به نام خدا

سیستم عامل - بهار ۱۴۰۴

استاد: دکتر مهدی کارگهی

پروژه ۱: مبانی برنامه‌نویسی سوکت

طراحان: علی عشقی موحد - سبحان علاءالدینی

اهداف پروژه

هدف اصلی این پروژه، آشنایی عمیق با فراخوانی‌های سیستمی[1] و به‌ویژه برنامه‌نویسی سوکت[2] بر بستر IPv4 است. در این پروژه، شما یک پلتفرم تعاملی و رقابتی برای برگزاری مسابقات برنامه‌نویسی تیمی[3] طراحی و پیاده‌سازی خواهید کرد که در آن تیم‌های دو نفره متشکل از یک برنامه‌نویس[4] و یک راهنما[5] با همکاری یکدیگر به حل مسائل برنامه‌نویسی در زمان محدود می‌پردازند.

این پروژه بر مفاهیم کلیدی زیر تمرکز دارد:

- برنامه‌نویسی سوکت در سیستم عامل Linux
- پروتکل‌های انتقال TCP و UDP
- روش‌های ارتباطی Unicast و Broadcast

---

[1] System Calls

[2] Socket Programming

[3] Collaborative Code Battle Platform

[4] Coder

[5] Navigator

شرح عملکرد سیستم

این پلتفرم به صورت یک سیستم کلاینت-سرور عمل می‌کند که در آن یک سرور مرکزی، مسابقه را مدیریت کرده و کلاینت‌ها (اعضای تیم‌ها) از طریق سوکت‌ها با سرور و یکدیگر ارتباط برقرار می‌کنند.

- **ثبت‌نام و تشکیل تیم‌ها:**

کاربران با انتخاب یک نام کاربری یکتا وارد سیستم شده و نقش خود را (Coder یا Navigator) مشخص می‌کنند. سیستم به‌طور خودکار کاربران را در تیم‌های دو نفره قرار می‌دهد.

- **ارسال و دریافت مسائل:**

سرور مرکزی مسائل برنامه‌نویسی را به‌صورت همگانی برای تمامی تیم‌ها ارسال می‌کند.

- **مسائل مسابقه:**

مسابقه شامل سه سوال ثابت است:

1. جمع کردن دو عدد: تابع add_numbers که دو عدد ورودی را دریافت و جمع آن‌ها را برمی‌گرداند.

```python
def add_numbers(a, b):
  return a + b
```

2. معکوس کردن رشته: تابع reverse_string که یک رشته را معکوس می‌کند.

```python
def reverse_string(s):
  return s[::-1]
```

3. بررسی پالیندروم بودن یک رشته: تابع is_palindrome که بررسی می‌کند آیا رشته ورودی از دو طرف به‌طور یکسان خوانده می‌شود یا خیر.

```python
def is_palindrome(s):
  return s == s[::-1]
```

برای هر مسئله، قالب استاندارد از پیش تعریف شده به تیم‌ها اعلام شده است تا کدهای ارسالی به زبان Python یکدست و منظم باشد. به این منظور signature توابع باید دقیقاً به صورت زیر باشد:

```python
def add_numbers(a, b):
def reverse_string(s):
def is_palindrome(s):
```

- **نحوه همکاری تیمی:**

در هر تیم، برنامه‌نویس مسئول نوشتن کد اولیه و راهنما مسئول بررسی و ارائه بازخورد است. در طول زمان تعیین‌شده، اعضای تیم از طریق کانال‌های ارتباطی خصوصی در تعامل هستند و کد نهایی توسط راهنما در قالب یک ساختار داده مشخص به سرور ارسال می‌شود:

```cpp
struct Submission {
    char team_name[50];
    char problem_id[20];  // "add_numbers", "reverse_string" or "is_palindrome"
    char code[2000];
};
```

- **اعلام نتایج:**

پس از اتمام زمان تعیین‌شده برای هر مسئله، سرور پاسخ‌ها را دریافت و به سیستم ارزیابی ارسال می‌کند. سیستم ارزیابی، کد را با تست‌کیس‌های از پیش تعریف شده صحت‌سنجی کرده و نتیجه (Passed یا Failed) را به سرور برمی‌گرداند. سرور بر مبنای سیاست امتیازدهی مسابقه، امتیاز هر تیم را مشخص کرده و نتیجه نهایی را به‌صورت عمومی اعلام می‌کند.

