrara -i berrada -i cerrada -i  
cercada -i cercado -i mercado -i marcado -i marrado -i marrada -i marrava  
-i barrava -i borrava -i borrata -i borrato -i borrado -i forrado -i forrada  
-i forrara -i formara -i formava -i forcava -i foscava -i roscava -i rascava -i  
lascava -i lassava -i lassara -i passara -i pausara -i causara -i cansara -i  
cansava -i cantava -i cintava -i cintada -i fintada -i fintado -i pintado -i  
pingado -i vingado -i vingada -i vincada -i vincara -i fincara -i findara -i  
fundara -i fundira -i fundida -i fundido -i fendido -i pendido -i pendida -i  
rendida -i rendada -i rendava -i vendava -i vendara -i ventara -i lentara -i  
lentava -i sentava -i sentada -i dentada -i dentado -i tentado -i testado -i  
restado -i reptado -i reptada -i reptara -i reatara -i reatava -i restava -i  
testava -i tostava -i tostara -i bostara -i bostada -i gostada -i gostado -i  
costado -i cortado -i cortada -i coutada -i coutara -i contara -i montara  
-i montava -i mondava -i mondada -i mondado -i sondado -i soldado -i  
toldado -i toldada -i toldava -i soldava -i soldara -i saldara -i baldara -i  
baleara -i galeara -i galeava -i gaseava -i gastava -i pastava -i pasmava -i  
pasmada -i pasmado -i palmado -i calmado -i caldado -i caldada -i calcada  
-i cascada -i cascado -i rascado -i riscado -i piscado -i piscada -i pispada -i  
pispava -i pingava -i gingava -i gingara -i mingara -i mangara -i mancara  
-i marcara -i mareara -i pareara -i pareava -i parlava -i parlada -i parlado  
-i pareado -i rareado -i raleado -i raleada -i rabeada -i rabeira -i rateira -i  
tateira -i tateada -i tapeada -i mapeada -i mapeava -i capeava -i capeira  
-i captara -i captada -i captado -i capeado -i cadeado -i ladeado -i laceado  
-i laceada -i lacrada -i lacrava -i ladrava -i ladeava -i ladeira -i ladeira -i  
padeira -i padeiro -i pateiro -i rateiro -i raleiro -i raleira -i caleira -i coleira  
-i coxeira -i coxeira -i roxeira -i roxeada -i rodeada -i rodeava Longest path  
for 19 letter words Diameter: 25 amoucar-lhes-íamos -i aloucar-lhes-íamos -i  
alourar-lhes-íamos -i agourar-lhes-íamos -i agoirar-lhes-íamos -i amoirar-lhes-  
íamos -i amoitar-lhes-íamos -i amontar-lhes-íamos -i apontar-lhes-íamos -i  
apostar-lhes-íamos -i acostar-lhes-íamos -i acossar-lhes-íamos -i amossar-lhes-  
íamos -i amassar-lhes-íamos -i amansar-lhes-íamos -i amanhar-lhes-íamos -i  
apanhar-lhes-íamos -i apinhar-lhes-íamos -i alinhar-lhes-íamos -i alindar-  
lhes-íamos -i blindar-lhes-íamos -i brindar-lhes-íamos -i brincar-lhes-íamos  
-i trincar-lhes-íamos -i truncar-lhes-íamos -i trunfar-lhes-íamos

