Template

```
%% Commands for TeXCount
%TC:macro \cite [option:text,text]
%TC:macro \citep [option:text,text]
%TC:macro \citet [option:text,text]
%TC:envir table 0 1
%TC:envir table* 0 1
%TC:envir tabular [ignore] word
%TC:envir displaymath 0 word
%TC:envir math 0 word
%TC:envir comment 0 0


\documentclass[sigconf]{acmart}
\citestyle{acmauthoryear}
%%
%% \BibTeX command to typeset BibTeX logo in the docs
\AtBeginDocument{%
  \providecommand\BibTeX{{%
    Bib\TeX}}}

\setcopyright{rightsretained}
\copyrightyear{2025}
\acmYear{2025}
\acmConference{SIGGRAPH Appy Hour '25}{August 10-14, 2025}{Vancouver, BC, Canada}
\acmBooktitle{Special Interest Group on Computer Graphics and Interactive Techniques Conference Appy Hour
(SIGGRAPH Appy Hour '25), August 10-14, 2025}
\acmDOI{10.1145/3721260.3733980}
\acmISBN{979-8-4007-1552-5/2025/08}

\begin{document}
```

```
%%
%% The "title" command has an optional parameter,
%% allowing the author to define a "short title" to be used in page headers.
\title{ScavengeAR: From Licensing Fees to Free — Rebuilding Mobile AR with Unity-Native Tools}
\author{Victor Leung}
\email{thevictor2225@gmail.com}
\orcid{0009-0000-0600-668X}
%%\affiliation{%
%%  \institution{Independent Developer}
%%  \city{California}
%%  \state{Mountain View}
%%  \country{USA}
```

```
\begin{abstract}
ScavengeAR, an augmented reality creature-collecting mobile game, returns to SIGGRAPH in 2025 after a six-year hiatus, featuring a modernized tech stack and refined gameplay. The reboot preserves the core player experience while eliminating costly third-party dependencies.
\end{abstract}
```

```
\begin{teaserfigure}
  \includegraphics[width=\textwidth]{scavengear_teaserbanner.jpg}
  \caption{Testing ScavengeAR's 2017 mobile build in Unity}
  \Description{A person holds a smartphone displaying an augmented reality (AR) creature over a colorful marker pattern. Behind the phone, a computer screen shows the same marker being processed in Unity with a grid of image target icons. The AR creature is a teal, cartoon-style robot with yellow antlers and big eyes, floating above text describing SIGGRAPH's AR showcase.}
  \label{fig:teaser}
\end{teaserfigure}
```

```
%%\received{20 February 2007}
%%\received[revised]{12 March 2009}
%%\received[accepted]{5 June 2009}
```

```
\maketitle
```

\section{Introduction}
ScavengeAR was SIGGRAPH's official AR mobile app from 2017 to 2019, offering attendees an engaging augmented reality experience and providing Luddy School of Informatics, Computing, and Engineering, Indiana

University Indianapolis students with hands-on opportunities in interactive media development.

In the game, SIGGRAPH attendees choose a role — Artist, Scientist, or Educator — and explore the conference venue to discover printed image targets that, when scanned through the app, spawn 3D AR creatures into their physical surroundings. Like a digital safari, players use the in-app camera to photograph each creature, capturing them into a personal library. New creatures are released each day of the conference, encouraging daily exploration and unlocking new layers of narrative and prize opportunities over time. In addition, the attendee has their own creature based on their role, which encourages real interactions with other attendees in order to catch all the creatures for the best prizes.

```
\begin{figure}[ht]
  \centering
  \includegraphics[width=\linewidth]{scavengear_threephoneview.PNG}
  \caption{ScavengeAR 2019 Los Angeles Convention Center map screen, profile screen, and role share screen.}
  \label{fig:teaser}
\end{figure}
\section{Motivation}
```

ScavengeAR ran from 2017-2019, with over a thousand daily active users during the conference. In 2020, the Covid Pandemic stopped in-person activity for a number of years, and effectively halted the app's development, leaving it in a deprecated state.

