

# Sistem Pengelolaan Transaksi Multi-Fungsi Menggunakan C++: Solusi Terkomputerisasi untuk Efisiensi Pencatatan Penjualan, Pembelian, dan Sewa

Aditya Listyawan<sup>1</sup>, Fadzel Auraffa Destian Abdylla<sup>2</sup>, Fajar Maulana Sandra Arrasya<sup>3</sup>

<sup>1,2,3</sup>Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Kota Semarang

## Artikel Info

### Kata kunci:

Sistem Transaksi  
Pemrograman C++  
Struktur Data  
Console Application  
Pemrograman Terstruktur

## ABSTRAK

Perkembangan teknologi informasi menuntut adanya transformasi dari sistem manual ke sistem terkomputerisasi dalam berbagai bidang, termasuk pengelolaan transaksi bisnis. Pengolahan data secara manual memiliki berbagai keterbatasan, seperti risiko kesalahan yang tinggi, proses yang memakan waktu, serta kesulitan dalam penyimpanan dan pencarian data. Penelitian ini bertujuan untuk mengembangkan sistem pengelolaan transaksi penjualan, pembelian, dan sewa menggunakan bahasa pemrograman C++ berbasis konsol guna mengatasi permasalahan tersebut. Metode pengembangan sistem ini meliputi analisis kebutuhan, perancangan struktur data menggunakan *struct*, perancangan antarmuka menu berbasis teks, serta implementasi logika pemrograman terstruktur menggunakan fungsi, percabangan, dan perulangan. Sistem ini dirancang dengan pendekatan modular untuk meningkatkan keterbacaan, keterpeliharaan (*maintainability*), dan keterperluasan (*extensibility*). Struktur data disusun menggunakan *struct* untuk merepresentasikan entitas transaksi dengan atribut identitas transaksi, nama barang, jenis transaksi, jumlah, harga satuan, dan total pembayaran, yang kemudian diorganisasi dalam *array* untuk pengelolaan data jamak. Antarmuka menu mengimplementasikan *switch-case* untuk navigasi dan perulangan *do-while* demi persistensi program. Hasil implementasi menunjukkan bahwa sistem mampu mengelola data transaksi secara efektif melalui perhitungan otomatis yang akurat, menyajikan laporan riwayat transaksi yang terstruktur, serta memberikan pengalaman pengguna yang intuitif. Meskipun memiliki keterbatasan, seperti belum tersedianya fitur penyimpanan permanen ke berkas eksternal, sistem ini telah memenuhi kebutuhan dasar pengelolaan transaksi dan dapat menjadi media pembelajaran yang efektif untuk memahami konsep fundamental pemrograman C++.

## Penulis Korespondensi :

Aditya Listyawan,  
Program Studi Teknik Informatika  
Fakultas Ilmu Komputer  
Universitas Dian Nuswantoro, Kota Semarang, 50131  
Email: 111202516203@mhs.dinus.ac.id

## 1. PENDAHULUAN

Era digital yang berkembang pesat saat ini telah membawa perubahan signifikan dalam berbagai aspek kehidupan, termasuk dalam pengelolaan informasi dan transaksi bisnis [1]. Kemajuan teknologi informasi menuntut adanya transformasi dari sistem manual menuju sistem terkomputerisasi yang lebih efisien dan akurat [1], [2]. Hal ini berlaku di berbagai bidang, tidak terkecuali dalam proses pengelolaan transaksi seperti penjualan, pembelian, dan sewa yang merupakan aktivitas fundamental dalam kegiatan bisnis dan operasional organisasi [3]. Salah satu aspek penting dalam operasional bisnis yang memerlukan dukungan sistem terkomputerisasi adalah pengelolaan transaksi penjualan, pembelian, dan sewa, yang merupakan aktivitas fundamental dalam siklus bisnis.

Sistem pengolahan data secara manual yang masih banyak digunakan, khususnya pada usaha kecil dan menengah, memiliki berbagai keterbatasan yang cukup signifikan [4]. Keterbatasan tersebut antara lain meliputi tingginya risiko terjadinya kesalahan dalam pencatatan dan perhitungan (*human error*), memerlukan waktu yang relatif lama dalam proses pengolahan data, kesulitan dalam penyimpanan dokumen secara sistematis, serta hambatan dalam pencarian dan pengambilan kembali informasi ketika dibutuhkan [5]. Kondisi ini dapat berdampak pada menurunnya produktivitas kerja dan kurang optimalnya pengambilan keputusan berbasis data [6]. Oleh karena itu, diperlukan suatu sistem yang mampu mengotomatisasi proses transaksi secara sistematis dan terintegrasi.

Pemrograman komputer hadir sebagai solusi inovatif untuk mengatasi berbagai permasalahan tersebut [7]. Melalui pemrograman, dapat dibangun sebuah sistem yang mampu mengolah data secara cepat, akurat, dan efisien, sehingga dapat meningkatkan kualitas pengelolaan informasi transaksi [8]. Salah satu bahasa pemrograman yang populer dan banyak digunakan dalam pembelajaran fundamental pemrograman adalah C++ [9]. Bahasa pemrograman C++ dipilih karena memiliki beberapa keunggulan, yaitu mendukung paradigma pemrograman terstruktur dan berorientasi objek, memiliki performa eksekusi yang baik, serta menyediakan kontrol tingkat rendah terhadap memori yang memungkinkan optimalisasi program [10], [11]. Selain itu, sintaks dan struktur C++ yang sistematis menjadikannya sebagai bahasa yang efektif untuk melatih logika pemrograman, pemahaman algoritma, serta pengelolaan struktur data.

Dalam konteks pengembangan sistem transaksi, penerapan konsep-konsep dasar seperti struktur data, fungsi modular, percabangan, perulangan, serta validasi input menjadi elemen penting dalam membangun aplikasi yang andal. Struktur data berperan dalam menyimpan dan mengorganisasi informasi transaksi secara efisien, sementara fungsi modular membantu membagi program ke dalam bagian-bagian yang lebih terstruktur dan mudah dipelihara. Penggunaan kontrol alur seperti percabangan dan perulangan memungkinkan sistem menangani berbagai kondisi transaksi secara dinamis. Dengan penerapan konsep-konsep tersebut, sistem yang dibangun tidak hanya berfungsi secara teknis, tetapi juga memiliki struktur yang rapi dan mudah dikembangkan lebih lanjut.

