# কোডকথা (CodeKotha) — পূর্ণ প্রজেক্ট রিপোর্ট

## 1. সারসংক্ষেপ

কোডকথা হলো একটি বাংলা প্রোগ্রামিং ল্যাঙ্গুয়েজের কম্পাইলার যা সম্পূর্ণ বাংলা কীওয়ার্ড, বাংলা সংখ্যা, এবং পরিচিত গাণিতিক/তুলনামূলক অপারেটর দিয়ে প্রোগ্রাম লেখার সুবিধা দেয়। প্রজেক্টটি একটি একক C++ ফাইল `codekotha.cpp`-এ লেখা, যা ইনপুট হিসেবে বাংলা সোর্স-কোড পড়ে টোকেনাইজ, পার্স, ইন্টারপ্রেটিভ এক্সিকিউশন এবং ইচ্ছা করলে C কোড জেনারেশন করতে পারে।

প্রধান লক্ষ্য: বাংলায় প্রোগ্রামিংকে সহজলভ্য করা, শিক্ষার্থীদের জন্য মাতৃভাষায় প্রোগ্রামিং শেখা সহজ করা।

## 2. প্রজেক্টের উদ্দেশ্য ও ক্ষেত্র

- শিক্ষার্থীদের জন্য মাতৃভাষায় (বাংলা) কোডিং অভিজ্ঞতা তৈরি করা

- খুবই সহজ কীওয়ার্ড ও সিনট্যাক্স দিয়ে বেসিক প্রোগ্রামিং কনসেপ্ট শেখানো

- লাইটওয়েট, এক ফাইলে কম্পাইলার — শেখা/ডেমো/এক্সপেরিমেন্টের জন্য যথেষ্ট

## 3. ফিচার তালিকা

- বাংলা কীওয়ার্ড: `পূর্ণসংখ্যা` (int), `দেখাও` (print), `যদি` (if), `নইলে` (else)

- বাংলা সংখ্যা সাপোর্ট: `০-৯` (ইনপুটে বাংলা সংখ্যা লিখলে অভ্যন্তরে ASCII-তে রূপান্তর)

- অপারেটর: `+`, `-`, `\*`, `/`, `>`, `<`, `=`

- একাধিক প্যারামিটার সহ প্রিন্ট: `দেখাও(x, y)`

- কন্ডিশনাল স্টেটমেন্ট: `যদি (...) { ... } নইলে { ... }`

- আউটপুট মোড:

- ডাইরেক্ট এক্সিকিউশন আউটপুট (প্রোগ্রাম চালিয়ে রেজাল্ট `output.txt`)

- C কোড জেনারেশন (লক্ষ্য ফাইলে C কোড লেখা)

## 4. সিস্টেম আর্কিটেকচার (উচ্চ-স্তর)

কম্পাইলারের চারটি প্রধান অংশ:

1) Lexer (টোকেনাইজার):

- সোর্স স্ট্রিং থেকে টোকেন তৈরি করে

- বাংলা সংখ্যা → ASCII ডিজিটে কনভার্ট করে

- কীওয়ার্ড সনাক্ত করে (`পূর্ণসংখ্যা`, `দেখাও`, `যদি`, `নইলে`)

- সংখ্যা, স্ট্রিং, ক্যারেক্টার লিটারাল সাপোর্টের অবকাঠামো আছে

2) Parser (সিনট্যাক্স বিশ্লেষণ):

- টোকেন লিস্ট থেকে AST (Abstract Syntax Tree) তৈরি করে

- সমর্থিত কন্সট্রাক্ট: ভেরিয়েবল ডিক্লেয়ারেশন, প্রিন্ট, কন্ডিশনাল

- এক্সপ্রেশন পার্সিং: যোগ/বিয়োগ/গুণ/ভাগ, তুলনা `>`, `<`

3) Interpreter/Executor:

- AST ট্রাভার্স করে প্রোগ্রাম চালায়

- ভেরিয়েবল স্টোরেজ/লুকআপ পরিচালনা করে

- `দেখাও` কমান্ডে রেজাল্ট `outputLines`-এ সংগ্রহ করে

4) Code Generator (C কোড জেনারেশন):