- **سیاست امتیازدهی مسابقه:**

امتیازات بر اساس موارد زیر محاسبه می‌شوند:

  - **سطح دشواری مسئله:**
    - جمع دو عدد: ۱ امتیاز
    - معکوس کردن رشته: ۳ امتیاز
    - بررسی پالیندروم: ۵ امتیاز
  - **زمان ارائه پاسخ:**
    - ارائه پاسخ در ۳/۴ زمان: ۲۰٪ امتیاز اضافه
    - ارائه پاسخ در نیمه زمان: ۵۰٪ امتیاز اضافه
  - **کسر امتیاز:**
    - پاسخ نادرست: عدم تخصیص امتیاز

معماری سیستم

سرور مرکزی

سرور نقش هسته‌ی پلتفرم را ایفا می‌کند و وظایف زیر را بر عهده دارد:

- **ثبت‌نام و مدیریت تیم‌ها:**
  - دریافت درخواست‌های اتصال از کاربران از طریق سوکت TCP
  - اعتبارسنجی نام کاربری‌ها و بررسی یکتا بودن آنها
  - تشکیل تیم‌های دو نفره به صورت خودکار
- **توزیع مسائل:**
  - ارسال مسائل به همه تیم‌ها از طریق سوکت UDP (Broadcast)
- **مدیریت زمان:**
  - اعلام شروع و پایان مسابقه به‌صورت پیام‌های همگانی به تمامی کلاینت‌ها
  - تنظیم تایمر برای هر مسئله (۱ دقیقه)
- **ارزیابی پاسخ‌ها:**
  - دریافت راه‌حل‌های ارسالی از طریق TCP
  - ارسال کد به سیستم ارزیابی
  - بررسی صحت پاسخ‌ها و تخصیص امتیاز
- **اعلام نتایج:**
  - ارسال نتایج هر دور و رتبه‌بندی نهایی به همه شرکت‌کنندگان از طریق UDP
- **مدیریت همزمانی:**
  - برای مدیریت ارتباطات همزمان بین چندین تیم، سرور از مکانیزم‌هایی مانند select یا poll استفاده می‌کند. این توابع سیستمی به سرور اجازه می‌دهند چندین سوکت را

همزمان نظارت کرده و به درخواست‌های متعدد (اتصال، ارسال کد، چت) بدون گیر کردن در حلقه‌های بی‌نهایت پاسخ دهد.

کلاینت

کلاینت‌ها (شرکت‌کنندگان) به دو نقش و با وظایف زیر تقسیم می‌شوند:

- **برنامه‌نویس:**
  - دریافت مسئله از سرور و نمایش آن در رابط کاربری
  - نوشتن کد برای حل مسئله
  - به اشتراک‌گذاری کد با راهنما جهت بازبینی
  - اعمال تغییرات پیشنهادی راهنما
- **راهنما:**
  - دریافت و بررسی کد نوشته شده توسط برنامه‌نویس
  - ارائه بازخورد از طریق چت خصوصی TCP
  - تأیید نهایی کد و ارسال آن به سرور

ارتباطات و پیام‌رسانی

ارتباطات در این پلتفرم به دو دسته اصلی تقسیم می‌شود:

- **ارتباطات TCP (اتصال‌گرا[6]):** TCP به دلیل تضمین تحویل و حفظ ترتیب داده‌ها انتخاب شده است تا هیچ اطلاعات حساسی از دست نرود.
- **ارتباطات UDP (بدون اتصال[7]):** UDP به دلیل سرعت بالا و مناسب بودن برای ارتباطات غیرحساس به خطا، انتخاب شده است و همچنین سربار کمتری دارد و برای ارسال پیام‌های همزمان به چندین گیرنده مناسب است.

