

**W celu uruchomienia programu należy posiadać następujące narzędzia:**

- CMake w wersji 3.5 lub nowszej
- kompilator c++ wspierający standard c++ 20

**Aby uruchomić program należy:**

- rozpakować plik Pysczak.zip

**Linux:**

(w terminalu) unzip file.zip -d nazwa\_katalogu

**Windows:**

Kliknij PPM oraz wybierz opcję „wypakuj”, a następnie wskaż docelowy katalog.

**Mac:**

Przenieś archiwum do docelowego katalogu, a następnie kliknij 2 razy LPM.

- wejść przy użyciu konsoli do katalogu z projektem

**Linux/Mac:**

(w terminalu) cd nazwa\_katalogu/AVL\_Tree

**Windows**

(w cmd) cd nazwa\_katalogu\AVL\_Tree

- utworzyć w katalogu z projektem katalog „build”

**Linux/Mac/Windows:**

(w terminalu) mkdir build

- wejść do katalogu „build”, a następnie uruchomić w nim komendę „cmake .. && make -j”

**Linux/Mac/ Windows:**

(w terminalu) cd build && make -j

- uruchomić program podając jako argument ścieżkę do zbudowanego pliku wykonywalnego z testami

**Linux/Mac:**

./avl\_tree tests/bst\_tests

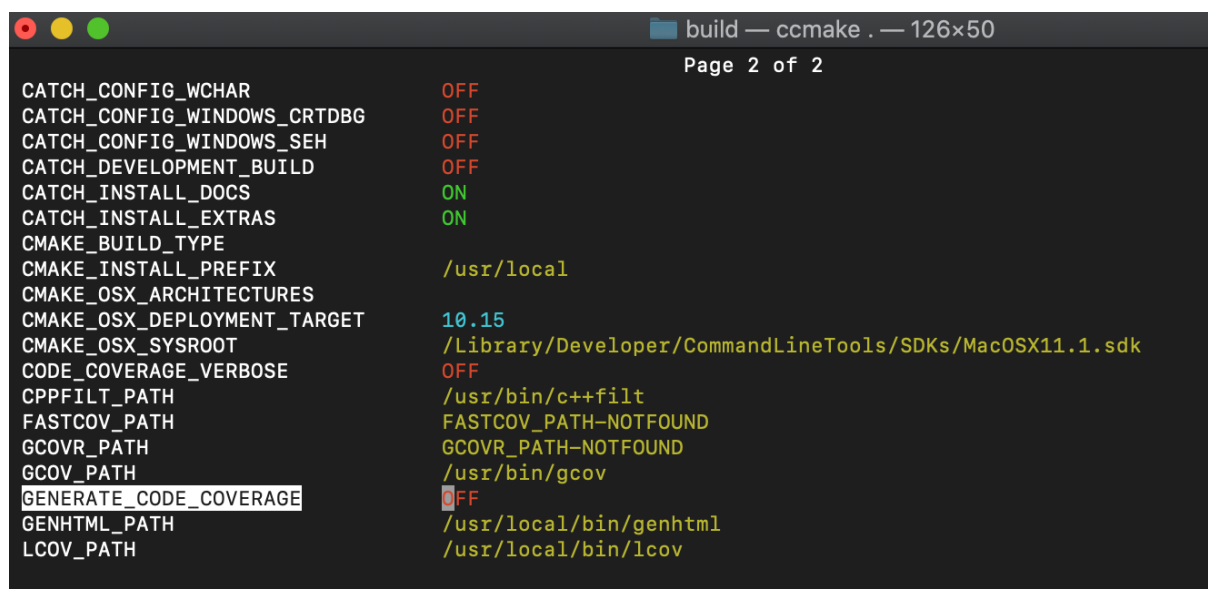
**Windows:**

(w cmd) Wejdź do katalogu z projektem, a następnie w folderze build wywołaj plik .exe z parametrem tests/bst\_tests.

- poruszanie się po menu obsługane jest za pomocą klawiatury numerycznej (klawisze 1, 2, 3). Klawisz 0 odpowiada za cofnięcie się lub powrót do systemu.