- AST থেকে সরলীকৃত C কোড জেনারেট করে

- ভেরিয়েবলকে `int` হিসেবে আউটপুট

- `যদি/নইলে` ব্লকের জন্য লেবেল/`goto` ব্যবহার করে ফ্লো নিয়ন্ত্রণ

## 5. টোকেন ও গ্রামার (সরল রূপ)

- টোকেন টাইপ (উদাহরণ):

- কীওয়ার্ড: `KW\_PURNO\_SONGKHA`, `KW\_DEKHAO`, `KW\_JODI`, `KW\_NOILE`

- অপারেটর: `=`, `+`, `-`, `\*`, `/`, `>`, `<`

- ডিলিমিটার: `;`, `,`, `(`, `)`, `{`, `}`

- লিটারাল: `INTEGER\_LITERAL`, `FLOAT\_LITERAL`, `STRING\_LITERAL`, `CHAR\_LITERAL`

- `IDENTIFIER`

- গ্রামার স্কেচ:

- Program → Statement\*

- Statement → VarDecl | PrintStmt | IfStmt

- VarDecl → `পূর্ণসংখ্যা` Identifier `=` Expression `;`

- PrintStmt → `দেখাও` `(` Expression (`,` Expression)\* `)` `;`

- IfStmt → `যদি` `(` Expression `)` Block `নইলে` Block

- Block → `{` Statement\* `}`

- Expression → Term ((`+`|`-`) Term)\*

- Term → Factor ((`\*`|`/`) Factor)\*

- Factor → Number | Identifier | `(` Expression `)` | Unary

## 6. ব্যবহার নির্দেশিকা

- কম্পাইল:

- g++: `g++ -std=c++17 -Wall -Wextra -O2 -o codekotha codekotha.cpp`

- Makefile (উদাহরণ ফাইলে `banglang\_compact` উল্লেখ থাকলেও এই প্রজেক্টে টার্গেট বাইনারি `codekotha` হিসেবে ব্যবহারের পরামর্শ)

- রান (ডিফল্ট):

- `./codekotha` (ইনপুট: `input.txt`, আউটপুট: `output.txt`)

- কাস্টম ফাইল:

- `./codekotha -i my\_program.kotha -o result.txt`

- C কোড জেনারেশন:

- `./codekotha -c -i input.txt -o generated.c`

- সাহায্য:

- `./codekotha -h`

## 7. উদাহরণ

ইনপুট (`input.txt`):

```banglang

পূর্ণসংখ্যা x = ৫;

পূর্ণসংখ্যা y = ৩;

দেখাও(x + y);

```

সম্ভাব্য আউটপুট (`output.txt`):

```

8

```

আরেকটি উদাহরণ (কন্ডিশনাল):

```banglang

পূর্ণসংখ্যা x = ১৫;

যদি (x > ১০) {

দেখাও("x বড় সংখ্যা");

দেখাও(x);

} নইলে {

দেখাও("x ছোট সংখ্যা");

দেখাও(০);

}

```

## 8. ইমপ্লিমেন্টেশন হাইলাইটস

- Unicode হ্যান্ডলিং: ইনপুট ফাইল UTF-8 থেকে ম্যানুয়ালি `std::wstring`-এ রূপান্তর করা হয়, যাতে বাংলা অক্ষর/সংখ্যা সঠিকভাবে প্রসেস হয়।

- বাংলা ডিজিট সাপোর্ট: Lexer বাংলা সংখ্যাকে ASCII ডিজিটে ম্যাপ করে, ফলে `১০` → `10` হিসেবে ব্যবহৃত হয়।

- Executor: `evaluateExpression(...)` ইন্টিজার রিটার্ন করে; ফলে বর্তমান ভার্সনে দশমিক মান ট্রাঙ্কেট/রাউন্ডডাউন হয়ে যেতে পারে।