Berdasarkan urgensi tersebut, dalam *project* ini dirancang dan dikembangkan sebuah Program Sistem Penjualan, Pembelian, dan Sewa menggunakan bahasa pemrograman C++. Program ini dirancang dengan tujuan untuk membantu pengguna dalam melakukan pencatatan transaksi secara terstruktur, melakukan perhitungan total transaksi secara otomatis dan akurat, serta menampilkan laporan data transaksi dalam format yang sederhana dan mudah dipahami melalui antarmuka berbasis *console*. Dengan adanya program ini, diharapkan dapat memberikan kemudahan dalam pengelolaan data transaksi sekaligus menjadi media pembelajaran praktis dalam menerapkan konsep-konsep pemrograman C++. Fokus utama pengembangan program ini adalah merancang arsitektur sistem penjualan, pembelian, dan sewa berbasis C++ yang efektif dengan mengimplementasikan konsep struktur data, fungsi, serta kontrol alur. Tujuannya adalah menciptakan aplikasi *console-based* yang *user-friendly* untuk mengelola transaksi secara terstruktur. Melalui proyek ini, pemahaman teoritis diterjemahkan menjadi keterampilan praktis guna mengasah kemampuan berpikir logis dan pemecahan masalah dalam pemrograman komputer. Secara akademis, program ini bermanfaat sebagai sarana pembelajaran mendalam mengenai logika pemrograman C++ pada kasus nyata. Secara praktis, aplikasi ini membantu pelaku usaha kecil mengorganisir data transaksi dengan lebih efisien serta berfungsi sebagai dokumentasi referensi bagi pengembang lain. Lebih jauh lagi, sistem ini diproyeksikan sebagai prototipe yang dapat dikembangkan ke skala yang lebih kompleks, seperti integrasi *database* atau antarmuka grafis di masa depan.

## 2. METODE

Bab ini akan menguraikan landasan teori yang menjadi kerangka acuan dalam penelitian ini. Pembahasan dimulai dari pemahaman mendasar mengenai konsep pemrograman, karakteristik bahasa pemrograman C++, hingga penerapan paradigma pemrograman terstruktur. Selain itu, bab ini juga akan membahas secara mendalam mengenai struktur data, penggunaan *array*, fungsi dan prosedur, serta struktur kontrol alur program yang meliputi perulangan dan percabangan. Pemahaman komprehensif terhadap teori-teori ini sangat krusial sebagai fondasi dalam perancangan dan pengembangan sistem transaksi yang menjadi fokus penelitian.

### 2.1 Pengertian Pemrograman

Pemrograman adalah proses sistematis menyusun instruksi logis menggunakan bahasa tertentu agar komputer dapat menyelesaikan masalah spesifik [12]. Proses ini tidak hanya sebatas penulisan kode, tetapi mencakup tahapan analisis, perancangan algoritma, hingga pengujian dan pemeliharaan program secara terstruktur [13], [14].

Dalam teknologi modern, pemrograman menjadi keterampilan fundamental untuk menciptakan berbagai solusi digital dan mengotomatisasi proses bisnis [15]. Melalui pemrograman, masalah kompleks dapat diurai dan diselesaikan secara efisien menggunakan algoritma yang tepat, sehingga meningkatkan efektivitas operasional dan pengambilan keputusan [16].

## 2.2 Bahasa Pemrograman C++

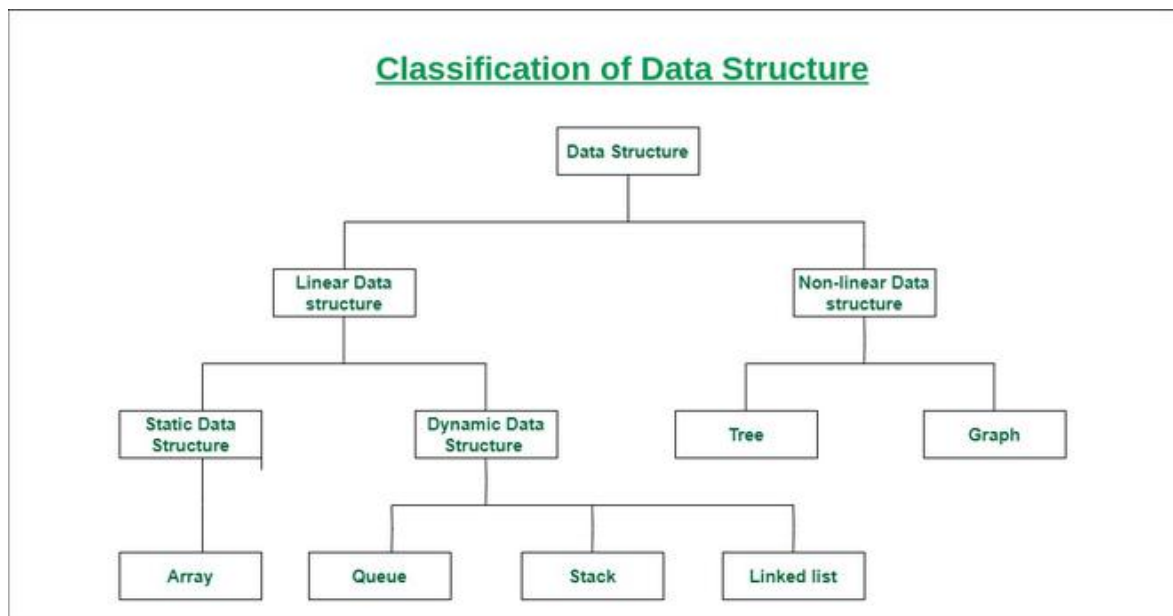
C++ adalah bahasa pemrograman tingkat menengah (*middle-level language*) yang dikembangkan oleh Bjarne Stroustrup pada tahun 1979 sebagai pengembangan dan perluasan dari bahasa pemrograman C [17]. Bahasa C++ dirancang untuk mendukung dua paradigma pemrograman utama, yaitu pemrograman terstruktur (*procedural programming*) dan pemrograman berorientasi objek (*object-oriented programming*) [17]. Hal ini memberikan fleksibilitas tinggi bagi pemrogram dalam mengembangkan aplikasi yang sesuai dengan kebutuhan dan kompleksitas sistem [18]. C++ merupakan bahasa pemrograman berperforma tinggi yang menawarkan fleksibilitas manipulasi memori serta dukungan berbagai paradigma, sehingga lazim digunakan dalam pengembangan sistem operasi hingga *game engine* [19]. Selain keunggulan teknis seperti efisiensi kode mesin dan kekayaan pustaka STL, C++ menjadi fondasi kuat dalam pembelajaran fundamental karena menuntut pemahaman mendalam atas konsep tipe data, struktur kontrol, hingga manajemen memori [20]. Menguasai C++ memberikan dasar yang kokoh untuk mempelajari bahasa modern lain seperti Java atau Python, mengingat banyak konsep dasarnya yang diadopsi dari bahasa ini [21].