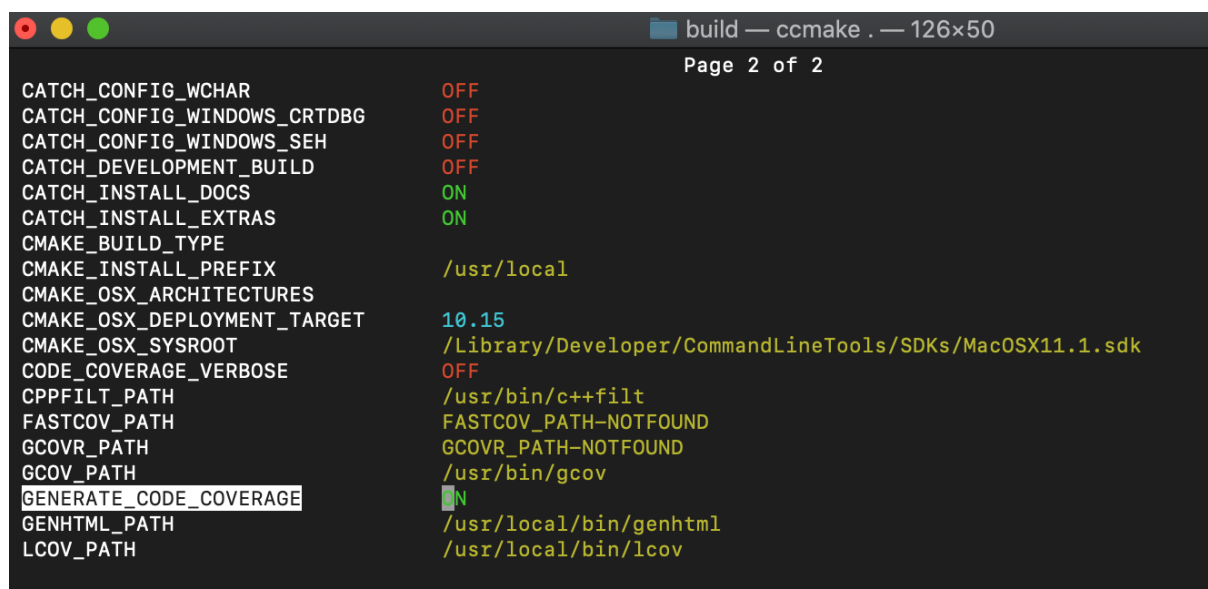
## **OPCJONALNIE:**

Istnieje możliwość wygenerowania **raportu pokrycia testami** algorytmu. W tym celu należy ustawić flagę budowania raportu na „ON” (domyślnie jest wyłączona) np. za pomocą komendy cmake . w katalogu build



```
build — cmake . — 126x50
Page 2 of 2
CATCH_CONFIG_WCHAR      OFF
CATCH_CONFIG_WINDOWS_CRTDBG OFF
CATCH_CONFIG_WINDOWS_SEH OFF
CATCH_DEVELOPMENT_BUILD OFF
CATCH_INSTALL_DOCS      ON
CATCH_INSTALL_EXTRAS    ON
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX    /usr/local
CMAKE_OSX_ARCHITECTURES
CMAKE_OSX_DEPLOYMENT_TARGET 10.15
CMAKE_OSX_SYSROOT       /Library/Developer/CommandLineTools/SDKs/MacOSX11.1.sdk
CODE_COVERAGE_VERBOSE   OFF
CPPFILT_PATH            /usr/bin/c++filt
FASTCOV_PATH            FASTCOV_PATH-NOTFOUND
GCOVR_PATH              GCOVR_PATH-NOTFOUND
GCOV_PATH               /usr/bin/gcov
GENERATE_CODE_COVERAGE  OFF
GENHTML_PATH            /usr/local/bin/genhtml
LCOV_PATH                /usr/local/bin/lcov
```

Rys. 1. Flaga generowania raportu wyłączona.



```
build — cmake . — 126x50
Page 2 of 2
CATCH_CONFIG_WCHAR      OFF
CATCH_CONFIG_WINDOWS_CRTDBG OFF
CATCH_CONFIG_WINDOWS_SEH OFF
CATCH_DEVELOPMENT_BUILD OFF
CATCH_INSTALL_DOCS      ON
CATCH_INSTALL_EXTRAS    ON
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX    /usr/local
CMAKE_OSX_ARCHITECTURES
CMAKE_OSX_DEPLOYMENT_TARGET 10.15
CMAKE_OSX_SYSROOT       /Library/Developer/CommandLineTools/SDKs/MacOSX11.1.sdk
CODE_COVERAGE_VERBOSE   OFF
CPPFILT_PATH            /usr/bin/c++filt
FASTCOV_PATH            FASTCOV_PATH-NOTFOUND
GCOVR_PATH              GCOVR_PATH-NOTFOUND
GCOV_PATH               /usr/bin/gcov
GENERATE_CODE_COVERAGE  ON
GENHTML_PATH            /usr/local/bin/genhtml
LCOV_PATH                /usr/local/bin/lcov
```

Rys. 2. Flaga generowania raportu włączona.

