KIT- Kalaignarkarunanidhi Institute of Technology
(An Autonomous Institution, Affiliated to Anna University Chennai)
Coimbatore-641402.

DATA STRUCTURE PROJECT
COURSE CODE : B23CST302

PROJECT TITLE: Real-Time Auction Bidding Analyzer using Sorting Algorithms

GUIDE: Ms Gokilapriya

TEAM MEMBERS	REGISTER NO
PRAGATHESWARAN	711524BCS120
PRADEEP KUMAR	711524BCS118

Abstract

Online auction platforms such as eBay, Amazon Auctions, and government tender portals require real-time monitoring and analysis of bids to ensure transparency and quick decision-making. Large numbers of bids may be placed within seconds, making manual tracking impossible.

This project, “Real-Time Auction Bidding Analyzer using Sorting Algorithms,” aims to develop a system that organizes and ranks live bids using efficient sorting techniques. Sorting algorithms such as Merge Sort / Quick Sort are used due to their time efficiency and ability to handle rapidly updating datasets.

The system accepts bidder name, bid amount, and timestamp. After every new bid, the list is automatically sorted in descending order of bid amount, ensuring the highest bidder always appears at the top. This real-time ranking helps auctioneers, buyers, and sellers make accurate decisions.

This project highlights the role of classical sorting algorithms in real-world high-speed applications and demonstrates how Data Structures can be used to solve computational problems in digital auction systems.

Introduction

In online auctions, bidding speeds have increased tremendously due to digital platforms. Thousands of bidders may compete, placing new bids every second. An auction system must:

- Track all incoming bids
- Sort them instantly
- Display updated highest bidders
- Ensure fairness and transparency

Sorting algorithms such as Quick Sort, Merge Sort, and Heap Sort are widely used to maintain dynamic lists efficiently.

This project implements a **Real-Time Bidding Analyzer** that:

1. Accepts bids from users (name + amount).
2. Stores them in a dynamic array.
3. Sorts them in descending order using a sorting algorithm.
4. Displays the **current highest bid** in real time.

AIM

To design and implement a program that analyzes live auction bids and sorts them in descending order using efficient sorting algorithms for real-time decision-making.

Objectives

- To accept and store multiple bids in real time.
- To sort bids automatically after each entry.
- To use Quick Sort / Merge Sort for efficient sorting.
- To display the highest bidder instantly.

Software and Hardware Requirements

Category	Details
Programming Language	C
Compiler	GCC / Turbo C
Operating System	Windows / Linux
Processor	Intel i3 or above
RAM	Minimum 2 GB
Storage	100 MB free space

Algorithm

Step 1: Start the program
Step 2: Input the number of bidders
Step 3: For each bidder, input name, bid amount, and timestamp
Step 4: Store details in structure array
Step 5: Apply sorting algorithm (Quick Sort / Merge Sort)
Step 6: Sort in *descending* order of bid amount
Step 7: Display sorted bid list with highest bidder at the top
Step 8: End the program

Program

```
#include <stdio.h>
#include <string.h>

struct Bid {
    char name[50];
    float amount;
};

void swap(struct Bid *a, struct Bid *b) {
    struct Bid temp = *a;
    *a = *b;
    *b = temp;
}

int partition(struct Bid arr[], int low, int high) {
    float pivot = arr[high].amount;
    int i = (low - 1);
    for (int j = low; j < high; j++) {
        if (arr[j].amount > pivot) {  // Descending order
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return (i + 1);
}

void quickSort(struct Bid arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
```

```
int main() {
    int n;
    printf("Enter number of bids: ");
    scanf("%d", &n);

    struct Bid bids[n];

    for (int i = 0; i < n; i++) {
        printf("\nEnter Bidder Name: ");
        scanf("%s", bids[i].name);
        printf("Enter Bid Amount: ");
        scanf("%f", &bids[i].amount);
    }

    quickSort(bids, 0, n - 1);

    printf("\n--- Sorted Bids (Highest First) ---\n");
    for (int i = 0; i < n; i++) {
        printf("%s - ₹%.2f\n", bids[i].name, 
bids[i].amount);
    }

    printf("\nHighest Bidder: %s (₹%.2f)\n", 
bids[0].name, bids[0].amount);

    return 0;
}
```

OUTPUT

```
Enter number of bids: 4

Enter Bidder Name: Arun
Enter Bid Amount: 5000

Enter Bidder Name: Kiran
Enter Bid Amount: 7800

Enter Bidder Name: Manoj
Enter Bid Amount: 6900

Enter Bidder Name: Devi
Enter Bid Amount: 8200

--- Sorted Bids (Highest First) ---
Devi - ₹8200
Kiran - ₹7800
Manoj - ₹6900
Arun - ₹5000

Highest Bidder: Devi (₹8200)
```

Result

The project successfully implements a real-time auction bid analyzer by sorting bid amounts using efficient sorting algorithms. The system accurately displays the highest bidder instantly, demonstrating the importance of sorting in online auction platforms.

Advantages

- Real-time analysis of auction bids
- Efficient sorting using Quick Sort
- Accurate identification of the highest bidder
- Useful for online auctions and tender systems
- Handles large number of bids effectively

Limitations

- No graphical user interface
- Timestamp-based tie-breaking not included
- Manual input only (not connected to live server)
- Does not detect invalid bids

Future Enhancements

- Add live server integration for real-time bidding
- Implement user authentication and bid tracking
- Add database storage
- Provide GUI / Web-based interface
- Handle auto-bidding, bid cancellation, and logs

References

1. E. Balagurusamy – Programming in ANSI C
2. Herbert Schildt – The Complete Reference (C)
3. Thomas H. Cormen – Introduction to Algorithms
4. GeeksforGeeks – Sorting Algorithms
5. Tutorialspoint – C Language Reference