## 5.5 Ficheiros em Python

## graph\_stats

January 10, 2023

```
[7]: %%capture
      %pip install seaborn
```

```
[8]: import re
      import sys
      import seaborn as sns
      import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt

      sns.set_style("darkgrid")

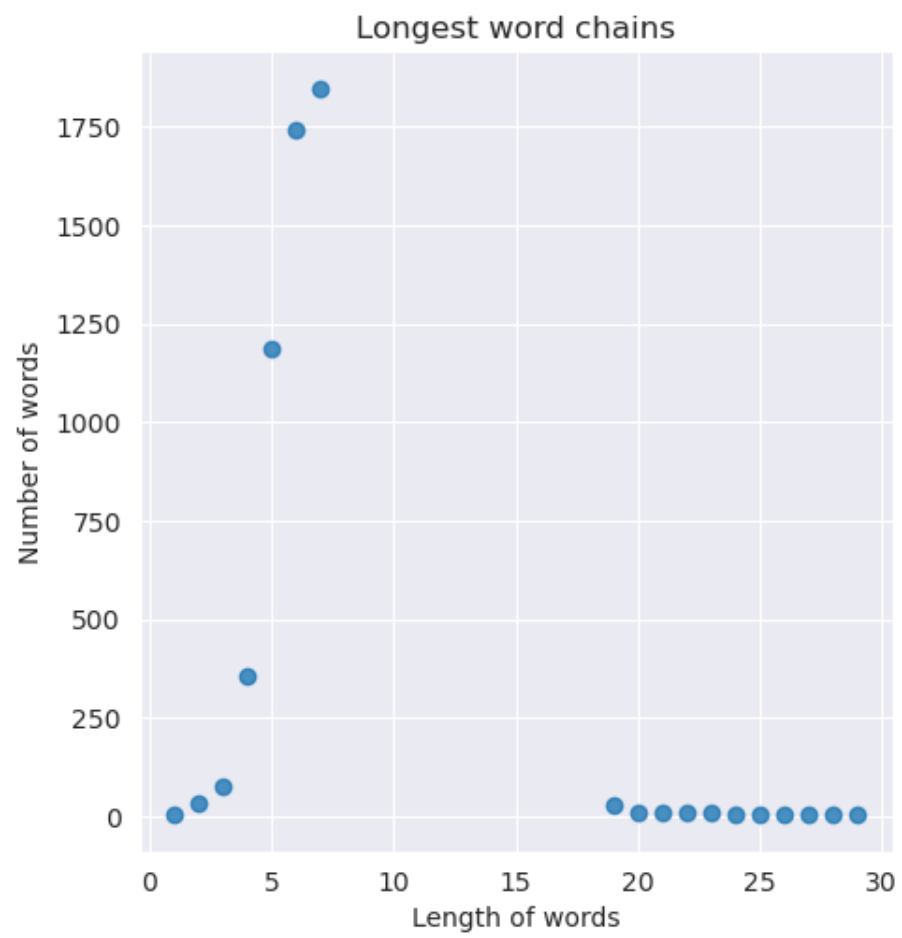
      class Chain():
          def __init__(self, length, words) -> None:
              self.length = length
              self.words = words
              self.size = len(words)
```

```
[9]: chains = []
      with open("../Server/longest.txt") as f:
          lines = f.readlines()
          for idx, line in enumerate(lines):
              if line.startswith("Longest"):
                  length = int(re.search(r"\d+", line)[0])
                  chain = Chain(length, lines[idx+2].strip().split(" -> "))
                  chains.append(chain)
```

```
[10]: df = pd.DataFrame([{"length": chain.length, "words": chain.words, "size": chain.
      ↪size} for chain in chains])
```

```
[11]: sns.lmplot(x="length", y="size", data=df, fit_reg=False)
      plt.xlabel("Length of words")
      plt.ylabel("Number of words")
      plt.title("Longest word chains")
```

```
[11]: Text(0.5, 1.0, 'Longest word chains')
```



```
[11]: Text(0.5, 1.0, 'Longest word chains')
```