## 2.3 Pemrograman Terstruktur

Pemrograman terstruktur adalah metodologi yang menekankan pengorganisasian kode melalui struktur kontrol urutan (*sequence*), percabangan (*selection*), dan perulangan (*iteration*) [22]. Paradigma ini hadir untuk menggantikan penggunaan pernyataan GOTO yang tidak teratur demi menghindari "*spaghetti code*", sehingga menghasilkan alur program yang lebih logis, mudah dipahami, dan lebih sederhana untuk diperbaiki atau dipelihara [23]. Prinsip utamanya meliputi penggunaan desain *top-down*, modularisasi melalui fungsi-fungsi spesifik, serta penerapan *single entry-single exit* [23]. Dalam sistem penjualan, pembelian, dan sewa ini, pemrograman terstruktur digunakan untuk mengelola alur transaksi secara sistematis, mulai dari validasi input hingga keluaran data. Hal ini memungkinkan setiap modul dikembangkan secara independen, sehingga meningkatkan efisiensi pengembangan dan pemeliharaan sistem di masa depan.

## 2.4 Struktur Data

Struktur data adalah metode pengorganisasian data yang krusial untuk mengoptimalkan efisiensi algoritma dalam hal penggunaan memori serta kecepatan eksekusi [24]. Dalam C++, struktur data mencakup tipe primitif hingga bentuk kompleks seperti *array*, *linked list*, dan *tree*, di mana pemilihan yang tepat sangat menentukan kemudahan pengembangan sistem [25]. Pada proyek transaksi ini, fitur *struct* digunakan sebagai instrumen utama untuk mengelola informasi transaksi secara terstruktur dan efisien.



Gambar 1 Tipe-Tipe Struktur Data [26]

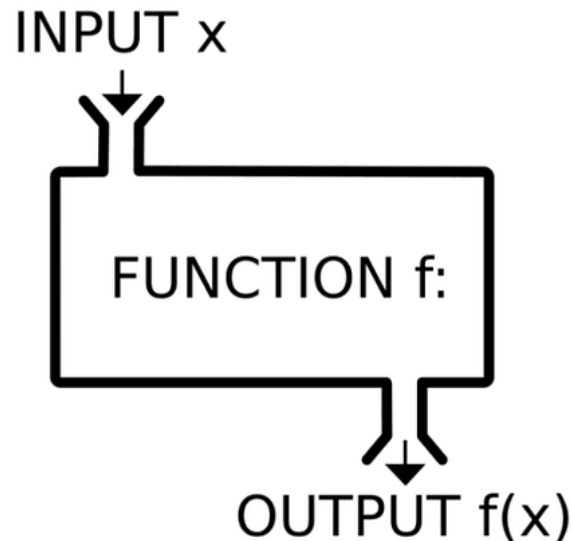
*Struct* dalam C++ memungkinkan penggabungan beberapa variabel dengan tipe data yang berbeda ke dalam satu kesatuan logis yang merepresentasikan suatu entitas [27]. Dalam konteks program ini, *struct* digunakan untuk menyimpan atribut-atribut transaksi seperti nama barang atau jasa, jenis transaksi (penjualan, pembelian, atau sewa), jumlah unit, harga satuan, total pembayaran, tanggal transaksi, dan informasi relevan lainnya. Penggunaan *struct* menjadikan kode lebih terorganisasi, mudah dipahami, dan memudahkan dalam pengiriman data antar-fungsi, karena seluruh informasi terkait satu transaksi terkemas dalam satu struktur data yang kohesif.

## 2.5 Array

*Array* adalah struktur data fundamental yang menyimpan sekumpulan elemen bertipe homogen dalam lokasi memori yang berdekatan, memungkinkan akses cepat melalui indeks numerik [28]. Dengan karakteristik ukuran tetap (*fixed size*) dan alokasi memori yang efisien, *array* mendukung akses data acak secara instan dengan kompleksitas waktu  $O(1)$  [29]. Hal ini memudahkan implementasi berbagai algoritma penting seperti pengurutan, pencarian, dan penelusuran data secara sistematis [29], [30]. Dalam pengembangan sistem transaksi ini, *array of struct* digunakan untuk menyimpan banyak *record* data secara sekaligus guna keperluan pelaporan dan kalkulasi statistik. Meski memudahkan proses iterasi untuk menampilkan daftar transaksi, keterbatasan ukuran tetap pada *array* menuntut penentuan kapasitas maksimum sejak awal. Sebagai solusi alternatif untuk fleksibilitas yang lebih tinggi, penggunaan struktur data dinamis seperti `std::vector` dari pustaka STL dapat dipertimbangkan jika jumlah transaksi bersifat fluktuatif.

## 2.6 Fungsi dan Prosedur

Fungsi dan prosedur merupakan blok program atau subprogram yang dirancang untuk melaksanakan tugas spesifik dan dapat dipanggil berulang kali dari bagian lain dalam program [31]. Konsep ini merupakan implementasi dari prinsip modularisasi dalam pemrograman, di mana program yang kompleks dipecah menjadi bagian-bagian kecil yang lebih sederhana, terkelola, dan independen [32]. Dalam bahasa C++, istilah "*function*" digunakan untuk mencakup baik fungsi yang mengembalikan nilai maupun yang tidak (*void function* atau prosedur) [32].



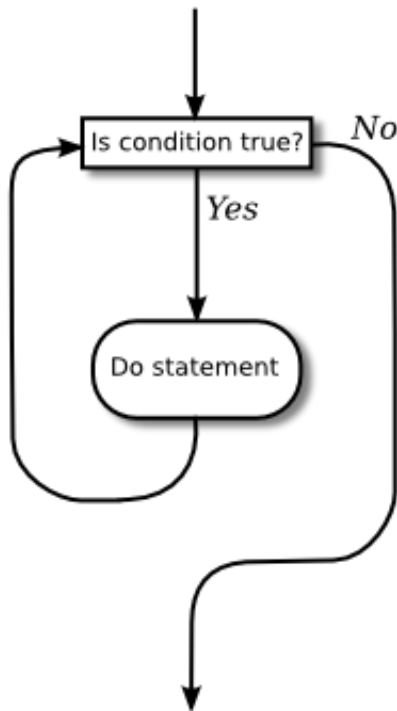
Gambar 2 Visualisasi Fungsi pada Pemrograman [33]

Dalam program sistem transaksi ini, modularisasi dilakukan dengan memisahkan logika ke dalam subprogram yang spesifik, seperti fungsi kalkulasi harga dan prosedur penampilan menu atau laporan. Pendekatan ini memudahkan proses *debugging* dan pengujian karena setiap komponen dapat diuji secara independen. Dengan meminimalkan ketergantungan antar-blok kode, pengembangan sistem menjadi lebih terorganisir, efisien, dan mudah untuk dikembangkan di masa mendatang.