In 2025, a small volunteer team undertook a comprehensive refactoring of ScavengeAR to improve its maintainability and accessibility, in order to revive the spirit of the app to a modern audience.

```
\section{Technical Approach}
```

The original version of ScavengeAR was built using Unity 2018 and Vuforia 8 - a robust and accessible image tracking solution that was among the most practical options for AR based 2D markers at the time. Vuforia's artist-friendly pipeline utilized a parent-child GameObject hierarchy within a single-scene architecture, allowing low-code AR development and rapid prototyping in a pre-ARKit/ARCore standardization era. However, Vuforia also required licensing fees for production deployment and cloud-hosted image target storage.

For the 2025 relaunch, ScavengeAR was refactored to using Unity 2022 and AR Foundation 5 — Unity's native, cross-platform AR framework built on top of ARKit and ARCore — to support lightweight, offline 2D image tracking and reduce long-term costs, build size, and technical dependencies.

Migrating to AR Foundation required a shift to its modular, multi-scene, prefab-driven architecture. While we

traded Vuforia's simplicity, the new system offered improved performance management and a clearer separation of concerns. AR Foundation also supports full in-editor XR simulation—replacing webcam-based workflows—which proved essential for rapidly iterating on the modernized photo capture experience.

Originally built using GameSparks for backend services and a licensed data binding plugin for UI management, we replaced GameSparks with Unity's built-in serialization for offline data handling, and transitioned to an open-source MVVM Toolkit. These changes not only further reduced external dependencies and licensing costs but also simplified the codebase, making it more approachable for new contributors.

While the core functionality remains consistent with previous iterations, this overhaul ensures that ScavengeAR can continue to be a sustainable and educational project. By leveraging Unity's modern XR capabilities and open-source tools, we've positioned the app for easier updates and potential future enhancements, aligning with our goal of providing valuable learning experiences for students and enjoyable interactions for conference attendees.

\section{Art and Design}
\begin{figure}[ht]
  \centering
  \includegraphics[width=\linewidth]{scavengear_newcreatures.jpg}
  \caption{New creatures for ScavengeAR}
  \label{fig:teaser}
\end{figure}

Luddy School of Informatics, Computing, and Engineering, Indiana University Indianapolis students has historically contributed original 3D creature designs for each iteration of ScavengeAR, and they return once again with a new cast of characters. This year's creatures were developed using a traditional animation and visual effects pipeline, continuing the project's tradition of experimenting with new styles and creative direction in each version.

For the user experience, we retained the core interaction flow from ScavengeAR 2019, which has been iteratively tested and refined through multiple conference deployments. While earlier versions of the app featured playful mini-games like Triviatron and Sigglet Falls, these proved non-essential to the primary gameplay loop. Instead, our focus shifted to improving the AR photo-taking experience—updating the camera overlay to better suit modern tall-screen phones, adding a tactile shutter animation, and introducing a polaroid-style fadeout to reinforce the analog camera motif.

\section{Future Work}
\begin{figure}[ht]
  \centering
  \includegraphics[width=\linewidth]{scavengear-2018creature.png}

\caption{A 2018 ScavengeAR creature, originally featured at the SIGGRAPH Studio venue in the Vancouver Convention Centre, shown here without the mobile user interface.}
  \label{fig:teaser}
\end{figure}

Mobile AR has historically offered a more accurate representation of augmented reality by leveraging the raw camera feed to detect physical markers in the environment. At the time, Mixed Reality headsets restricted access to raw passthrough video, limiting similar capabilities. That trend is now shifting, with newer headsets offering improved access to video passthrough.

As we look ahead, a key challenge will be unifying the user experience (UX), user interface (UI), and interaction patterns across both mobile and headset-based platforms. With our latest refactor, we're well-positioned to support both moving forward.