[6] Connection-Oriented
[7] Connectionless

نحوه اجرا

راه‌اندازی سرور

سرور مرکزی با دستور زیر راه‌اندازی می‌شود:

```shell
./server <port>
```

پس از راه‌اندازی، سرور روی شماره پورت مشخص‌شده به درخواست‌های اتصال کاربران گوش می‌دهد.

راه‌اندازی کلاینت

کاربران با دستور زیر به سرور متصل می‌شوند:

```shell
./client <username> <port> <role>
```

پارامترها:

- username: نام کاربری یکتا
- port: شماره پورت سرور
- role: نقش کاربر (coder یا navigator)

راه‌اندازی سیستم ارزیابی کد

سیستم ارزیابی کد در اختیار شما قرار می‌گیرد و نیازی به پیاده‌سازی آن نیست. این سیستم با دستور زیر اجرا می‌شود:

```
Shell
python3 evaluation_server.py
```

این سیستم روی پورت ۶۵۴۳۲ درخواست‌های ارزیابی را دریافت می‌کند.

مدیریت خطاها و شرایط خاص

سیستم باید شرایط مختلف و خطاهای احتمالی زیر را مدیریت کند:

- **بررسی یکتایی نام کاربری:**
  - پیش از پذیرش کاربر جدید، سیستم بررسی می‌کند که نام کاربری تکراری نباشد.
  - در صورت تکرار، پیامی مناسب به کاربر ارسال می‌شود.
- **مدیریت زمان‌بندی:**
  - کنترل خودکار زمان باقی‌مانده برای هر مسئله
  - توقف خودکار فرایند ارسال پاسخ پس از اتمام زمان
- **قطع ارتباط ناگهانی (امتیازی):**
  - در صورت قطع ارتباط یک عضو، ارسال پیام مناسب به عضو دیگر
  - ذخیره وضعیت فعلی تیم برای بازیابی پس از اتصال مجدد

نکات و نحوهٔ تحویل

- کدهای شما می‌بایست تنها به زبان C/C++ نوشته شده و همچنین در سیستم عامل Linux کامپایل و اجرا شوند، در غیر این صورت نمره‌ای به شما تعلق نمی‌گیرد.
- توجه داشته باشید که در تمامی مراحل از فراخوانی‌های سیستمی موجود (مانند read, write, open) بجای توابع آماده (مانند scanf, printf, fopen) استفاده نمایید. در صورت استفاده از توابع آماده در هر بخش، نمره آن بخش، لحاظ نمی‌شود.
- استفاده از داده‌ساختارها (مانند vector string unordered_map) و امکانات زبان C++ و همچنین توابع انجام عملیات رشته‌ای (مانند sprintf strtol atoi) بلامانع است.
- تمامی فایل‌های خود از قبیل کد و Makefile را در پوشه‌ای تحت عنوان OS-CA1-StudentNumber که در آن StudentNumber، شماره دانشجویی شما خواهد بود قرار داده، آن را zip کنید و در سامانه بارگذاری کنید. در هنگام تحویل تنها فایل‌های بارگذاری‌شده در سامانه پذیرفته خواهند شد.
- این تمرین صرفا برای یادگیری شما طرح شده است. در صورت محرز شدن تقلب در تمرین، مطابق با قوانین درس برخورد خواهد شد.
- سوالات خود را تا حد ممکن در گروه اسکایپی درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آنها بهره‌مند شوند. در صورتی که قصد مطرح کردن سوال خاص تری داشتید، از طریق ایمیل با طراحان این تمرین ارتباط برقرار کنید.
- مواردی که در گروه اسکایپی درس و ویدیوهای کمکی توضیح داده می‌شوند جزئی از پروژه خواهند بود.

موفق باشید!