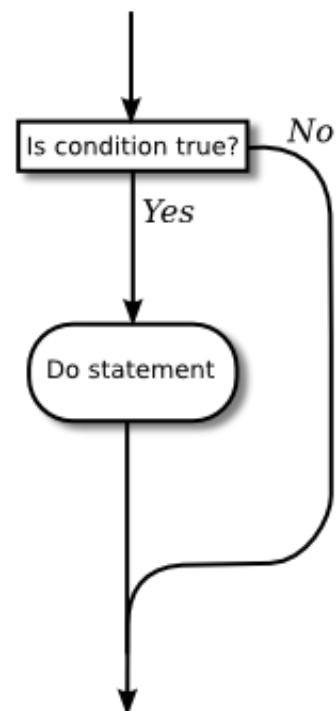
## 2.7 Perulangan dan Percabangan

Perulangan (*looping*) dan percabangan (*branching*) adalah struktur kontrol fundamental yang mengatur alur eksekusi program berdasarkan kondisi tertentu [34]. Khusus untuk perulangan, bahasa C++ menyediakan tiga jenis utama: *for* untuk iterasi dengan jumlah pasti seperti penelusuran *array*, *while* untuk iterasi berbasis kondisi yang fleksibel, dan *do-while* yang menjamin eksekusi minimal satu kali sehingga sangat ideal untuk implementasi program berbasis menu [35]. Secara keseluruhan, struktur kontrol ini memungkinkan program untuk membuat keputusan logis, melakukan validasi *input*, serta memproses data koleksi secara efisien dan otomatis [36].

**While Loop Flow of Control**



**If Statement Flow of Control**



Gambar 3 Diagram Visualisasi Perulangan (Kiri) dan Diagram Percabangan (Kanan) pada Pemrograman [37]

Percabangan dan perulangan merupakan komponen krusial yang memberikan fleksibilitas serta logika interaktif pada sistem transaksi ini. Struktur percabangan, seperti *if-else* dan *switch-case*, digunakan untuk memvalidasi *input* serta mengarahkan alur program berdasarkan pilihan menu pengguna (penjualan, pembelian, atau sewa). Sementara itu, struktur perulangan memastikan program tetap berjalan secara kontinu hingga pengguna memilih untuk keluar, sekaligus memungkinkan pemrosesan data koleksi dari *array* secara berulang. Sinergi keduanya menciptakan aplikasi yang responsif dan mampu menangani berbagai skenario transaksi secara otomatis dan efisien.

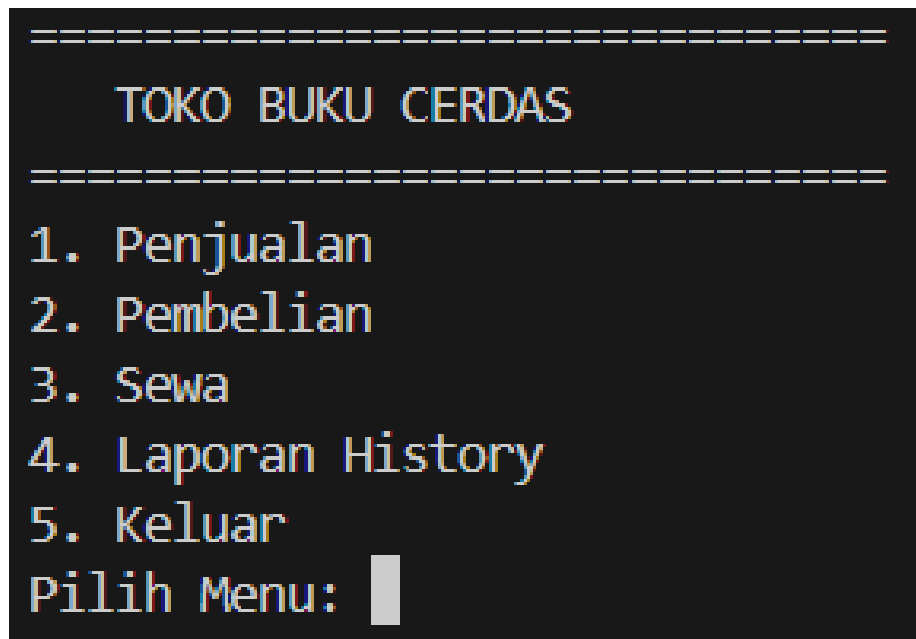
## 3. PEMBAHASAN HASIL

BAB 3 ini membahas hasil perancangan dan analisis sistem yang telah dikembangkan berdasarkan tujuan penelitian. Pembahasan difokuskan pada bagaimana rancangan sistem diterjemahkan ke dalam bentuk alur kerja, kebutuhan sistem, struktur data, serta mekanisme interaksi antara pengguna dan program. Setiap komponen sistem dianalisis untuk menunjukkan keterkaitannya dalam mendukung proses pengelolaan transaksi penjualan, pembelian, dan sewa secara terstruktur.

### 3.1 Gambaran Umum Sistem

Sistem yang dirancang dan dikembangkan dalam penelitian ini merupakan aplikasi berbasis *console* (*command-line interface*) yang dibangun menggunakan bahasa pemrograman C++. Sistem ini bertujuan untuk menyediakan solusi sederhana namun fungsional dalam pengelolaan data transaksi yang mencakup tiga jenis aktivitas utama, yaitu transaksi penjualan, pembelian, dan sewa. Melalui antarmuka berbasis teks yang intuitif, pengguna dapat dengan mudah memilih jenis transaksi yang diinginkan melalui menu utama, memasukkan

data-data yang diperlukan sesuai dengan jenis transaksi yang dipilih, dan sistem akan secara otomatis memproses data tersebut serta menampilkan hasil transaksi secara *real-time*.