## table\_stats

January 10, 2023

```
[10]: %%capture
      %pip install seaborn
```

```
[11]: import re
      import sys
      import seaborn as sns
      import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt

      sns.set_style("dark")

      class DictStats():
          def __init__(self, length, size, load, collisions, distribution) -> None:
              self.length = length
              self.size = size
              self.load = load
              self.collisions = collisions
              self.distribution = distribution.strip().split(" ")
```

```
[12]: entries = []
      dist_flag = False
      with open("stats.txt") as f:
          lines = f.readlines()
          for idx, line in enumerate(lines):
              if line.startswith("Hash"):
                  length = int(re.search(r"\d+", line)[0])
                  size = int(lines[idx+1].split(": ")[1])
                  load = float(lines[idx+2].split(": ")[1])
                  collisions = int(lines[idx+3].split(": ")[1])
                  if lines[idx+4].startswith("Distribution"):
                      dist_flag = True
                      distribution = lines[idx+5]
                      entries.append(DictStats(length, size, load, collisions,
↪distribution))
                  else:
                      entries.append(DictStats(length, size, load, collisions))
```



```

[13]: %%capture
df = pd.DataFrame(columns=["length", "size", "load", "collisions", "entries",
    ↳ "collision_rate"], dtype=int)
for entry in entries:
    df_entry = [entry.length, entry.size, entry.load, entry.collisions, entry.
    ↳ size*entry.load, entry.collisions/entry.size*entry.load]
    df = df.append(pd.Series(df_entry, index=df.columns), ignore_index=True)
# Biggest hash table (takes a lot of time to unpack)
if dist_flag:
    dist = pd.DataFrame(columns=["distribution"], dtype=int)
    for i in entries[14].distribution:
        dist = dist.append(pd.Series([int(i)], index=dist.columns),
    ↳ ignore_index=True)

[14]: if dist_flag:
        fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(20, 5))
    else:
        fig, ax1 = plt.subplots(figsize=(20, 5))
    ax1_2 = ax1.twinx()

    sns.regplot(df, x="entries", y="collision_rate", fit_reg=False, ax=ax1_2,
    ↳ color="green", label="Collision rate")
    sns.regplot(df, x="entries", y="collisions", fit_reg=False, ax=ax1, color="red",
    ↳ label="Collisions")
    sns.regplot(df, x="entries", y="load", fit_reg=False, ax=ax1_2, color="blue",
    ↳ label="Load Factor")

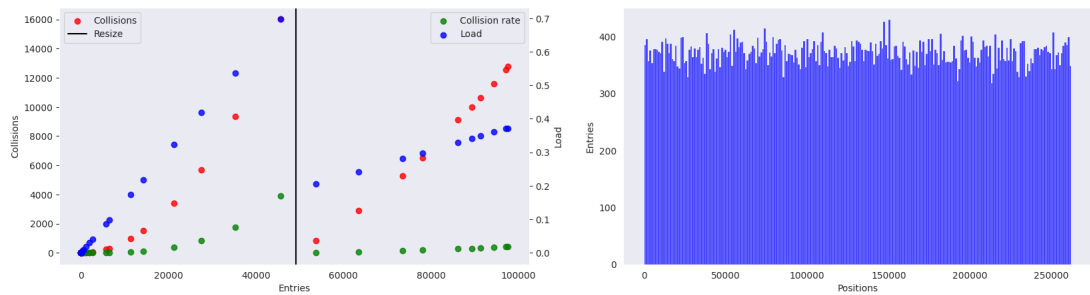
    ax1.set_ylabel("Collisions")
    ax1_2.set_ylabel("Percentage (%)")
    ax1.set_xlabel("Entries")
    ax1.legend(loc="upper left")
    ax1_2.legend(loc="upper right")

    if dist_flag:
        sns.histplot(dist, x="distribution", bins=250, ax=ax2, color="blue",
    ↳ label="Distribution")
        ax2.set_ylabel("Entries per 1000 positions")
        ax2.set_xlabel("Positions")

    # Resize happens at 75% of the hash table size
    ax1.axvline(x=entries[0].size*0.75, color="black", linestyle="-", label="Resize")
    ax1.legend(loc="upper left")

    plt.show()

```



```
[15]: print(f"Average load: {df[df['load'] != 0]['load'].mean()}")
      print(f"Collision per entry: {df['collisions'].sum() / df['entries'].sum()}")
      df.sort_values(by=['entries'], inplace=True, ascending=False)
      print(f"Most common word lengths: {df.head(5)['length'].values[0:5]}")
```

Average load: 0.1937381030366667

Collision per entry: 0.11980706984917514

Most common word lengths: [14. 15. 13. 12. 16.]