- C কোড জেনারেশন: সরলীকৃত `printf("%d\n", ...)` সহ `int`-বেসড আউটপুট; কন্ডিশনে লেবেল/`goto` ব্যবহৃত।

## 9. বর্তমান সীমাবদ্ধতা

- কেবল `int` টাইপ ভেরিয়েবল (`পূর্ণসংখ্যা`) সমর্থিত; `float`/`string`/`array` ইত্যাদি এখনো সীমিত/অসম্পূর্ণ

- `input.txt`-এ যদি দশমিক মান দেওয়া হয় (যেমন: `১.২`), তা ইন্টারপ্রেটেশনের সময় পূর্ণসংখ্যায় নেমে আসে; উদাহরণস্বরূপ বর্তমান `input.txt` → আউটপুট `1`

- লুপ কন্সট্রাক্ট (যেমন `পুনরাবৃত্তি`, `যতক্ষণ`) এখনো যুক্ত হয়নি

- এরর মেসেজিং বেসিক; উন্নত ডায়াগনস্টিকস ভবিষ্যৎ পরিকল্পনায়

## 10. রোডম্যাপ (সংক্ষেপ)

- লুপ স্টেটমেন্ট (`পুনরাবৃত্তি`, `যতক্ষণ`)

- দশমিক/ফ্লোট সাপোর্ট (`দশমিক`), স্ট্রিং (`লেখা`), অ্যারে (`তালিকা`)

- ইউজার-ডিফাইন্ড ফাংশন (`কাজ`)

- ফাইল অপারেশন, ইন্টারেক্টিভ REPL

- উন্নত এরর রিপোর্টিং ও ইউনিট টেস্ট কাভারেজ বৃদ্ধি

## 11. বিল্ড ও রান ইনস্ট্রাকশন (macOS/Linux)

```bash

g++ -std=c++17 -Wall -Wextra -O2 -o codekotha codekotha.cpp

./codekotha -i input.txt -o output.txt

```

C কোড জেনারেশন টেস্ট:

```bash

./codekotha -c -i input.txt -o generated.c

gcc generated.c -o test\_program && ./test\_program

```

## 12. টেস্টিং স্ট্র্যাটেজি

- ইউনিট-লেভেল: লেক্সিং/পার্সিংয়ের জন্য ক্ষুদ্র ইনপুট কেস

- ইন্টিগ্রেশন: `input.txt` → `output.txt` পাইপলাইন

- জেনারেটেড C ভ্যালিডেশন: কম্পাইল-রান করে তুলনা

## 13. প্রজেক্ট স্ট্রাকচার (বর্তমান)

```

CodeKotha/

├─ codekotha.cpp # মূল কম্পাইলার সোর্স

├─ compile\_kotha.sh # সহজ রান স্ক্রিপ্ট (ঐচ্ছিক/পুরোনো উদাহরণসমূহ বিদ্যমান)

├─ Makefile\_kotha # বিল্ড টার্গেটের উদাহরণ

├─ input.txt # ডিফল্ট ইনপুট

├─ output.txt # এক্সিকিউশন আউটপুট

├─ README.md # ডকুমেন্টেশন (বিস্তারিত)

└─ REPORT\_CodeKotha.md # এই রিপোর্ট

```

## 14. লাইসেন্স ও কৃতজ্ঞতা

- লাইসেন্স: MIT

- কৃতজ্ঞতা: বাংলাদেশী ডেভেলপার কমিউনিটি, ওপেন-সোর্স কন্ট্রিবিউটর, শিক্ষার্থীবৃন্দ

## 15. উপসংহার

কোডকথা একটি শিক্ষামুখী, লাইটওয়েট বাংলা প্রোগ্রামিং কম্পাইলার যা মাতৃভাষায় প্রোগ্রামিং শেখাকে উৎসাহিত করে। ভবিষ্যতে নতুন কন্সট্রাক্ট, উন্নত টাইপ সিস্টেম ও টুলিং যুক্ত হলে এটি আরো শক্তিশালী শেখার প্ল্যাটফর্ম হয়ে উঠবে।