Gambar 4 Antarmuka Program

Filosofi desain sistem ini mengutamakan kesederhanaan dan kemudahan penggunaan, baik sebagai media pembelajaran pemrograman C++ bagi pemula maupun sebagai alat pengelolaan transaksi yang praktis. Alur kerjanya dirancang secara sistematis dengan pendekatan terstruktur guna meminimalkan kesalahan pengolahan data, mempercepat proses *debugging*, serta menjamin kemudahan pemeliharaan sistem di masa depan.

Arsitektur sistem ini menerapkan pendekatan modular yang memisahkan setiap fungsionalitas utama ke dalam unit-unit independen namun tetap terintegrasi. Dengan mengimplementasikan konsep fundamental seperti struktur data, *array*, fungsi, serta kontrol alur, sistem ini berfungsi sebagai studi kasus komprehensif yang meningkatkan keterbacaan kode dan memungkinkan pengujian komponen secara terpisah bagi pengembangan tingkat dasar hingga menengah.

### 3.2 Perancangan Struktur Data

Perancangan struktur data merupakan aspek krusial dalam pengembangan sistem karena menentukan tata cara penyimpanan, pengorganisasian, dan manipulasi informasi di dalam memori selama program berjalan. Struktur data yang dirancang dengan baik akan menghasilkan program yang efisien, mudah dipelihara, serta memiliki skalabilitas (*scalable*) untuk pengembangan fitur di masa mendatang. Dalam sistem ini, struktur data berbasis *struct* (*user-defined type*) dirancang dan diimplementasikan untuk merepresentasikan entitas transaksi. Pemilihan *struct* sebagai mekanisme penyimpanan data didasarkan pada beberapa pertimbangan utama: (1) kemampuan *struct* untuk mengelompokkan beberapa atribut dengan tipe data berbeda ke dalam satu kesatuan logis; (2) kemudahan dalam pengiriman data antar-fungsi, baik melalui mekanisme *pass-by-value* maupun *pass-by-reference*; (3) peningkatan keterbacaan kode karena relasi antar-atribut data menjadi lebih eksplisit; serta (4) kesesuaian dengan paradigma pemrograman terstruktur yang diterapkan dalam penelitian ini. Struktur data transaksi yang dirancang terdiri atas beberapa atribut atau *field* yang merepresentasikan karakteristik penting dari sebuah transaksi. Atribut-atribut tersebut meliputi: (1) ID transaksi, yang berfungsi sebagai pengidentifikasi unik untuk setiap transaksi guna memudahkan proses pengacuan (*referencing*) dan pencarian (*searching*); (2) nama barang atau jasa, yang menyimpan deskripsi *item* objek transaksi; (3) jenis transaksi, yang mengkategorikan transaksi sebagai penjualan, pembelian, atau sewa menggunakan tipe data yang sesuai seperti *string* atau *enumeration*; (4) jumlah barang atau unit, yang merepresentasikan kuantitas *item* dalam transaksi; (5) harga satuan, yang menyimpan nilai harga per unit; dan (6) total pembayaran, yang merupakan hasil kalkulasi perkalian antara jumlah barang dan harga satuan.

Tabel 1 Implementasi Struktur Data

|                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| main.cpp                                                                                                                                                                                                                                                                       |
| <pre>// ===== (ADT) ===== struct Transaksi {     string nama;     string jenis; // Penjualan / Sewa     int jumlah;     int harga;     int total; };  // ===== KONSTANTA ===== const int MAX = 100;  // ===== ARRAY ===== Transaksi history[MAX]; int jumlahHistory = 0;</pre> |

Struktur data ini dirancang agar bersifat mandiri (*self-contained*), artinya setiap instans *struct* memuat seluruh informasi yang diperlukan untuk merepresentasikan satu transaksi secara utuh. Untuk mengelola banyak transaksi, struktur data ini kemudian diorganisasi dalam bentuk *array*. Hal ini memungkinkan penyimpanan sejumlah rekaman (*record*) transaksi secara sekuensial yang dapat diakses menggunakan indeks. Pendekatan ini memfasilitasi berbagai operasi, seperti iterasi untuk menampilkan seluruh transaksi, pencarian untuk menemukan transaksi tertentu, serta agregasi untuk menghitung statistik seperti total penjualan atau rata-rata nilai transaksi. Desain struktur data ini juga mempertimbangkan aspek ekstensibilitas, di mana *field* tambahan dapat disisipkan di masa mendatang tanpa mengubah struktur fundamentalnya. Contoh pengembangan tersebut meliputi penambahan *field* untuk tanggal transaksi, nama pelanggan, metode pembayaran, atau status transaksi, sehingga sistem dapat berkembang seiring dengan evolusi kebutuhan pengguna.

### 3.3 Perancangan Antarmuka Menu Program

Perancangan antarmuka pengguna (*user interface*) merupakan aspek penting yang memengaruhi tingkat ketergunaan (*usability*) dan pengalaman pengguna (*user experience*) dalam berinteraksi dengan sistem. Meskipun sistem ini berbasis konsol (*console*) dengan keterbatasan dalam aspek visual, perancangan menu yang baik tetap dapat menghasilkan antarmuka yang intuitif, efisien, dan ramah pengguna (*user-friendly*). Menu program dirancang menggunakan pendekatan menu berbasis teks (*text-based menu*) atau antarmuka pengguna karakter (*Character User Interface* - CUI), di mana seluruh interaksi dilakukan melalui teks tanpa elemen grafis. Menu utama berfungsi sebagai pusat navigasi yang menyediakan akses ke seluruh fungsionalitas sistem melalui pilihan-pilihan yang terstruktur dan bernomor untuk memudahkan seleksi. Struktur menu dirancang dengan hierarki datar (*flat*) atau satu tingkat untuk menjaga kesederhanaan, di mana setiap pilihan menu mengarah langsung ke fungsionalitas tertentu tanpa sub-menu yang bertingkat.