,a następnie wywołać komendę `make coverage` w katalogu `build/tests`. W katalogu `build` zostanie stworzony katalog „coverage”, w którym będzie znajdował się plik `index.html`, który można otworzyć za pomocą przeglądarki.

#### WYMAGANIA:

Aby stworzyć raport pokrycia testami użytkownik musi posiadać `lconv` w przypadku używania `clang` jako kompilatora lub `gconv`.

```
admin@MacBook-Pro-admin build % ./avl_tree tests/bst_tests
[2
tests/bst_tests

Manu glowne

1) Trywialny przyklad
2) Zaawansowany przyklad
3) Uruchomienie scenariuszy testowych
0) Wyjście z menu

Wybierz numer opcji menu (0 - 3) :1

Trywialny przyklad

1) Podaj liczbe elementow do dodania do drzewa
0) Wyjście z menu

Wybierz numer opcji menu (0 - 1) :0

Manu glowne

1) Trywialny przyklad
2) Zaawansowany przyklad
3) Uruchomienie scenariuszy testowych
0) Wyjście z menu

Wybierz numer opcji menu (0 - 3) :2

Zaawansowany przyklad

1) Znajdz liczbe w drzewie
0) Wyjście z menu

Wybierz numer opcji menu (0 - 1) :0

Manu glowne

1) Trywialny przyklad
2) Zaawansowany przyklad
3) Uruchomienie scenariuszy testowych
0) Wyjście z menu

Wybierz numer opcji menu (0 - 3) :0
admin@MacBook-Pro-admin build %
```

Rys.3. Przykładowa egzekucja programu.

## LCOV - code coverage report

Current view: [top level](#) - [Users/admin/cpp/AVL\\_Tree/libs](#) - [bst.hpp](#) ([source](#) / [functions](#))Test: [coverage.info](#)Date: [2022-06-04 07:44:03](#)

	Hit	Total	Coverage
Lines:	45	45	100.0 %
Functions:	6	6	100.0 %

Line data	Source code
1	: #pragma once
2	:
3	: #include <memory>
4	: #include <string>
5	: #include <iostream>
6	: namespace bst
7	: {
8	:     template <typename T>
9	110 :     struct Node
10	:     {
11	110 :         Node(T data, std::unique_ptr<Node<T>> leftChild, std::unique_ptr<Node<T>> rightChild)
12	55 :         : data{data},
13	55 :         leftChild{std::move(leftChild)},
14	55 :         rightChild{std::move(rightChild)}
15	55 :         {
16	110 :         }
17	:
18	:         T data;
19	:         std::unique_ptr<Node<T>> leftChild;
20	:         std::unique_ptr<Node<T>> rightChild;
21	:     };
22	:
23	:     template <typename T>
24	46 :     [[nodiscard]] std::unique_ptr<Node<T>> insertNode(std::unique_ptr<Node<T>> node, T data)
25	:     {
26	46 :         if (node == nullptr)
27	:         {
28	1 :             return std::make_unique<Node<T>>(data, nullptr, nullptr);
29	:         }
30	45 :         Node<T> *current = node.get();
31	45 :         Node<T> *parent = nullptr;
32	209 :         while (true)
33	:         {
34	209 :             parent = current;
35	209 :             if (data < parent->data)
36	:             {
37	94 :                 current = current->leftChild.get();
38	94 :                 if (current == nullptr)
39	:                 {
40	21 :                     parent->leftChild = std::make_unique<Node<T>>(data, nullptr, nullptr);
41	21 :                     return node;
42	:                 }
43	73 :             }
44	188 :             if (data > parent->data)
45	:             {
46	114 :                 current = current->rightChild.get();
47	114 :                 if (current == nullptr)
48	:                 {
49	23 :                     parent->rightChild = std::make_unique<Node<T>>(data, nullptr, nullptr);
50	23 :                     return node;
51	:                 }
52	91 :             }
53	165 :             if (data == parent->data)
54	:             {
55	1 :                 break;
56	:             }
57	:         }
58	1 :         return node;
59	46 :     }
60	:
61	:     template <typename T>
62	8 :     [[nodiscard]] Node<T> *search(Node<T> *node, T data)
63	:     {
64	8 :         auto *current = node;
65	:
66	24 :         while (current != nullptr && current->data != data)
67	:         {
68	18 :             if (current->data > data)
69	:             {
70	7 :                 if (!current->leftChild)
71	:                 {
72	1 :                     return nullptr;
73	:                 }
74	6 :                 current = current->leftChild.get();
75	6 :             }
76	17 :             if (current->data < data)
77	:             {

Rys. 4. Pokrycie testami biblioteki BST.