%%
%% The next two lines define the bibliography style to be used, and
%% the bibliography file.
%%\bibliographystyle{ACM-Reference-Format}
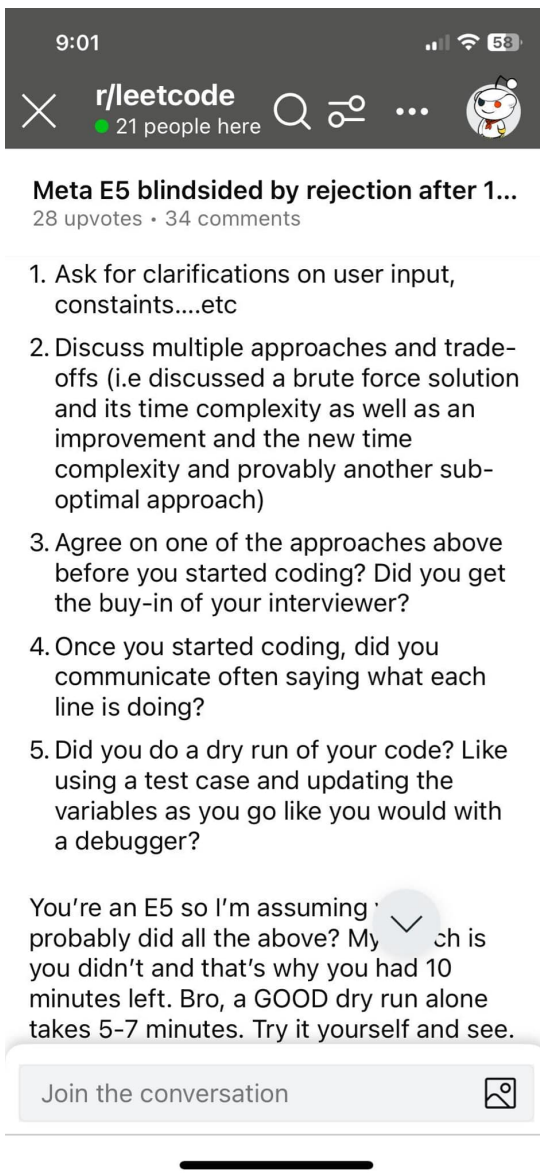%%\bibliography{sample-base}

%%
%% If your work has an appendix, this is the place to put it.
\appendix

\end{document}
\endinput

which is to be published in siggraphhappyhour25 to The ACM Publishing System (TAPS). The system will be sending 3 reminders within 120 HRS of sending the first notification to upload inputs of your paper. Please upload your files within this time frame, to avoid support delays due to last minute rush.

Please use this link to upload the zip file of your paper for processing, the instructions regarding how to prepare your zip file for submission are posted at the top of the page:

http://camps.aptaracorp.com/AuthorDashboard/dashboard.html?key=0&val=3645720d-3532-11f0-ada9-16bb50361d1f

Note: Please make sure to verify your paper details from the option "CHECK PAPER DETAILS"

provided at the mentioned link, before you upload the zip file, as that information will be printed in your paper?s HTML version.

In addition, any supplements/videos associated with your paper will need to be uploaded using a separate link mentioned below:

https://cms.acm.org/artifactSubmission/fileUpload.cfm?parent=50565B0300&id=525550020B0B51

Should you have any issues preparing or uploading the zip file for your paper, please lodge a ticket using "Contact Support" form on your TAPS dashboard; If you face any problem while adding the ticket, then you can also contact confsupport@aptaracorp.com for assistance.

Sincerely,
ACM Production

https://github.com/mdmzfzl/NeetCode-Solutions

```python
def solve_problem(inputs):
    # Step 1: Understand the Problem
    # - Parse inputs and outputs.
    # - Clarify constraints (e.g., time, space).
    # - Identify edge cases.

    # Step 2: Plan and Design
    # - Think about the brute-force approach.
    # - Optimize: Can you use dynamic programming, divide & conquer, etc.?
    # - Choose the appropriate data structures (e.g., arrays, hashmaps, heaps).

    # Step 3: Implement the Solution
    # - Use helper functions for modularity.
    # - Write clear, well-commented code.

    def helper_function(args):
        # Optional: For recursion, BFS, DFS, etc.
        pass
```

```python
    # Main logic
    result = None  # Initialize result or output variable.


    # Example logic
    # for num in inputs:
    #    result += num  # Or other computations.


    return result


# Driver code (for testing locally)
if __name__ == "__main__":
    inputs = []  # Replace with example test cases.
    print(solve_problem(inputs))
```