Pilihan-pilihan yang tersedia dalam menu utama dirancang secara komprehensif untuk mencakup seluruh kebutuhan fungsional sistem. Pilihan pertama adalah "Input Transaksi Penjualan", yang mengarahkan pengguna ke formulir masukan untuk mencatat transaksi penjualan dengan meminta data relevan seperti nama barang, jumlah, dan harga. Pilihan kedua adalah "Input Transaksi Pembelian", yang memiliki mekanisme serupa namun dalam konteks pencatatan barang atau jasa yang dibeli. Pilihan ketiga adalah "Input Transaksi Sewa", yang dapat mencakup parameter tambahan terkait durasi atau periode sewa sesuai dengan logika bisnis yang diterapkan. Pilihan keempat, "Tampilkan Laporan Transaksi", menyediakan akses untuk melihat riwayat atau daftar transaksi yang telah dilakukan dengan menampilkan informasi dalam format tabel atau daftar yang terorganisasi dengan baik. Pilihan kelima adalah "Keluar dari Program", yang memberikan opsi kepada pengguna untuk mengakhiri sesi dan menutup aplikasi secara teratur (*graceful exit*), yang idealnya dilengkapi dengan konfirmasi untuk menghindari penutupan program yang tidak disengaja. Implementasi menu dirancang menggunakan kombinasi struktur kontrol percabangan (*branching*) dan perulangan (*looping*) untuk menciptakan alur program yang interaktif dan persisten. Percabangan, khususnya menggunakan pernyataan *switch-case*, digunakan untuk menangani pemilihan menu berdasarkan masukan numerik dari pengguna, di mana setiap *case* mengarah ke fungsi atau blok kode yang menangani fungsionalitas spesifik. Struktur *switch-*

*case* dipilih karena lebih mudah dibaca dan efisien dibandingkan *multiple if-else* ketika menangani banyak pilihan diskrit.

Tabel 2 Implementasi Antarmuka Program

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> main.cpp // ===== PROSEDUR UI ===== void header() {     cout &lt;&lt; "=====\\n";     cout &lt;&lt; "  TOKO BUKU CERDAS  \\n";     cout &lt;&lt; "=====\\n"; }  void footer() {     cout &lt;&lt; "=====\\n";     cout &lt;&lt; "Alamat: Jl. Belajar No. 1000\\n";     cout &lt;&lt; "=====\\n"; } // Bagian tampilan menu dalam main() header(); cout &lt;&lt; "1. Penjualan" &lt;&lt; endl; cout &lt;&lt; "2. Pembelian" &lt;&lt; endl; cout &lt;&lt; "3. Sewa" &lt;&lt; endl; cout &lt;&lt; "4. Laporan History" &lt;&lt; endl; cout &lt;&lt; "5. Keluar" &lt;&lt; endl; cout &lt;&lt; "Pilih Menu: "; cin &gt;&gt; pilihan; </pre> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Perulangan, menggunakan *do-while loop* atau *while loop*, membungkus seluruh logika menu untuk memastikan program terus berjalan dan menampilkan menu kembali setelah setiap operasi selesai, sampai pengguna secara eksplisit memilih opsi untuk keluar. Pendekatan ini menciptakan menu persisten yang memungkinkan pengguna untuk melakukan banyak transaksi atau operasi dalam satu sesi tanpa perlu menjalankan ulang program berkali-kali. Kondisi keluar dari *loop* biasanya dikontrol melalui variabel penanda (*flag variable*) atau dengan memeriksa pilihan menu tertentu. Selain struktur navigasi, perancangan menu juga memperhatikan aspek panduan pengguna dan penanganan kesalahan (*error handling*). Setiap menu dilengkapi dengan instruksi yang jelas mengenai cara memberikan masukan, validasi untuk memastikan masukan yang diterima valid (misalnya memastikan pilihan menu berada dalam rentang yang benar), serta pesan kesalahan yang informatif ketika terjadi masukan yang tidak valid guna memberikan umpan balik konstruktif kepada pengguna untuk melakukan koreksi.

### 3.4 Alur Kerja dan Logika Sistem

Alur kerja sistem merepresentasikan rangkaian proses dan titik pengambilan keputusan sejak program dijalankan hingga program berakhir. Pemahaman terhadap alur ini penting untuk memastikan seluruh skenario penggunaan dapat ditangani secara tepat dan efisien. Proses diawali dengan inisialisasi program yang meliputi deklarasi dan inisialisasi variabel, seperti *array* untuk menyimpan data transaksi, penghitung jumlah transaksi, serta variabel kontrol lainnya. Setelah tahap ini, program memasuki perulangan utama (*main loop*) yang menampilkan menu utama kepada pengguna. Pada tahap penampilan menu, sistem menyajikan pilihan transaksi dalam format yang terstruktur dan menunggu masukan dari pengguna. Setiap masukan divalidasi untuk memastikan nilai yang dimasukkan berada dalam rentang yang sesuai. Jika pengguna memilih transaksi penjualan, pembelian, atau sewa, sistem akan meminta masukan data transaksi seperti nama barang, jumlah, dan harga satuan. Data yang dimasukkan kemudian diproses melalui perhitungan otomatis untuk menentukan total transaksi, dengan kemungkinan penyesuaian tambahan pada transaksi sewa. Seluruh data transaksi yang telah diproses disimpan ke dalam *array* dan dikonfirmasi kepada pengguna dalam bentuk ringkasan transaksi.

Tabel 3 Implementasi Alur Kerja dan Logika Sistem

|                                                                                                |
|------------------------------------------------------------------------------------------------|
| <pre> main.cpp // ===== FUNGSI KALKULASI ===== int hitungTotal(int harga, int jumlah) { </pre> |
|------------------------------------------------------------------------------------------------|

```

    return harga * jumlah;
}
// ===== MENU PENJUALAN =====
void menuPenjualan() {
    Transaksi t;
    t.jenis = "Penjualan";

    cout << "Nama Buku   : "; cin >> t.nama;
    cout << "Jumlah       : "; cin >> t.jumlah;
    cout << "Harga Satuan: "; cin >> t.harga;

    t.total = hitungTotal(t.harga, t.jumlah); // Pemanggilan fungsi hitung

    history[jumlahHistory] = t; // Penyimpanan ke array
    jumlahHistory++;          // Increment counter

    cout << "Total Bayar : " << t.total << endl;
}

```

Program akan kembali ke menu utama setelah transaksi selesai, sehingga memungkinkan pengguna melakukan transaksi lain atau memilih opsi untuk menampilkan laporan transaksi. Pada menu laporan, sistem menampilkan seluruh data transaksi yang tersimpan secara terorganisasi, atau menampilkan pesan informatif jika belum terdapat data. Proses ini berlangsung secara berulang hingga pengguna memilih opsi keluar yang kemudian mengakhiri eksekusi program.

#### 4. KESIMPULAN

Berdasarkan seluruh tahapan penelitian yang meliputi perancangan, implementasi, hingga pengujian sistem, dapat disimpulkan bahwa sistem pengelolaan transaksi penjualan, pembelian, dan sewa berbasis C++ telah berhasil dikembangkan sesuai dengan spesifikasi kebutuhan yang ditetapkan. Secara fungsional, sistem ini mampu melakukan manajemen data transaksi yang mencakup proses *input*, kalkulasi biaya, hingga pelaporan riwayat transaksi melalui antarmuka berbasis konsol (*console*). Dari sisi teknis, penelitian ini membuktikan bahwa penerapan konsep fundamental pemrograman C++ meliputi penggunaan struktur data (*struct*), *array*, modularisasi fungsi, serta logika percabangan dan perulangan dapat menghasilkan aplikasi yang terstruktur dan efisien. Keberadaan sistem ini tidak hanya memberikan solusi komputasi yang mempercepat proses perhitungan transaksi, tetapi juga berfungsi sebagai model referensi yang efektif bagi pembelajaran logika pemrograman dasar dan manajemen data sederhana.

Mengingat adanya keterbatasan pada sistem yang dikembangkan saat ini, terdapat beberapa aspek yang direkomendasikan untuk pengembangan lanjutan guna meningkatkan nilai guna dan keandalan sistem. Prioritas utama pengembangan selanjutnya adalah implementasi mekanisme persistensi data melalui integrasi basis data (*database*) atau penyimpanan berkas (*file handling*) agar integritas data tetap terjaga setelah program dihentikan. Selain itu, transformasi antarmuka pengguna dari berbasis teks menjadi antarmuka grafis (*Graphical User Interface/GUI*) sangat disarankan untuk meningkatkan aspek interaktivitas dan pengalaman pengguna. Pengembangan fitur keamanan melalui mekanisme otentikasi atau *login* juga diperlukan untuk membatasi akses hak pengguna. Terakhir, arsitektur sistem perlu ditingkatkan agar mampu menangani skala data yang lebih besar, sehingga sistem dapat diadopsi untuk kebutuhan operasional yang lebih kompleks di masa mendatang.

#### REFERENCES

- [1] Nikmah Dalimunthe *et al.*, "PENGARUH TEKNOLOGI DIGITAL TERHADAP HUKUM BISNIS: TANTANGAN DAN PELUANG," *JSE J. Sharia Econ.*, vol. 4, no. 3, pp. 148–157, Jul. 2025, doi: 10.46773/jse.v4i3.2147.
- [2] R. D. N. Afanda, Sutrisno, E. Tambunan, and M. S. Anwar, "Peningkatan Kompetensi Sumber Daya Manusia dalam Menghadapi Perkembangan Teknologi Digital untuk Pengembangan Bisnis Mikro (UMKM) pada Masyarakat Pondok Petir," *Karimah Tauhid*, vol. 4, no. 2, pp. 1167–1175, Feb. 2025, doi: 10.30997/karimahtauhid.v4i2.18064.
- [3] M. M. Sulaeman, "Peran Strategi Integratif Optimalisasi Sistem Informasi Bisnis berbasis Teknologi Digital untuk Meningkatkan Produktivitas Karyawan melalui Pendekatan Manajemen Personalia Berkelanjutan," *J. Minfo Polgan*, vol. 14, no. 1, pp. 553–562, May 2025, doi: 10.33395/jmp.v14i1.14783.

- [4] C. Putri, I. Ayuni, and V. F. Sanjaya, "Inovasi dan Kreativitas Bisnis dalam Meningkatkan Daya Saing melalui Penerapan Teknologi Digital Studi Kasus Bakso Favorite Sukarama," *J. Kaji. Ilmu Dan Teknol. JKIT*, vol. 2, no. 1, pp. 21–32, Sep. 2025, doi: 10.71200/jkit.v2i1.9.
- [5] Suli Da'im, Moh Ali, Dina Novita, Moh Ali Fais, and Irwansyah, "ANALISIS EKONOMI KREATIF DAN PENGGUNAAN TEKNOLOGI DIGITAL STUDI TENTANG TRANSFORMASI DAN PELUANG PERTUMBUHAN BISNIS," *J. LENTERA BISNIS*, vol. 14, no. 1, pp. 1–14, Jan. 2025, doi: 10.34127/jrlab.v14i1.1258.
- [6] Timoty Agustian Berutu, Dina Lorena Rea Sigalingging, Gaby Kasih Valentine Simanjuntak, and Friska Siburian, "Pengaruh Teknologi Digital terhadap Perkembangan Bisnis Modern," *Neptunus J. Ilmu Komput. Dan Teknol. Inf.*, vol. 2, no. 3, pp. 358–370, Jul. 2024, doi: 10.61132/neptunus.v2i3.258.
- [7] H. Sutanto, J. Jumaedi, E. Nurhaliza, and A. Mardiah, "Pengaruh Adopsi Teknologi Digital dan Strategi Pemasaran Online Terhadap Kinerja Bisnis dalam Kewirausahaan di Indonesia," *Sanskara Ekon. Dan Kewirausahaan*, vol. 3, no. 01, pp. 53–66, Oct. 2024, doi: 10.58812/sek.v3i01.467.
- [8] Larasati Pingkan Cahya Hernita and Agussalim, "Tinjauan Literatur Peran Teknologi Digital Dalam Bisnis: Dampak Disruptif TI Pada Perusahaan," *J. Manaj. Kreat. Dan Inov.*, vol. 2, no. 2, pp. 157–164, Apr. 2024, doi: 10.59581/jmki-widyakarya.v2i2.2997.
- [9] I. Trianiza, E. N. Khirdany, E. Wahyudi, A. Y. Vandika, and S. Sofyan, "Pengenalan Pemrograman Dasar Dunia Koding dengan C++," *YPAD Penerbit*, Jul. 2025, Accessed: Feb. 04, 2026. [Online]. Available: <https://journal.yayasanpad.org/index.php/ypadbook/article/view/434>
- [10] A. A. Aldiansyah, D. S. Nugroho, I. U. W. Mulyono, and E. H. Rachmawanto, "Pembuatan Program Stok Toko Alat Musik Dengan C++," *J. Apl. Teknol. Dan Komputasi*, vol. 1, no. 3, pp. 41–48, Dec. 2025.
- [11] J. J. M. T. S. ST, *Algoritma Dan Pemrograman Dasar Dengan C++: Panduan Terstruktur Dari Konsep Hingga Implementasi*. Penerbit Adab.
- [12] S. H. M. T. S. Kom and I. W. J. M. Cs S. Kom, *Konsep Algoritme dan Aplikasinya dalam Bahasa Pemrograman C++*. Penerbit Andi, 2020.
- [13] A. Zubaidi, A. H. Jatmika, W. Wedashwara, and A. Z. Mardiyansyah, "Pengenalan Algoritma Pemrograman Menggunakan Aplikasi Scratch Bagi Siswa SD 13 Mataram:," *J. Begawe Teknol. Inf. JBegaTI*, vol. 2, no. 1, Mar. 2021, doi: 10.29303/jbegati.v2i1.423.
- [14] F. Aulia and Yahfizham, "Mengenal Bahasa Pemrograman Pada Algoritma Pemrograman," *J. Inform. Busines*, vol. 1, no. 4, pp. 223–228, Jan. 2024, doi: 10.47233/jibs.v1i1.521.
- [15] R. R. Septianisa and T. Anggoro, "Implementasi Bisnis Digital dengan Perancangan Website E-Commerce untuk Usaha Kuliner Seblak Waja (Studi Kasus : Warung Seblak Waja)," *Innov. J. Soc. Sci. Res.*, vol. 4, no. 4, pp. 8406–8417, Aug. 2024, doi: 10.31004/innovative.v4i4.14132.
- [16] A. Nugrahanoro, L. B. Lestari, I. Masfuha, A. A. Pohan, and G. M. Prabowo, "PENDAMPINGAN PEMBUATAN APLIKASI BISNIS UNTUK PEMUDA BOYOLALI MELALUI PEMROGRAMAN DASAR," *SUBSERVE Community Serv. Empower. J.*, vol. 3, no. 1, pp. 111–118, Jan. 2025, doi: 10.36728/scsej.v3i1.77.
- [17] B. Stroustrup, *A Tour of C++*. Addison-Wesley Professional, 2022.
- [18] V. V. Ponggawa, U. B. Santoso, G. A. Talib, M. A. Lamia, A. R. A. Manuputty, and M. F. Yusuf, "Comparative Study of C++ and C# Programming Languages," *J. Syntax Admiration*, vol. 5, no. 12, pp. 5743–5748, Dec. 2024, doi: 10.46799/jsa.v5i12.1926.
- [19] J. Horton, *Beginning C++ Game Programming: Learn C++ from scratch by building fun games*. Packt Publishing Ltd, 2024.
- [20] M. M. Ariffin, N. M. Aszemi, and M. S. Mazlan, "CodeToProtect©: C++ Programming Language Video Game for Teaching Higher Education Learners," *J. Phys. Conf. Ser.*, vol. 1874, no. 1, p. 012064, May 2021, doi: 10.1088/1742-6596/1874/1/012064.
- [21] A. F. Rosmani and M. S. H. S. Nizam, "C++ Rush: An Educational Gaming Experience," *J. Intelek*, vol. 19, no. 1, pp. 79–90, 2024.
- [22] I. K. S. Buana, H. Setiawan, and P. A. W. Putro, *Pemrograman Terstruktur*. Syiah Kuala University Press, 2022.
- [23] A. Sakti, A. A. N. Risal, and V. Alfiani, "Pelatihan Peningkatan Literasi Pemrograman Dasar bagi Siswa SMA melalui Pemrograman Terstruktur Berbasis Java," *POTENSI J. Pengabd. Kpd. Masy.*, vol. 2, no. 2, pp. 86–93, Jun. 2025, doi: 10.61689/potensi.v2i2.36.
- [24] O. Michel, R. Bifulco, G. Rétvári, and S. Schmid, "The Programmable Data Plane: Abstractions, Architectures, Algorithms, and Applications," *ACM Comput Surv*, vol. 54, no. 4, p. 82:1–82:36, May 2021, doi: 10.1145/3447868.
- [25] D. E. Knuth, *The Art of Computer Programming, Volume 4B: Combinatorial Algorithms*. Addison-Wesley Professional, 2022.
- [26] "Data Structure Types, Classifications and Applications," GeeksforGeeks. Accessed: Feb. 04, 2026. [Online]. Available: <https://www.geeksforgeeks.org/dsa/what-is-data-structure-types-classifications-and-applications/>
- [27] B. C. Pierce, *Advanced Topics in Types and Programming Languages*. MIT Press, 2024.
- [28] S. N. Mohanty and P. K. Tripathy, *Data Structure and Algorithms Using C++: A Practical Implementation*. John Wiley & Sons, 2021.
- [29] P. Fatima, "Optimizing Algorithm Efficiency through Advanced Data Structures in C++: A Comparative Analysis of Performance, Scalability, and Complexity," *Int. J. Comput. Inf. Manuf. IJCIM*, vol. 3, no. 2, pp. 66–72, Dec. 2023, doi: 10.54489/ijcim.v3i2.256.

- [30] J. Farrier, *Data Structures and Algorithms with the C++ STL: A guide for modern C++ practitioners*. Packt Publishing Ltd, 2024.
- [31] S. S. C. McAdam and R. J. Spiteri, "Algorithm 1057: FunC: A Minimally Invasive C++ Library for the Generation and Analysis of Univariate Lookup Tables," *ACM Trans Math Softw*, vol. 51, no. 2, p. 14:1-14:30, Jun. 2025, doi: 10.1145/3734692.
- [32] Y. Manzhos and Y. Sokolova, "A type system for formal verification of cyber-physical systems C/C++ software," *Radioelectron. Comput. Syst.*, vol. 2024, no. 1, pp. 127–142, Feb. 2024, doi: 10.32620/reks.2024.1.11.
- [33] L. Mathew, "Understanding Pure Functions: A Core Concept in Functional Programming," Medium. Accessed: Feb. 04, 2026. [Online]. Available: <https://medium.com/@linz07m/understanding-pure-functions-a-core-concept-in-functional-programming-d2189b688e00>
- [34] P. A., "BASICS OF PROGRAMMING IN THE C++ PROGRAMMING LANGUAGE," *Символ Науки*, no. 10–2, pp. 33–34, 2023.
- [35] A. Brahmakshatriya and S. Amarasinghe, "BuildIt: A Type-Based Multi-stage Programming Framework for Code Generation in C++," in *2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, Feb. 2021, pp. 39–51. doi: 10.1109/CGO51591.2021.9370333.
- [36] L. Josipović, A. Guerrieri, and P. Ienne, "From C/C++ Code to High-Performance Dataflow Circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 41, no. 7, pp. 2142–2155, Jul. 2022, doi: 10.1109/TCAD.2021.3105574.
- [37] "Javanotes 9, Section 3.1 -- Blocks, Loops, and Branches." Accessed: Feb. 04, 2026. [Online]. Available: <https://math.hws.edu/javanotes-swing/c3/s1.html>